THE FAIRCHILD FST-1 COMPUTER

Part Number 67095031

Issued: September, 1973

FAIRCHILD SYSTEMS TECHNOLOGY

1725 TECHNOLOGY DRIVE
SAN JOSE, CA 95110

FAIRCHILD
SYSTEMS TECHNOLOGY
A DIVISION OF FAIRCHILD CAMERA AND INSTRUMENT CORPORATION

FAIRCHILD
SYSTEMS TECHNOLOGY

TABLE OF CONTENTS

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

Table of Contents (Continued)

LIST OF ILLUSTRATIONS

List of Illustrations (Continued)

List of Illustrations (Continued)

LIST OF TABLES

FAIRCHILD
SYSTEMS TECHNOLOGY

Preface

This manual describes the Central Processing Unit (CPU) and Memory System of the FST-1 computer. The Common Peripheral Interface Unit (CP1) that contains the interface cards and controllers for the I/O devices are described in the FST-1 Common Peripheral Interface Manual. Also described in the CPI manual are the interface requirements for the TTY/VKT, Tape Cassette and Line Printer. The interface cards for these items are physically located in the CPU.

REFERENCE DOCUMENTS

FST-1	Assembler Manual	Part #67094951
FST-1	Common Peripheral Interface Manual	Part #67095021
FST-1	Diagnostics Manual	Part # TBA

CHAPTER I
GENERAL SYSTEM DESCRIPTION

Section 1
General Information

1.0 INTRODUCTION

This chapter provides a general description of the FST-1 computer. Included are general information, and information covering controls and indicators.

The computer consists of a three basic subsystems as follows: (1) Central Processing Unit (CPU), (2) Memory System, as well as a memory interface, and (3) Common Peripheral Interface (CPI).

The computer system is housed in the left hand door of the Sentry system mainframe, as shown in Figure I-1-1. The memory system is located in the A0 module, while the CPU and memory interface (data bus) boards are contained in the A1 module. The A3 module provides space for the CPI boards and controllers for the I/O (input/output) peripheral devices.

In all sentry systems (except the S-100 system) the common peripheral interface module (A3) is present and provides interface for I/O peripherals such as card reads, magnetic tape units and disk storage units. In all sentry systems (including the S-100 system) interface cards for the TTY/VKT, tape cassette and line printer are contained within the CPU module. This division of CPI cards enables the elimination of the entire common peripheral interface module (A3) in the S-100 system (which does not support card readers, mag tape units or disk storage units) and at the same time supports standard options such as the VKT/TTY and line printer on all systems. Interface between the computer and tester are provided by 3 CPI cards located in the M2 module. This placement is common to all systems.

1.1 SYSTEM DESCRIPTION

The basic FST-1 system configuration is presented in Figure I-1-2. Each of the blocks shown in this diagram is discussed in progressively greater detail in the corresponding sections below.

The CPU (Central Processing Unit) executes the instructions contained in the program, performs arithmetic operations, and processes interrupt requests from the peripherals. Data is

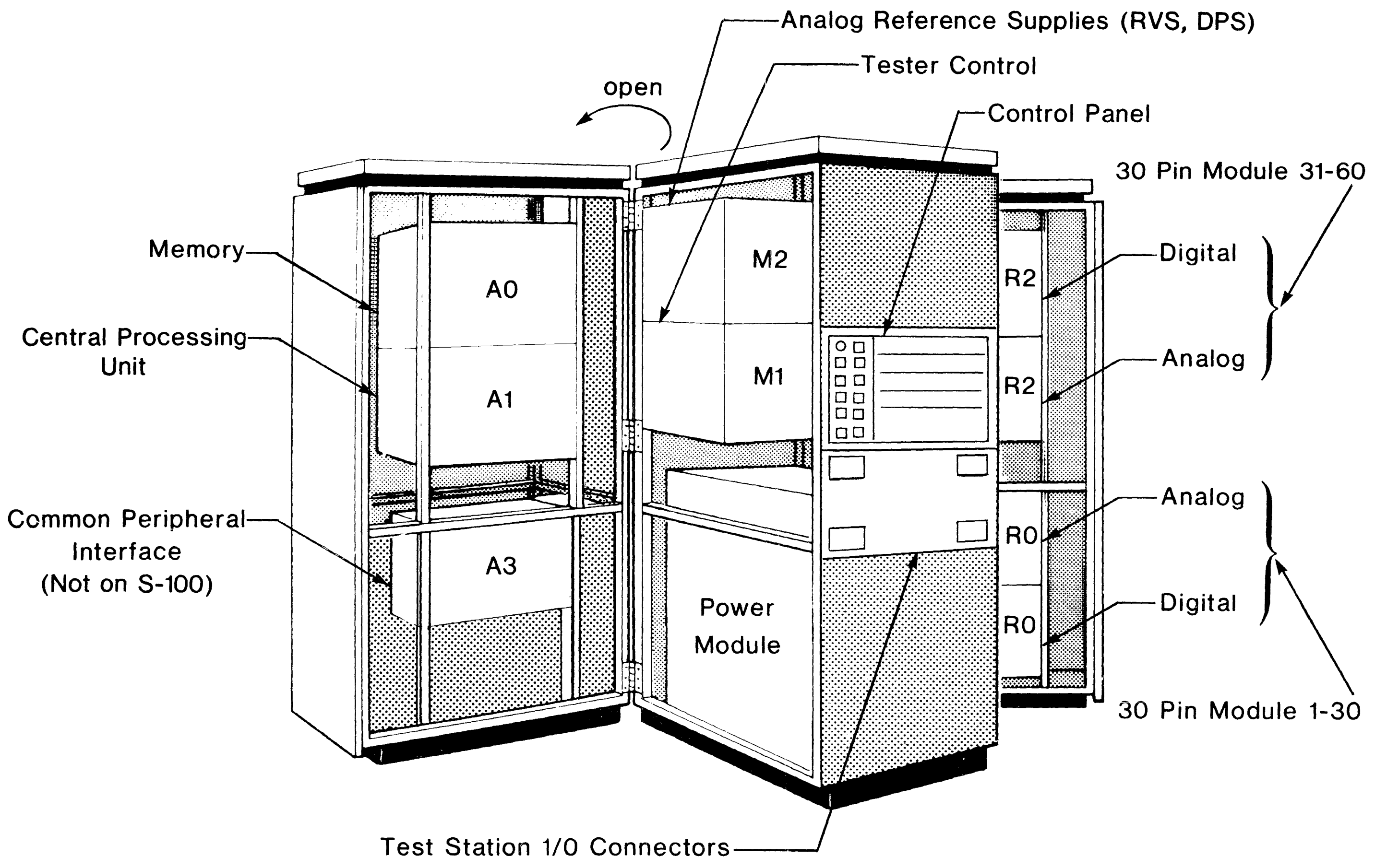


FIGURE I-1-1. SENTRY SYSTEMS MAINFRAME

transferred between the CPU and peripherals via the bidirectional N bus (also called the accumulator bus).

The program itself is stored in core memory. Two memories are shown in Figure I-1-2, designated A memory and B memory. A memory is present in all Sentry systems and is organized as an 8Kx24 bit memory -- that is, it provides storage locations for 8192 words of 24 bits each (K=1024). B memory has the same storage capacity as A memory, and, while technically an option, is present in almost all test system applications.

The memory interface contains the necessary bus control circuitry to provide and regulate access to the memory, both by the CPU and by selected peripherals. Each peripheral (except the keyboard device and line printer) can interact with the computer by either of two methods.

The first method is an interaction between the peripheral and the CPU via the bidirectional N (accumulator) bus. The CPU controls operations within the peripheral by transmitting SPU (Select Peripheral Unit) instructions. In a more restricted way, the peripheral may initiate an interchange with the CPU by means of a program interrupt (also transmitted via the N bus).

The second method is an interaction between the peripheral and memory (through the memory interface) via the bidirectional A bus (A memory bus) and/or B bus (B memory bus). This occurs without any explicit involvement of the CPU, and is called DMA (Direct Memory Access) mode.

A typical example of such a DMA operation is the reading of a series (block) of 24 bit data words from A or B memory into the peripheral. In such a case, certain control information, such as the initial memory address location and number of words, is needed prior to starting DMA mode. This information is sent from the CPU to the CPI in the form of SPU instructions, via the N bus.

The keyboard device and line printer do not connect to the computer through CPI cards and do not operate in DMA mode. Thus they do not have access to the A and B buses.

One peripheral may operate in DMA mode, even while the CPU is simultaneously transmitting SPU instructions to another. Thus, for example, the tester may be executing a DMA instruction sequence stored as a data block in B memory, while the computer is performing data analysis on previous test measurements, and logging the results onto a disc.

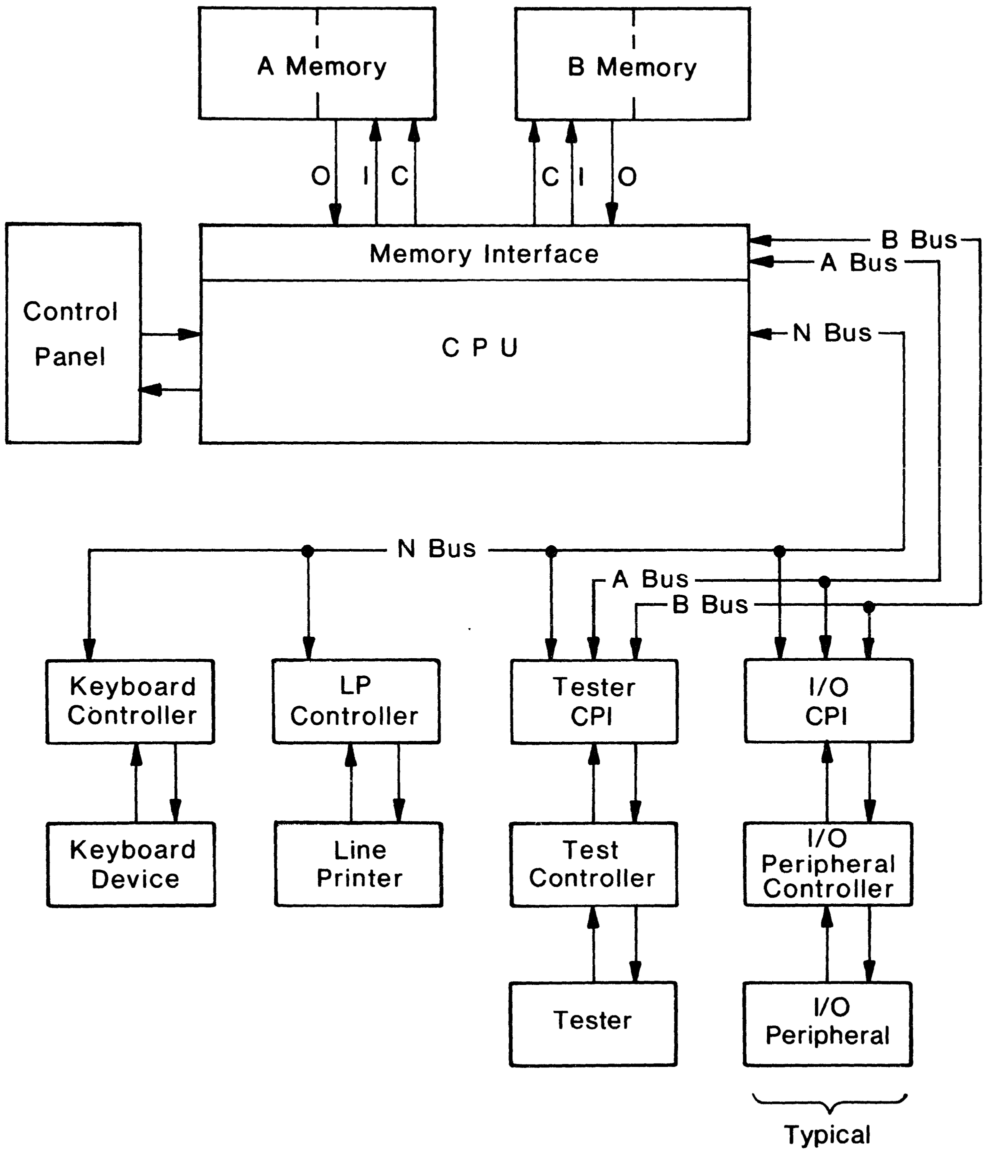


FIGURE I-1-2. BASIC FST-1 SYSTEM CONFIGURATION

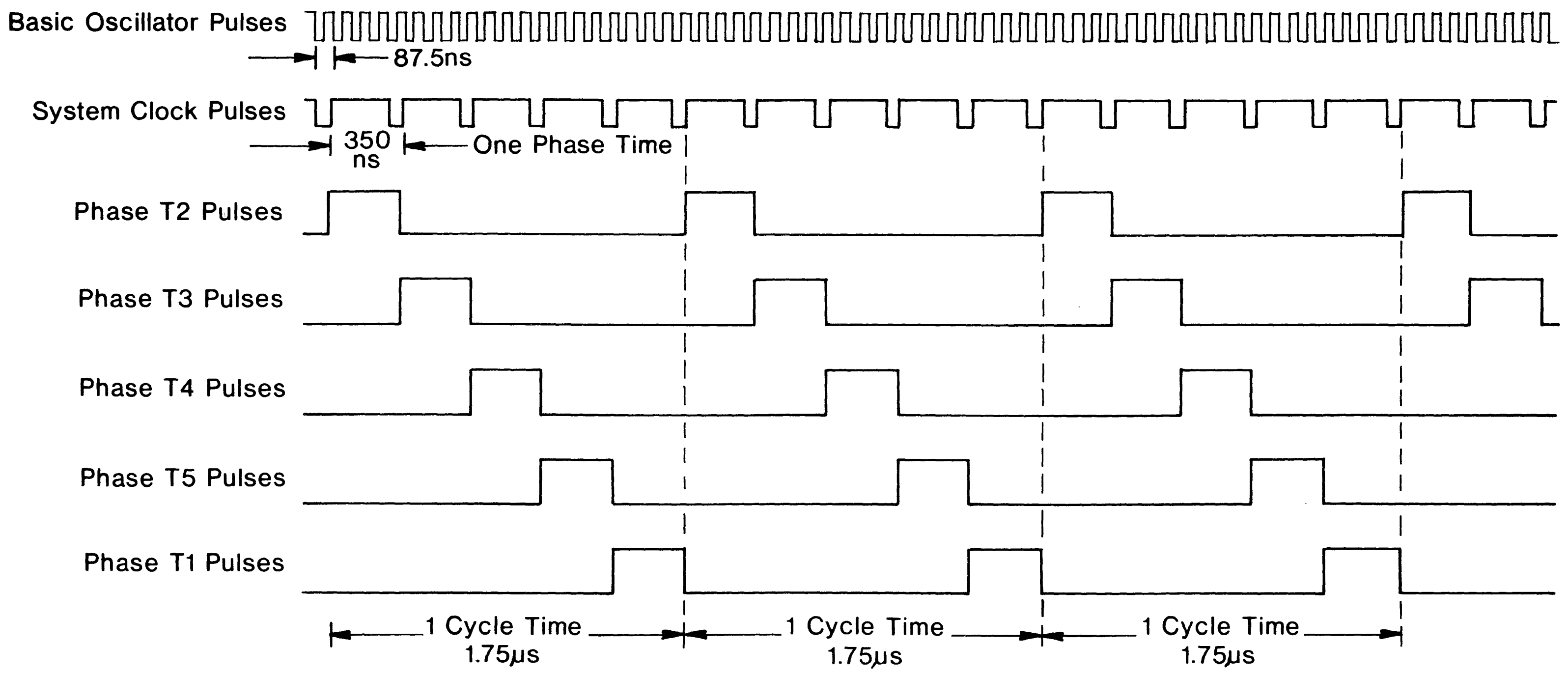


FIGURE I-1-3. BASIC FST-1 TIMING DIAGRAM

Figure I-1-2 shows one I/O peripheral as a typical case. In practice, there are three I/O devices which are commonly used in Sentry test systems. These are a magnetic tape unit, disc, and card reader.

The keyboard device may be a teletype (TTY) or a video keyboard terminal (VKT), or it may be a DICOM (magnetic tape cassette) unit with TTY and/or VKT attached.

1.1.1 Basic Timing

The computer operates on a 1.75 usec 5-phase clock cycle, which is generated in the following manner. An 87.5 nsec crystal oscillator produces a square wave output, which is fed into a divide-by-four counter to produce a train of system clock pulses 350 nsec apart. This basic 350 nsec clock rate is used, through appropriate gating, to generate five separate timing signals, designated T2, T3, T4, T5, and T1, occuring in the relationship shown in Figure I-1-3.

The basic timing circuitry is located on the CPU Clock board, slot D0 in the A1 card cage. The individual sections below contain discussions of how these five timing signals are used in the control of the various operations within the CPU and memory.

1.2 OPERATION AND MAJOR COMPONENTS

1.2.1 CPU

Figure I-1-4 is a block diagram of the FST-1, with emphasis on the functional composition of the CPU. The arrows between blocks indicate the direction of data or control signal flow. This section discusses in turn the functions of each of the blocks represented in the CPU portion of Figure I-1-4. Specific CPU hardware information is contained in chapter II of this manual.

Memory Interface

The memory interface unit comprises both temporary storage facilities and gating circuits for the transfer of all instructions and data between memory and the CPU. The memory interface unit handles 24-bit words exchanged between "A" and "B" banks of the core memory and the programam counter, the command register, and the buffer register. Also, for data exchanges between the core memory and the peripheral units, the memory interface unit performs the same service, transferring data along

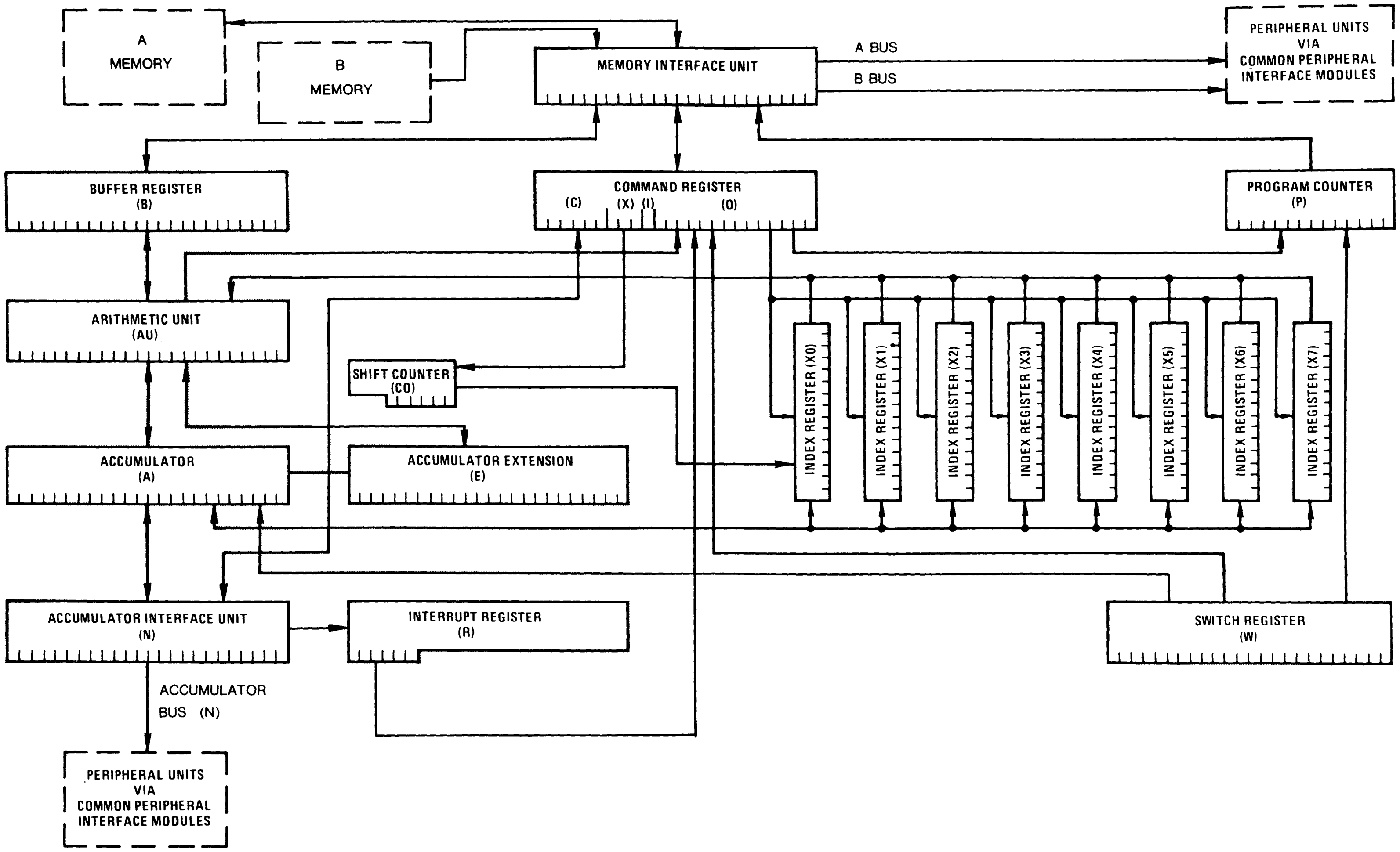


FIGURE I-1-4. FST-1 SIMPLIFIED BLOCK DIAGRAM

the A and B memory busses between memory and the common-peripheral-interface modules of the peripheral units.

Program Counter (P)

The 14-bit program counter keeps track of the address of the next instruction. As an instruction is fetched from the computer memory, the contents of the program counter are incremented by a count of one. Thus, instruction words are fetched in consecutive order from memory locations designated by the program counter.

The first address of a computer program may be manually entered into the CPU program counter by way of the switch register on the control panel. The program counter is initialized to 00100_8 by resetting the CPU from the control panel.

Certain instructions alter the contents of the program counter to branch from a consecutive-address sequence of operation. In such cases the operand-address bits replace the contents of the program counter for the first branching step. Subsequent steps again rely on the program counter for consecutive addressing of instructions until otherwise altered by another branch instruction.

The contents of the program counter are displayed on the control-panel register display and may be seen when the CPU is in the STOP condition. They designate the core-memory address of the next instruction to be fetched from memory.

Index Registers (X_0 through X_7)

There are eight 14-bit index registers, X_0 through X_7. Index registers X_1 through X_7 may be addressed by the computer program for operand-address modification. There is no index enabling bit, so index register X_0 may not be addressed for operand-address modification. It is used for program control operations requiring the comparison of two index registers (one an odd-numbered register and the other the next lower even numbered register) for the "add-one-to-index, ATX" instruction. Index register X_0 is also used to store the shift count after a "double-shift normalize, DSN" instruction.

The contents of any index register may be displayed on the control panel register display during the STOP condition by rotating the register display rotary switch to the appropriate position. During RUN, only X_0 may be selected.

Command Register (C, X. I, and O)

Computer-program instruction words are transferred from core memory to the 24-bit command register. The instruction words are then decoded to produce the various control signals that effect the commanded operation.

The command register comprises various groups of bits whose significance are explained in chapter II. The segments of the command register designated on Figure I-1-4 represent the most frequently used word format, which is used for memory reference instructions. The immediately following explanation is limited in scope and only generally relates the functional aspects of the command register to other CPU functional units. A more extensive understanding of the diversity of instruction-word usage and the roles of the command register for various types of instructions requires knowledge of computer-word formats and the various operation codes contained in chapter II.

The significance of the six "C" bits of instruction words are always the same: they represent the operational code in a binary coded octal, two digit number. These six bits are the one group that never changes in function. Deciphering of the "C" bits is the first step of decoding the rest of the instruction word and, thus, determines the significance of the other 18 operational bits of the command register.

Fourteen bits of the command register are designated "O" and represent an operand-address for memory reference operational codes. Fetching or storing of operands are based on the operand-address "O" bits. For indexed operations, the "X" bits designate one of the index registers whose contents are added to the "O" bits in the command register. For branch instructions, the "O" bits designate the next instruction and replace the contents of the program counter. For indirect addressing, the "I" bit is a "1," and in this case the "O" bit designate the address of a memory location whence 17 instruction-word bits replace the "O," "I," and "X" bits in the command register.

Although not designated in Figure I-1-4, six bits taken from the operand-address portion of the command register designate the number of shifts, and, in this case, these bits are transferred to the shift counter.

Other operand-address "O" bits in various configurations signify other operational control information and are covered in the descriptions of computer-word formats.

Input-output communications also involve the use of the command register for the processing of computer-program instructions and for peripheral-unit interrupts. Command-register/peripheral-unit operational code and status information are exchanged through the accumulator interface unit and the interrupt register.

Shift Counter (CO)

During the first phase time of any shift instruction, the 6-bit shift counter is loaded with the 6-bit number-of-shifts taken from the command register. For the remainder of the shifting operation, the shift counter is decremented one count per shift until the shift count reaches zero. Thereupon, the shifting operation terminates.

One shift instruction, "double-shift-normalize, DSN," terminates either when the shift count becomes zero, as described above, or when the two most significant bits of the double-word being shifted are not equal. In either case, the remainder of the shift register is stored in index register X after termination of the DSN shifting operation.

Buffer Register

Figure I-1-4 shows data flow in both directions between the buffer register and the memory interface unit, and between the buffer register and the arithmetic unit. All data-storage and retrieval paths are routed through the buffer register to transfer data (not used as CPU instruction words) between memory and the accumulator, the accumulator extension, and peripheral units - the last by way of the accumulator interface unit. Thus, the buffer register functions as a buffer between the arithmetic unit and memory.

However, the buffer register also performs another function for many instructions that require two operands. In such cases, one operand is already being held in the accumulator, whereas the other operand is specified in the operand-address bits of the instruction word and must be fetched from memory. During the execution of the instruction, the buffer register supplies the fetched operand to the arithmetic unit, whereas the accumulator supplies the other accumulated (held) operand. The result of the operation goes to the accumulator, whose functional description follows immediately.

Accumulator (A)

The accumulator accumulates data for arithmetic, logical manipulation, shifts, and miscellaneous processing. Results are usually stored immediately in the accumulator for subsequent use or to be transferred elsewhere by a subsequent instruction. Data (not instructions) destined for or received from peripheral units flow through the accumulator. The lower numbered 14 bits of a data word may be transferred in either direction between the accumulator and any index register (X_0 through X_7).

Data are usually transferred to the accumulator as a discrete instruction preceding the operation during which data are altered. Following the data alteration, data temporarily stored in the accumulator are then transferred by one of the paths shown on Figure I-1-4 to another unit. The contents of the accumulator may be loaded from the switch register.

Thus, the accumulator performs multiple functions as an accumulator for arithmetic, logic, shifting, and other operations and as a temporary storage unit during transfers of data.

During double-precision operations, the accumulator connects serially to the accumulator extension; that is, bit 0 of the accumulator is joined to bit 24 of the accumulator extension, whose description follows.

Accumulator Extension (E)

The accumulator extension extends the capacity of the accumulator so a double precision data word of 48 bits may be processed. Thus, bit 0 of the accumulator and bit 24 of the extension register are juxtaposed to form a 48 bit register. Data bits may be shifted in either direction during arithmetic, logical, and shifting operations. But, for the parts of arithmetic wherein data from either the accumulator or the accumulator extension are processed in or transferred elsewhere by way of the arithmetic unit, the 48-bit word is handled as two 24-bit less significant and more significant halves. This is necessary because of the 24 bit word capacity limitation of the other registers involved and of each memory location.

As in the case of the accumulator, data are usually loaded into the accumulator extension preceding a data alteration and transferred elsewhere following the operation. As shown in Figure I-1-4, such transfers are routed through the arithmetic unit.

Arithmetic Unit (AU)

Arithmetic and logical operations are processed by the arithmetic unit. Operands for these operations usually are transferred to the arithmetic unit from the buffer register, the accumulator, the accumulator extension, and in some cases from the index registers. Data transfers also are routed through the arithmetic unit. The various data flow paths are shown in Figure I-1-4.

Accumulator Interface Unit (N)

The accumulator interface unit comprises both temporary storage facilities and gating circuits for the transfer of all data and instructions between the peripheral units and the CPU. The operations involving the accumulator interface unit occur upon commands emanating from input/output instructions or peripheral unit interrupts.

Some input/output instructions transfer data, others commands, and others both command and data. Thus, data flow is shown on Figure I-1-4 to and from both the command register and the accumulator on one side and to and from peripheral units on the other side.

Interrupts from peripheral units are initiated by an interrupt request to the accumulator interface unit and, then, a transfer of an interrupt address from the peripheral unit to the CPU interrupt register through the accumulator interface unit. The interrupt address is used by the command register to address the required interrupt routine in memory.

The capacity of the accumulator interface unit is 24 bits. There are various types of communications between the CPU and peripheral units, some requiring the use of the 24-bit capacity and others only a portion thereof. These variations are covered under the explanations of input/output operations.

Interrupt Register (R)

The 6-bit interrupt register accepts and temporarily stores interrupt addresses received from the peripheral units through the accumulator interface unit. Next, the interrupt address is transferred to the command register by which it selects the memory location from which the first instruction of the appropriate interrupt routine is fetched. The interrupt routine terminates itself, whereupon program control is returned to the next instruction in the main program.

1.2.1.1 Memory Interface

The memory interface is contained on three cards (Data Bus Boards C4, C6, C8), located in the A1 card cage, with the CPU. The three cards are identical, and each effects the data busing for eight of the 24 data bits, as indicated in Figure I-1-5. The input bus lines shown as P, O, and B, come from the program counter, the command register (O = operand address), and the buffer register, respectively. The program counter is located on

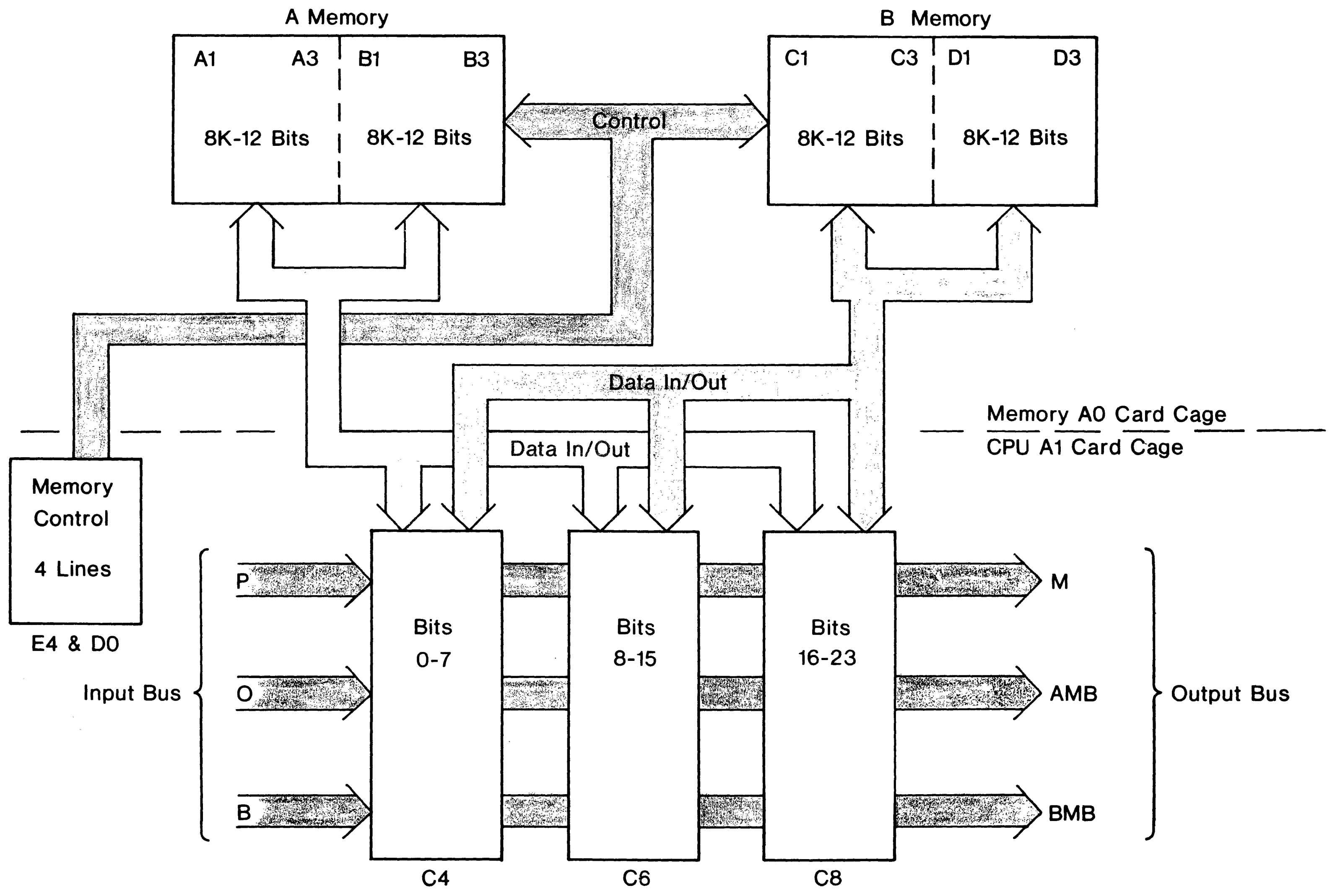


FIGURE I-1-5. MEMORY INTERFACE

the P Counter card (A1-D4), while the command and buffer registers are located on the 12 Two-Bit Slice boards.

The output bus lines designated M go to the buffer and command registers, while those labeled AMB and BMB go to the tester and I/O CPI cards, via the A and B memory buses, respectively.

Figure I-1-6 shows details of the memory interface gating functions for one bit, with attention restricted to A memory. Note that the flow of data into A memory from the A memory bus is unconditional, while data transfer from A memory to the A memory bus is gated by timing signal T1. This apparent anomaly can be resolved by careful study of Figure I-1-8, from which it can be seen that all data transfers into A memory are gated by the write initiate pulse (SWC), which occurs during CPU phase time T5. This gating is implemented, however, within the BSM itself; hence it is not represented in Figure I-1-6, which details the data bus boards only. Specific hardware information on the Memory interface is contained in chapter II section 2.

1.2.2 Memory

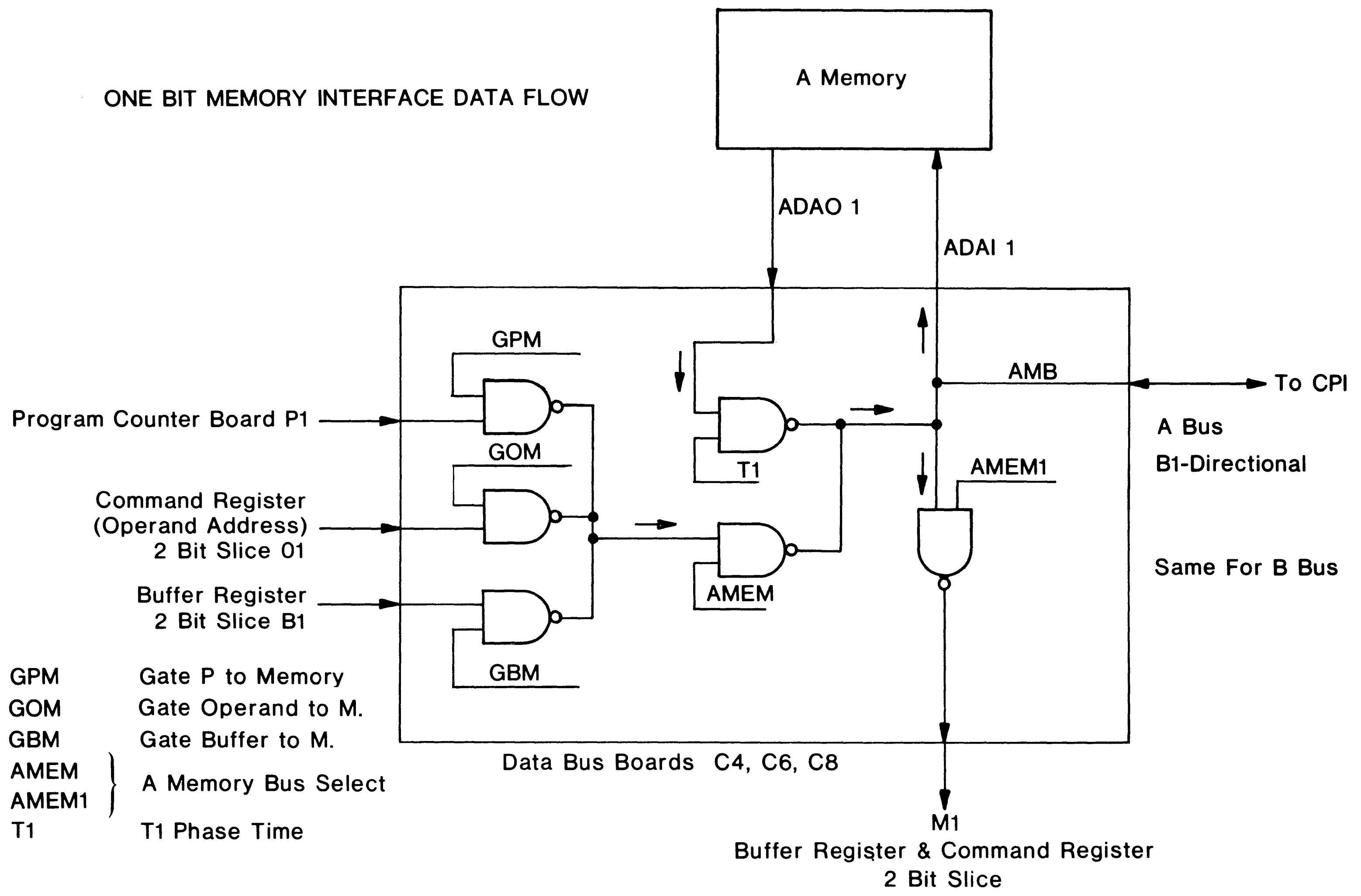
A memory and B memory are identical in size and configuration. Each is organized as an 8Kx24 bit memory, and consists of two Data Products STORE/333 magnetic core memory Basic Storage Modules (BSM), having a capacity of 8Kx12 bits each. The BSM is self-contained, providing all necessary diode addressing matrices and drive circuitry for complete memory operation.

Thirteen address bits are required to specify a given 24 bit data word location from among the 8192 (K=1024) available. A fourteenth bit selects A or B memory. The following table lists, in all, the input and output lines of either A or B memory:

Data In	24 lines
Data Out	24 lines
Address In	13 lines
SOC (Start of Cycle, Read Initiate)	1 line
RMW (Read Modify Write)	1 line
SWC (Start Write Cycle, Write Initiate)	1 line
T2F WO (Memory Inhibit, Memory Reset)	1 line

Data input and output lines are distinct from one another -- that is, a bidirectional bus is used. However, the first 13 of the 24 data input lines are also used as the 13 address lines, on a time-shared basis. This is possible because address information is entered at CPU time T3, and data information at time T5.

ONE BIT MEMORY INTERFACE DATA FLOW



GPM	Gate P to Memory
GOM	Gate Operand to M.
GBM	Gate Buffer to M.
AMEM AMEM1	A Memory Bus Select
T1	T1 Phase Time

FIGURE I-1-6. ONE BIT MEMORY DATAFLOW

The full timing relationships are shown in Figure I-1-7 for a memory read cycle, and Figure I-1-8 for a memory write cycle. Figure I-1-5 shows the overall data flow between A and B memory and the three data bus boards comprising the memory interface. The control signals originate from the Memory Control board and the CPU Clock board, are part of the CPU. Specific Hardware details on the memory are contained in chapter IV.

1.2.3 Common Peripheral Interface

Interface circuitry for the line printer and keyboard device is located in the A1 card cage with the CPU. The line printer interface occupies one card, the Line Printer Controller, in slot A1-E8. The TTY interface occupies three cards -- TTY Data Transfer (A1-E0), TTY Control & Code Error (A1-F2), and CRT Interrupt (A1-F4). These same three cards are used as interface for the VKT and tape cassette unit.

All other I/O interfaces are contained on the Common Peripheral Interface (CPI) and I/O controller cards found in card cage A3. The controllers are individually designed to match the interface requirements of the particular I/O device used.

The CPI boards are identical for all peripherals which use them. They consist of a set of three boards (designated CPI1, CPI2, and CPI3) for each peripheral.

Detailed discussions of the CPI and I/O cards contained within the CPU are contained in the FST-1 Common Peripheral Interface Manual, Part #67095021.

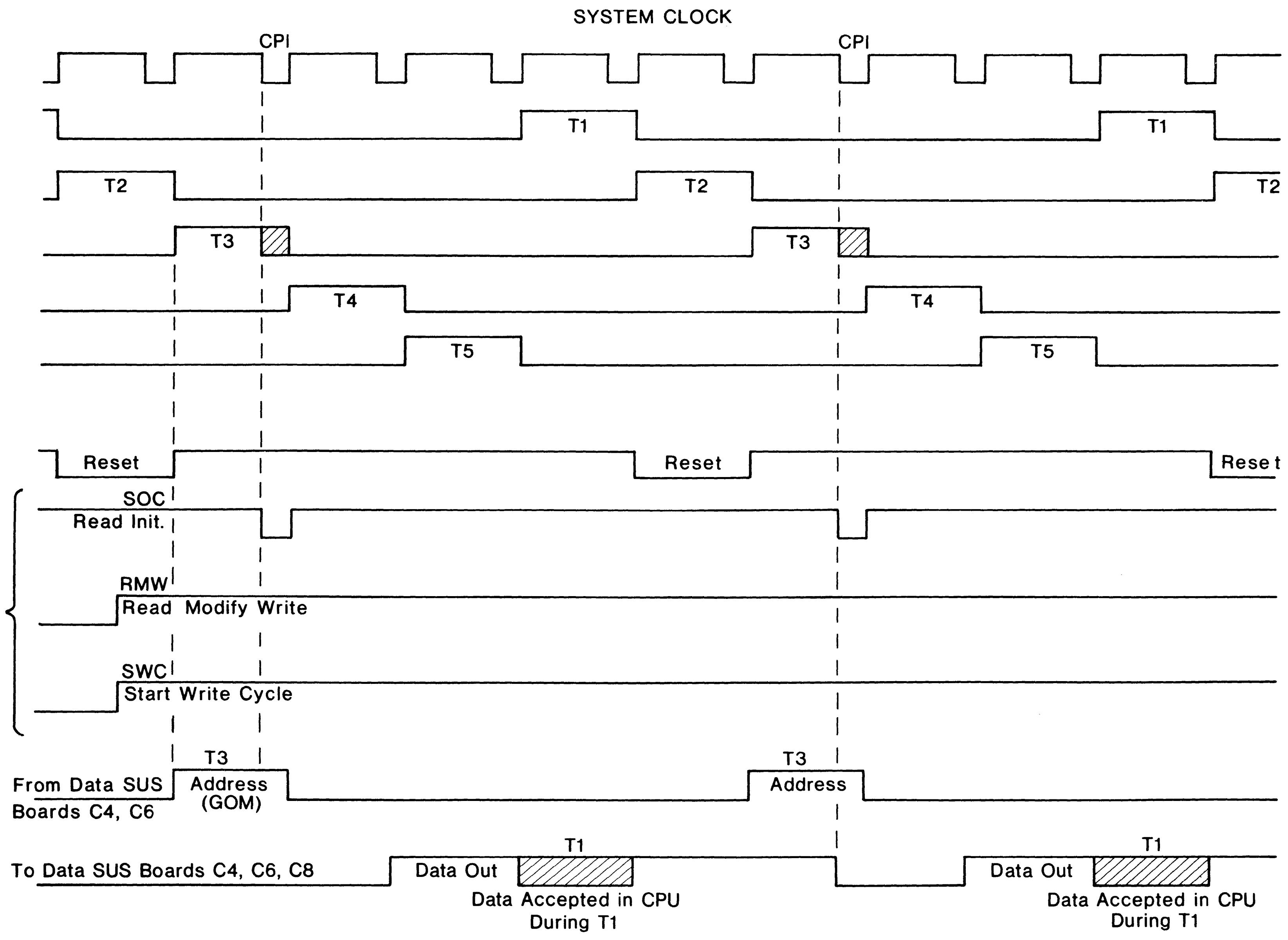


FIGURE I-1-7. MEMORY READ CYCLE

FAIRCHILD SYSTEMS TECHNOLOGY

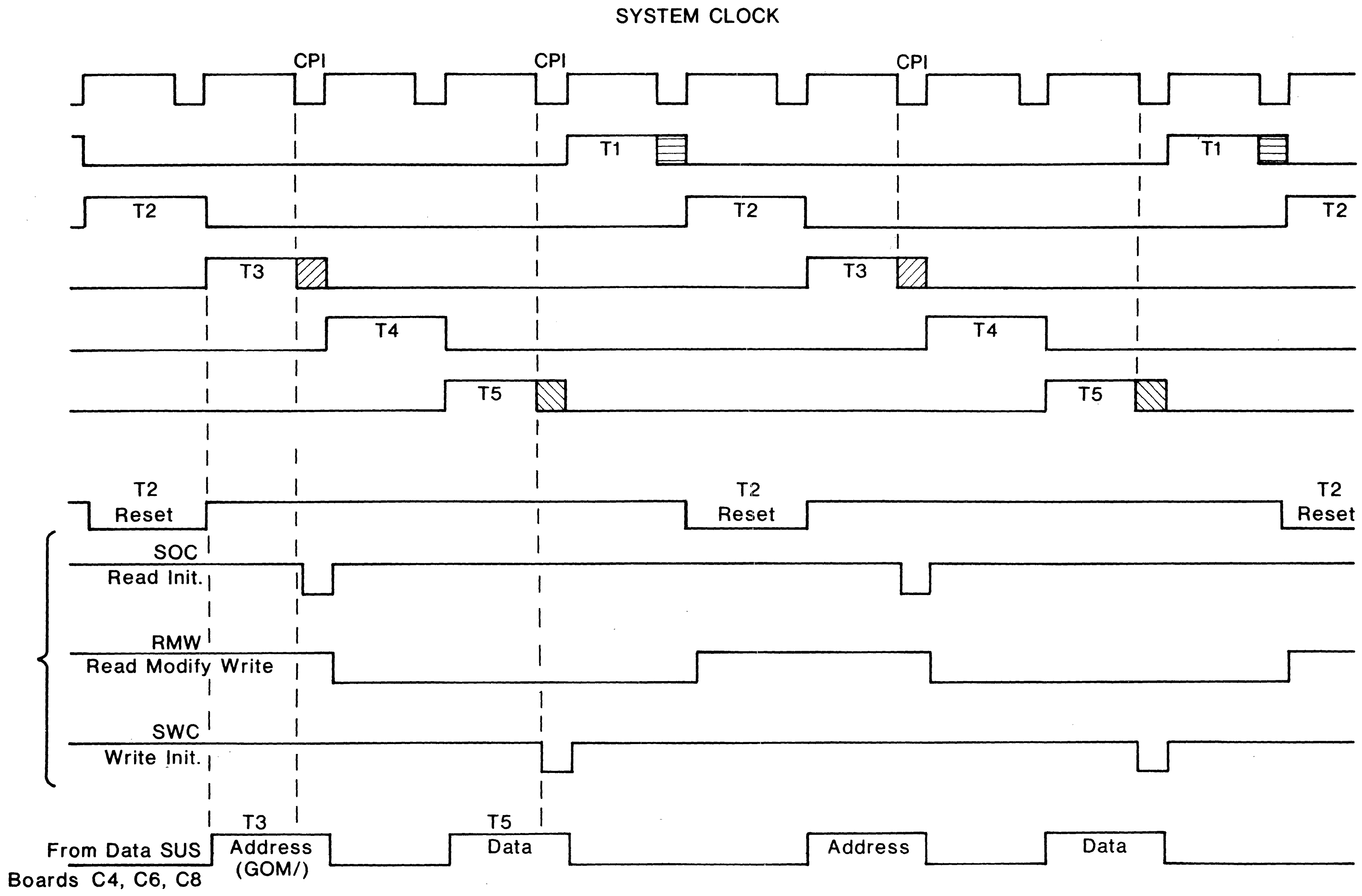


FIGURE I-1-8. MEMORY WRITE CYCLE

Section 2
Controls And Indicators

2.1 GENERAL

Table I-2-1 provides a description of all controls and indicators for the FST-1 computer. This includes all front panel controls and indicators and switches/indicators mounted on individual circuit cards contained within the CPU. Figure I-2-1 is a front panel view of the FST-1 computer.

TABLE I-2-1. CONTROLS AND INDICATORS

CONTROL	FUNCTION
START Pushbutton/ Indicator	Causes the CPU to start executing program instructions, beginning with the instruction currently held in the command register and illuminates the START indicator. While the START indicator is lit, all CPU control switches are disabled except STOP, SIC, and SMC. If SMC (single memory cycle) or SIC (single instruction cycle) are in the up position, execution is limited to single operation as selected.
STOP Pushbutton/ Indicator	Halts program processing at the termination of the instruction currently being executed, turns off the START indicator, and turns on the STOP indicator. While the STOP indicator is lit, all console control switches are enabled.
RESET Pushbutton	Resets the program counter to 00100 , and clears any interrupts which may be in process.
LOAD CR Pushbutton	Causes the card reader to read a single card (in binary read mode). The binary data on the card is loaded into 40 consecutive core memory locations beginning with address 00100 . This control is primarily used to read the first card of the card object Loader program (8 card BOOT) or the one card Disc Bootstrap Program.

TABLE I-2-1. CONTROLS AND INDICATORS (Continued)

CONTROL	FUNCTION
LOAD MT Pushbutton	Loads one record of the Magnetic Tape DOPSY System Bootstrap program into 40 consecutive core-memory locations beginning with address 00100 .
Switch Register Switch (0 thru 23)	These 24 switches allow manual entry of a 24-bit word. The contents of the Switch Register are loaded into the program counter by the LDP switch, into the command register by the LOAD CR control, into the A register by the LDA switch, or by executing an RSR instruction.
Register Displays (0 thru 23)	Displays contents of the respective operating register as selected by the register display selector rotary switch.
LDA Switch	Loads the contents of the switch register into the A register. The SIC or SMC switch must also be on (up) when attempting this operation.
LDP Switch	Loads the contents of the switch register into the Command register. The SIC or SMC switch must also be on (up) when attempting this operation.
STW Switch	Loads the contents of the switch register into the Buffer register and the core-memory location specified by the current content of the Program Counter. When the store operation is completed, the program counter is incremented by one. Thus, information in sequential memory addresses may be stored by repeated operation of the STW switch. The SIC or SMC switch must also be on (up) when attempting this operation.
EXAM Switch	Loads the contents of the core-memory location specified by the current contents of the Program Counter into the Buffer register. When the examine operation is completed, the Program Counter is incremented by one. This, the contents of sequential core-memory addresses may be examined by repeated depression of the EXAM switch. The SIC or SMC switch must also be on (up) when attempting this operation.

TABLE I-2-1. CONTROLS AND INDICATORS (Continued)

CONTROL	FUNCTION
CLU (command lock-up) Switch	Locks the current command in the command register. The effective memory address is formed by joining bits 12 and 13 from the command register with bits 0 thru 11 from the program counter. The P counter advances by one after each execution of the command. This switch, when used with the SIC and START switches, affords an alternate means to load manually or to examine consecutive core-memory locations, one at a time, with either the STA or LDA instruction, respectively, in the Command register. It may be used also to clear core memory by loading a STA instruction in the command register, zero in the A register, and then pressing START.
SMC Switch (SINGLE MEMORY CYCLE)	Halts the CPU at the end of the current memory cycle of operation. Repeated depression of the START switch steps the program one memory cycle at a time so the contents of the various register displays and indicators may be examined.
SIC Switch (SINGLE INSTRUCTION CYCLE)	Halts the CPU at the end of the last memory cycle of the program instruction being executed. Repeated depression of the START switch steps the program one instruction at a time.
Console Switches	The six console sense switches allow manual control of the execution sequence of any program that contains appropriate Branch on State (BOS) instructions. The state of each switch may be individually tested with a BOS instruction. The switches have particular significance during diagnostic procedures and their use is explained fully in the diagnostics manual.

TABLE I-2-1. CONTROLS AND INDICATORS (Continued)

Register Display Selectors, Rotary Switch

POSITION	FUNCTION
E (E-Register)	Selects and displays the contents of the 24-bit extension register on the register display indicators. This register is an extension of the accumulator register and is used with double-precision arithmetic instructions such as DADD, DSUB, MUL, and DIV.
A (A-Register)	Selects and displays the contents of the 24-bit accumulator register on the register display indicators. The accumulator is the main arithmetic register for such operations as ADD, SUB, MUL, and DIV, as well as the logical operations of AND and OR. It also serves as the input/output register for the transfer of data under program control.
C (C-Register)	Selects and displays the contests of the 24-bit command register on the register display indicators. In the idle state, the command register stores the next instruction word.
B (B-Register)	Selects and displays the contents of the 24-bit buffer register on the register display indicators. All information written into or read out of core memory from the CPU during the execute phase is temporarily held in the buffer register. This information can thus be monitored by the operator using the STW and EXAM switches while the SIC (single instruction cycle) or the SMC (single memory cycle) switch is on.
P (P-Register)	Selects and displays the contents of the 14-bit program counter on the register display indicators (bits 0-13). In the STOP state, the program counter holds the core memory address of the next instruction word that will be loaded into the command register, if the current instruction is not a branch instruction.

TABLE I-2-1. CONTROLS AND INDICATORS (Continued)

POSITION	FUNCTION
S (Control States)	Selects and displays the contents of 8 control states and programmable flip-flops. The 8 flip-flop (1 2 3 4 5 6 7 8) are displayed by bits 0 thru 7. Each of the eight bits indicates the state of its corresponding flip-flop. The state of a programmable flip-flop may be used to automatically control the execution sequence of any program that contains appropriate BOS instructions. An illuminated lamp indicates its corresponding programmable flip-flop has been set to the "1" state by a SST instruction. Each flip-flop can be reset to the "0" state, turning its lamp off, with a RST instruction. The state of each programmable flip-flop may be individually tested with a BOS instruction.
X0 thru X7 Registers	Selects and displays the contents of the appropriate 14-bit index register.
TIF	When executing any instruction, the TIF (time-of-instruction-fetch) flip-flop will be set, illuminating the TIF indicator, while the CPU is in the instruction-fetch cycle.
TOF	When executing any memory-reference instruction, the TOF (time-of-operand-fetch) flip-flop is set, illuminating the TOF indicator, while the CPU is in the operand-fetch cycle.
TEX 1	The TEX 1 (time-of-execution, phase 1) flip-flop is set, illuminating the TEX 1 indicator, for at least one memory-cycle time while executing any instruction that requires two or more memory-cycle times.
TEX 2	The TEX 2 (time-of-execution, phase 2) flip-flop is set, illuminating the TEX 2 indicator, for one memory-cycle time while executing any of the instructions AOM, SOM, MUL, or DIV.

TABLE I-2-1. CONTROLS AND INDICATORS (Continued)

POSITION	FUNCTION
TV	The TV (timing-for-variable-length-shift) flip-flop will be set, illuminating the TV indicator while any shift instruction with a non zero shift count is being executed.
IEN Interrupt Enable	The interrupt-enable flip-flop is set (and the IEN indicator lighted) as the result of executing an IEN instruction. The flip-flop may be reset, turning off the indicator, by executing an IDA instruction, by executing a priority interrupt, or by pressing the RESET pushbutton.
DBU	The disc-busy flip-flop is set, (lighting the DBU indicator) when the disc is performing an operation such as read, write, or parity check. The flip-flop is reset, turning the indicator off, when the operation is completed.
DER	When illuminated, indicates that a disc parity-check error has been detected.
MTB	When illuminated, indicates that the Magnetic Tape Unit is Busy.
MER	When illuminated, indicates that a mag tape read or write error has been detected.
INP	Then input-pending flip-flop is set, lighting the INP indicator, by an ION instruction for the VKT or TTY. This indicator is a visual indication only to the operator that the program is expecting data from that input device. The flip-flop is reset, turning the indicator off, by an IOFF instruction for the VKT or TTY, or by depressing the RESET switch. The state of the flip-flop cannot be tested; hence, it cannot control the program-execution sequence.

TABLE I-2-1. CONTROLS AND INDICATORS (Continued)

POSITION	FUNCTION
Status Register Flip-Flop Indicators (bits 19-23)	Each of the five lamps indicates the state of its associated status-register flip-flop. The mnemonic definition of each indicator follows:

Mnemonic	Definition
GT	Greater than
EQ	Equal
LT	Less than
BE	Bit equal
OV	Overflow

The indicators GT, EQ, LT, BE, and OV are lighted (with the associated flip-flop set) in various configurations after executing one of the instructions CAM, ATX, SPU, or BRU (BRU with indirect bit set). The indicators affected by each instruction are shown below; refer to the detailed description of each instruction to interpret the meaning of each indicator for that specific condition.

Instruction	Used
CAM	GT,EQ,LT,BE
ATX	GT,EQ,LT,(ignore BE state)
SPU	GT,EQ,LT,BE
BRU	GT,EQ,LT,BE,OV

POSITION	FUNCTION
OV	The overflow flip-flop will be set and the OV indicator lighted, in addition to a BRU instruction, by one of the following conditions: If the accumulator overflows as the result of executing an ADD, SUE, DADD, DSUB, or DTC instruction; by executing the appropriate SST instruction. The overflow flip-flop can be reset, turning off the OV indicator by executing the appropriate RST instruction.
P-COUNT	Defines P Register contents for rotary switch position P.

TABLE I-2-1. CONTROLS AND INDICATORS (Continued)

POSITION	FUNCTION
OPERAND	Defines the instruction word memory address field.
(memory address field)	
O P X I OPERAND	Define command register fields: operation code (O P), index (X), indirect (I), and operand (OPERAND) fields.
(placards) C P S	Idenfiby placards which apply when register selector switch is set for Command Register (C), Program counter(P), and status(S) respectively.
START Indicator	Illuminates when CPU is executing program instructions.
STOP Indicator	Illuminates when CPU program instruction execution has been halted.
IER Indicator	Illuminates if an illegal instruction has been detected by the CPU.

The following switches are located on boards within the CPU.

SWITCH	LOCATION	FUNCTION
A Memory Disable	Memory Control A. Slot E4. Top switch.	In the down position disables A Memory. Red indicator lamp will be illuminated.
B Memory Disable	Memory Control A. Slot E4. Bottom switch	In the down position disables B Memory. Red indicator lamp will be illuminated.

TABLE I-2-1. CONTROLS INDICATORS (Continued)

SWITCH	LOCATION	FUNCTION
TTY/CRT Transfer	TTY/CRT Data transfer baord slot F0. Two switches.	Both switches in the down position will provide communication with the VKT. Both switches in the up position will provide communication with the TTY.

FAIRCHILD
SYSTEMS TECHNOLOGY

CHAPTER II
CENTRAL PROCESSING UNIT

Section 1
General Information

1.0 INTRODUCTION

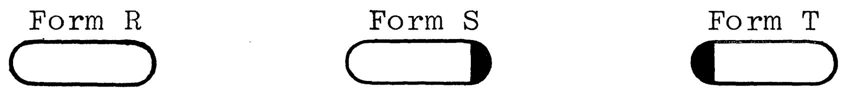
This section describes the circuitry found on each of the printed circuit cards contained in the CPU and the memory interface. These cards are all located in the A1 card cage.

Table II-1 shows the layout of the A1 card cage. The CPU occupies positions A0 through E6, except for C4, C6, and C8, which house the three data bus boards comprising the memory interface. Positions E8 through F4 contain the line printer and VKT/TTY controllers, which are described in chapter IV. G0 holds the load board for the accumulator bus.

1.1 NOTATION CONVENTIONS

Each of the following descriptions use a schematic diagram as a reference. Certain drafting conventions are used to specify connecting points on a single board, and also to categorize inputs and outputs to other boards. As an example refer to schematic 97166001-04 (Two Bit Slice, CPU).

This schematic is reproduced on two separate pages. Each page is divided into nine zones, comprising a 3x3 matrix, indexed by the letters A, B, C and numbers 1, 2, 3 at the margins, much as in an ordinary road map. Notice the presence of the rotundoid figure, in the following three forms:



Form R represents an input from another board or location not on this board. Inside the rotundoid is the mnemonic, or abbreviated signal name. To the left or above the rotundoid is the number of the specific pin in the card connector to which that signal is wired. As an example, in zone A3 on page 1 the SPU signal (decoded Select Peripheral Unit command signal) enters this card on connector pin B0 and goes to IC package F1 pin 9 (also to E1 pin 10).

Form S represents an output whose sole point of origin is on this board as shown. For example, page 2 zone C3 shows the signal CAON (the full adder "carry bit"), which is generated at A6-8 and leaves this board on connector pin H2, on its way to the adjacent (more significant) Two Bit Slice card.

Form T represents an output which occurs in a wired - OR configuration, with other points of origin located on other boards. For example, on page 2 zone B1, F8-2 is shown generating DISN1, which exits this board on pin J9. DISN1 is an output to a display on the control panel, which is also fed from other sources.

Connections between points on this board are indicated in two ways. Signal paths to points on the same page are represented simply by specifying the zone in which to look. For instance (page 2 zone C1), B8-6 (MXN/) connects to C6-7 in zone A2. Sometimes more than one connection is indicated. SN/, for example (page 2 zone C2), generated at A5-9, goes to two separate points in zone B2 (B1-5 and C1-10), in addition to its output from the card at connector pin G1. Similarly (page 2 zone C3), CAOM/, generated at B5-8, goes to A6-13 in zone B3 and also to A5-12 in zone B2.

Connections between points shown on two different pages of a schematic are indicated by a rectangular box of the form

P	Z

where P is the page number and Z is the zone where the connecting point can be found. For example (page 2 zone C2), SN, generated at A5-10, goes to E1-6 in zone A3 on page 1.

Signal names ending in / (e.g., BN/) represent the complement or logical negative of the corresponding mnemonic (i.e., BN). An older notation uses a suffix F in the same manner. Although this latter notation persists in some schematics, it is now obsolete.

TABLE II-1-1. A1 Card Cage

CARD SLOT	CARD NAME		PART NO.
A0	TWO BIT SLICE	(BITS 0, 1)	97166001
2	TWO BIT SLICE	(BITS 2, 3)	97166001
4	TWO BIT SLICE	(BITS 4, 5)	97166001
6	TWO BIT SLICE	(BITS 6, 7)	97166001
8	TWO BIT SLICE	(BITS 8, 9)	97166001
B0	TWO BIT SLICE	(BITS 10,11)	97166001
2	TWO BIT SLICE	(BITS 12,13)	97166001
4	TWO BIT SLICE	(BITS 14,15)	97166001
6	TWO BIT SLICE	(BITS 16,17)	97166001
8	TWO BIT SLICE	(BITS 18,19)	97166001
C0	TWO BIT SLICE	(BITS 20,21)	97166001
2	TWO BIT SLICE	(BITS 22,23)	97166001
4	DATA BUS	(BITS 0-7)	97340203
6	DATA BUS	(BITS 8-15)	97340203
8	DATA BUS	(BITS 16-23)	97340203
D0	CPU CLOCK		97166004
2	CONTROL FF		97166003
4	P COUNTER		97166005
6	INSTRUCTION DECODE		97166008
8	GATE CONTROL 1		97166006
E0	GATE CONTROL 2		97166007
2	MISC REGISTERS		97166009
4	MEMORY CONTROL A		97340201
6	MEMORY PARITY		97166011
8	LINE PRINTER CONTROLLER		97166124
F0	TTY DATA TRANSFER		97340202
2	TTY CONTROL & CODE ERROR		97166102
4	CRT INTERRUPT		97340205
6	NOT USED		
8	NOT USED		
G0	M1 LOAD BOARD (OPTION)		97340204

Section 2
Individual Board Description

2.0 GENERAL

The following paragraphs provide descriptions of each individual board within the CPU. All descriptions are referenced to the appropriate logic diagrams contained within the Sentry Systems Reference Manual.

2.1 Two–Bit Slice (97166001–04, 2 pages)

The CPU contains twelve identical Two-Bit Slice boards. These twelve boards, as a group, contain the accumulator interface and the arithmetic unit, as well as most of the registers in the CPU.

Specifically, the following functional blocks of Figure I-1-4 are implemented on these boards:

Accumulator Interface
A (Accumulator) Register
E (Extension) Register
Arithmetic Unit
Buffer Register
Command Register
Index Registers

The organization of the twelve boards is such that each board implements two bits of each unit from the above list. All units have 24 bits, except for the index registers, which have 14 bits each. The most significant bits are at the C2 end of the 12-board array, while the least significant are at the A0 end.

Of the two bits on any given board, the more significant is represented with a suffix N, and the less significant with a suffix M. In the following discussion, these two bits will be called simply the N bit and the M bit, respectively.

Shown on page 1, zone B3, is the A register, consisting of flip-flops B2 and inverters C2. Outputs for the M and N bits are at C2-8 and C2-6, respectively. Data inputs, gated by the signal GCPA (gate clock pulse to A register) are SAM on B2-4 and SAN on B2-13.

Since the A register can be loaded from several sources, the signals SAM and SAN must be assembled by the logic shown in zone A3 -- F1, A3, F2, and D1 for SAM and E1, C1-11, D2-6, E2, and D2 for SAN.

In a similar manner, the E register consists of flip-flops B3 (zone B3) and the associated inverters shown, with inputs SEM and SEN/ gated by GCPE. SEM and SEN/ are assembled by F3, B1-3, and E3 (zone A2).

In analogous fashion, the command register is implemented by flip-flops C4 and B4, with inputs SOM/ and SON/ assembled at E4 and D4 (zone A1), respectively. Note that the M and N bits are gated into the command register by (respectively) two separate signals, GCPO1/ and GCPO2/. This notation represents that the gating signal is one of several clock pulses, all generated on the Gate Control 1 board, according to which bit of the command register is involved.

The buffer register consists of flip-flops C5, with inputs SBM/ and SBN/ gated by GCPB. SBM/ and SBN/ are assembled at D3 (zone A1).

The eight index registers are implemented by D6 (M bit) and E6 (N bit). Each of these two IC packages contains eight bits of storage, organized as eight words of one bit each. Address information is entered at pins 4, 5, and 6, in the form of signals X15, X16, and X17, respectively, which comprise the X field, taken from bits 15, 16, and 17 of the command register. M and N bit data inputs are the signals SXM and SXN, assembled by the logic shown immediately below the index registers. Since the index registers are only 14 bits in length, this circuitry is unused on five of the twelve boards.

At the top of page 2 (zone C3), the signal AZERO is generated. This signal is shown within a form T rotundoid figure (see Notation Conventions, Section 3.1), since it is OR-tied with outputs from 11 other boards. AZERO is true if and only if all bits of the A register are zero. A ZERO in turn contributes to the BAT (Branch A register Test) command signal.

The remainder of the upper half of page 2 contains the circuitry for the arithmetic unit. A5 is a full adder, with data inputs UAM and UBM (M bit), and UAN/ and UBN/ (N bit). The carries are generated at B5 and A6, with the carry from the next less significant board CAO(M1) entering at connector pin F2, and the carry to the next more significant board CAON leaving at pin H2.

The data inputs to the adder are collected by the multiplexers B7, A7, and C6. B7 and A7 are B-input multiplexers. The A inputs (UAM and UAN/) come from one of eight sources, as selected by the three bits GUA0, GUA1, and GUA2. C6 is a dual 4-input multiplexer. The B inputs (UBM and UBN/) come from one of four sources, as selected by the two bits GBUBF (more correctly known as GBUB1/) and GBFUBF (GBUB0/).

A similar pair of multiplexers, C7 and D7, routes the contents of various registers to the control panel for display purposes. One of five registers is selected, under the control of the three bits GD1S0, GD1S1, and GD1S2.

The ZERO signal (zone C2), like AZERO, is a composite of signals from all twelve boards. ZERO is true if and only if all outputs of the adder are zero. This signal is used to set the EQ indicator, when appropriate.

Below the ZERO circuit are two OR-gates, which generate the ASA signals for the M and N bits. These signals are high when either of the corresponding bits of the A register or the adder output are one. ASA signals are used to set the A register, when the OR signal (page 1, zone A3) is high.

The contents of particular index registers can be gated (by GZUB) into the adder (through the "B" multiplexer) by the circuit shown below and ASA circuit (zone B1). This is done during the ADD-TO-X operations in which the contents of adjacent index registers are compared.

Gates E8 and D8 (zone B1) and the associated inverters of F7 and F8 comprise the accumulator interface. D8-8 and D8-6 drive the M and N bits of the N (accumulator) bus, respectively. The signal GAN gates the contents of the A register onto the N bus, and similarly, GCRN gates the command register to the N bus.

At the bottom of the page, C8 drives BEQ/, the "Bit Equal" signal. BEQ/ is low if any pair of corresponding bits in the A register and buffer register are both one.

2.2 Data Bus (97340203—04, 2 pages)

The A1 card cage contains three identical Data Bus boards. These three boards, as a group, comprise the memory interface. The basic layout and operation of the memory interface are described in Section 1.2.4.

Eight bits of the 24-bit interface are implemented on each board -- bits 0-7 at card slot C4, 8-15 at slot C6, and 16-23 at slot

C8. Of the eight bits on each board, page 1 of the schematic shows the first four, and page 2 shows the second four.

At the extreme left are the inputs from the CPU. These are designated B, P, and O, for the buffer register, program counter, and the operand field of the command register, respectively. The numbers in parentheses are the bit numbers from the register indicated. For example, B(1,9,17) means buffer register bit 1 on board 1, bit 9 on board 2, and bit 17 on board 3.

The contents of the buffer register are gated by GBM (gate buffer to memory), the program counter by GPM, and the operand field by GOM.

While the buffer register has 24 bits, the program counter has only 14. Also, although the command register has 24 bits, the operand field, which is used here, has only 14. Thus there are an additional 10 bits of circuitry available, as an extension of both the P and O fields, for the purpose of gating signals from the CPU onto the A and B memory buses. In practice, some of these bits are indeed used, as described in the following two paragraphs.

Bits 14 and 15 of the "extended P field" (designated X and Y) on board 2, are tied low, as are bits 16, 17, and 18 on board 3. Bits 19, 20, 21, 22, and 23, however, carry the signals BE (bit equal), LT (less than), EQ (equal), GT (greater than), and OV (overflow). These signals are also gated by GPM, which is true during computer phase time T3 when the P counter contents are used as a memory address, but also can be true during time T5, when the P counter contents are used as the 14 least significant bits of a 24 bit data word. This typically occurs in relation to branches to subroutines, and when it does, the 5 bits mentioned above become the 5 most significant bits of that data word.

The case of the "extended O field" is much simpler. All ten most significant bits are tied low, except for bit 18, to which the signal WRM is fed. Computer phase time T3 is used in place of GOM on board 3, so that, in effect, T3 gates WRM onto the A or B memory bus, and this action is actually independent of what happens to the 14 bits of the operand field itself. The function of WRM is to select the write or read operation for the memory data; WRM is true if a word is being stored, and false if a word is being fetched. However, the memory itself does not monitor bit 18 during time T3, but responds to the (14) address bits only. Bit 18 runs to the memory control board, where WRM is used as a basis for generating the WI (write initiate) and RI (read initiate) signals.

At the top left of page 1 is the signal GOP1M/. This signal is used in connection with double precision arithmetic. A double precision (48 bit) word is always stored with the least significant half at an even address location, and the most significant half at an odd location which differs only in that is has a one for bit 0. The P counter supplies the address for the even location and the most significant half word is transferred into (or from) the A register. GOP1M/ then goes low, and (during the next memory cycle) the least significant half word is transferred to (or from) the E register. This operation manifests on board 1 only; the point marked GOP1M/ is tied high on boards 2 and 3.

At the middle of the page are the (9944) drivers for the A and B memory buses. Inputs to these drivers are the 24-bit memory data output signals designated ADAO and BDAO, as well as the signals from the CPU discussed in the preceding paragraphs. ADAO and BDAO are gated by the memory select signals AMEM and BMEM, respectively, at computer phase time T1.

Outputs from these (9944) drivers are OR-tied (in pairs) to the A and B memory buses. The signals on these buses are designated AMB/ and BMB/, respectively. These signals are also carried to the memory data (and address) inputs by the (9016) inverters, and, as inputs to memory, they are designated ADAI and BDAI, respectively.

At the extreme right are the 24-bit memory data signals going to the buffer register. These signals, designated M, are taken from either the A or B memory bus, and gated by AMEM or BMEM, respectively.

2.3 CPU Clock (97166004–04, 1 page)

The principal function of this board is to generate the basic CPU timing signals described in Chapter I, Section 1.2.1. In addition, a variety of start, stop, and single cycle control functions are implemented.

At the top of the page (zone C2), a crystal oscillator, consisting of crystal Y1, transistor Q1, and the associated discrete resistors and capacitors, generates an 87.5 nsec square wave. When the enable clock signal (ENCLK) is true, this square wave becomes the input to the divide-by-four counter E7. The outputs are decoded by the AND gates at the right of the page. Of the four possible states, three are decoded at F5-6, F5-8, and F5-11. The fourth state is decoded by all the gates from D5-6 down. This signal is used as input to the 5-stage ring counter shown in the lower right corner of the page. The ring counter consists of the (9300) 4-bit shift register A7 and the flip-flop B7. The outputs from this counter are the five phase time signals T2, T3, T4, T5, and T1. Proper operation of the ring counter requires that one and only one of the five stages contain a one at any given time. To assure this, gates C7-6 and C7-8 are used. C7-6 eliminates the possibility of having all zeros in the counter, while C7-8 precludes the presence of two or more ones.

The remaining circuitry on this board serves to implement basic control functions affecting the CPU as a whole. At the top center of the schematic, built around flip-flop C3, is a circuit which assures proper CPU functioning with respect to memory access and operating mode (manual or automatic). The input signal ACCS is high if any peripheral desires access to memory. In such a case, GCPS and GRCPS are both off during phase time T2, and ACC is set, to disable them until the end of the CPU cycle (T1). In manual mode, the HALT flip-flop must be set. In this case the signal LRMAN is used to produce GRCPS, which permits manual loading of registers. KFML is not used. The output ML/ (E4-6), when low, inhibits instruction decoding, instruction fetch, and the gating of address information to memory.

B3 is the halt-run flip-flop. The circuitry to the left of B3 collects at C2-8 all the conditions which bring the CPU to a halt. The following list defines the functions of the input signals involved:

ISTOP	Not used
BAH	Branch and Halt
SMC	Single Memory Cycle
INVCR	Invalid Command
MAPER	Not Used
MBPER	Not Used
MPFL/	Not Used
STOP	Stop (push button)
SIC	Single Instruction Cycle (panel switch)
INA/	Indirect Address
IDX/	Index
AMEM/	Not Used
MABZ/	Not Used
BMEM/	Not Used
MBBZ/	Not Used

In all the above cases the current CPU cycle is finished before stopping; the halt flip-flop is set at phase time T1.

Flip-flop A3-6 (and the circuitry to its left) synchronizes the Start operation. The input KSTRT is a pushbutton and A2 is a one-shot which generates a pulse when the START button is pressed. The input XSTRT is not used.

Flip-flops A4-6 and A3-10 serve functions analogous to those of B3-6 and A3-6, respectively, but for manual register loading functions rather than run-halt. The relevant inputs are:

KLDA	Load A Register button
KLDP	Load Program Counter button
KLDCR	Load Command Register button
KSTO	STORE (memory location) button
KEXAM	EXAMINE (memory location) button

Flip-flop A5 generates the Command Register Lock-up signals CRLU and CRLU/, under the control of the STORE and EXAMINE pushbuttons, and the Command Register Lock-up switch on the control panel (KCRLU). STORE forces a STORE A operation; EXAMINE forces LOAD A. The CRLU switch serves a more general function using the C register.

2.4 Control FF (97166003—04, 1 page)

This board contains ten basic control flip-flops, and the logic needed to set or reset these flip-flops. They are as follows:

Time of Operand Fetch (TOF)
Time of Instruction Fetch (TIF)
Time of Execution 1 (TEX 1)
Time of Exectuion 2 (TEX 2)
Time Variable (TV)
Interrupt Enable (IE)
Interrupt (INT)
Overflow (OV)
Carry (CY)
Instruction Error (INSER)

The Time of Operand Fetch (TOF) flip-flop (A1, zone A2) controls the fetching of data from memory to the buffer register. TOF is on for one or more full memory cycles, immediately following TIF, whenever operation codes (command register bits 18-23) octal 10-37 or octal 07 are decoded. The octal 07 instructions are the shift instructions and the double two's complement. TOF is also set by a program interrupt. Necessary conditions for setting TOF are that no indexing or indirect address operation is in progress, and that TIF is on. Double add, double subtract, and double store operations hold TOF on more than one cycle, while command register lock up holds TOF indefinitely. Otherwise, TOF is cleared by TEX 1 coming on.

The Time of Instruction Fetch (TIF) flip-flop (A2, zone B2) controls the fetching of data from memory to the command register. TIF is on for one memory cycle at the end of each instruction execution sequence -- that is, TIF is set at the completion of a current instruction, to fetch the next instruction. The gates which are ORed at A4 identify the various completion conditions. Note that TIF and TOF can both be off at the same time, but they can not both be on at the same time.

At the top left of the page is the logic for setting TIF (or TOF) when loading the command register manually from the switch register (using the signal GWCR) or during system reset (using RST 1). In both these cases, TIF + TOF = 1; that is, one or the other (TIF or TOF) must be on for any given memory cycle.

The Time of Execution 1 (TEX 1) flip-flop (E5-6, zone C1) controls part of the sequence for multiplication and division, and also some single cycle operations for instructions in the group beginning with octal 10-37. This flip-flop is set whenever such an instruction is present, provided TOF is on and there is no indexing or indirect addressing in progress. TEX 1 is cleared whenever the operation is not multiplication or division or (during multiplication or division) when the iteration counter equals zero. An overflow during division clears TEX 1, terminating division.

The Time of Execution 2 (TEX 2) flip-flop (E5-10, zone C1) controls the operations add 1 to memory, subtract 1 from memory, multiply (if the iteration counter equals zero), and divide (if no overflow occurs). TEX 2 sets when one of these four conditions appears, and clears at the end of that operation.

To summarize the control sequences for divide and multiply, the division algorithm uses one cycle of TOF, one TEX 1, 23 cycles of both TEX 1 and TEX 2, one TEX2, and one TIF, while multiplication entails one cycle of TOF, 23 of TEX 1, and one cycle of both TEX 2 and TIF.

The Time Variable (TV) flip-flop (E6-6, zone B1) controls the execution of shift instructions (a subset of the 07 instructions with appropriate values for bits 13, 12, 11, 10). TV is set by the decoded shift instruction, provided TOF is on and there is no indexing or indirect addressing. TV is cleared at the end of the shift sequence, when the iteration counter equals one.

The Interrupt Enable (IE) flip-flop (E6-10, zone B1) enables the setting of the Interrupt flip-flop. IE is set by a Set State instruction (bit 8 = 1), when no indexing or indirect addressing is being performed. IE is reset by the corresponding Reset State instruction (bit 8 = 0), or by any interrupt.

The Interrupt (INT) flip-flop (D8) is set whenever TIF is on, and the contents of the interrupt register are not zero. The interrupt register can be set during any memory cycle (at T3 time) when an interrupt request is sent by a peripheral. The Interrupt flip-flop, however, can be set only when Interrupt Enable (IE) is on. INT is reset by TIF, that is, when the following instruction is fetched.

The Overflow (OV) flip-flop (D7) is set during TEX 1 for (add, subtract, double add, double subtract, divide, add 1 to memory, and subtract 1 from memory) operations in which the result exceeds the limits of the A register. OV can also be set by a Set State instruction, if bit 9 is true. If OV causes the instruction sequence to enter a subroutine, then upon exit from this subroutine, OV is restored to the same state as upon entry. OV is cleared by a Reset State instruction (bit 9=0), or by being tested with a Branch-On-State instruction.

The Carry (CY) flip-flop (E7) is set when a carry appears out of the most significant bit, during divide operations, during double add or double subtract if TEX 1 is on, and during double two's complement if TOF is on. CY is cleared if (under the same conditions) that carry output bit is zero.

The Instruction Error (INSER) flip-flop (B1, zone A2) is set, and the Invalid Command (INVER) signal is generated, whenever an invalid command appears, from memory (M) or from the switch register (W). The logic (zone A3) to the left of B1 decodes bits 18-23, such that octal numbers 41-77 in these six bits are recognized as invalid commands.

2.5 P Counter (97166005–04, 1 page)

The main circuitry on this board is the 14-bit Program Counter. This consists of 14 flip-flops and 14 dual 4-input (9309) multiplexers at the flip flop inputs.

The program counter can be loaded from the switch register (W), or from the operand field of the command register (O). In addition, the address in the program counter can be incremented by one in response to the control signal IPC. Selection among the four sets of multiplexer inputs is provided by the two bits S1 and S0, which are coded from the control signals GWP, GOP, and IPC by the logic at the bottom left of the page (zone A3), as follows:

S1	S0	Mnemonic	Signal Name
0	0		Ground
0	1	GWP	Gate Switch Register to Program Counter
1	0	IPC	Increment Program Counter
1	1	GOP	Gate Operand Field to Program Counter

GWP comes from the CPU Clock board, and is used for external loading of the program counter from the control panel. GOP comes from the Gate Control #2 board, and is used in executing branch instructions. IPC comes from the Gate Control #1 board, and is used to advance the program counter one count, for purposes of accessing consecutive address locations in memory.

When IPC is true, both inputs to bit 0 are high, and the inputs to bits 1-13 are those generated by the carry logic shown at the lower right corner of the page.

Clock pulses and reset signals are provided by the logic shown at the left center of the page (zone B3). The clock pulse (CCP1/) occurs when all three inputs to A4-6 (RCPS, MLDCRA/, and A4-8) are true. RCPS is the system clock pulse from the CPU Clock board. MLDCRA/ is true if the CPU is not executing a manual load operation of the command register or the A register. A4-8 is true if one of its three input signals -- GWP/, GOP/, or IPC/ -- is low.

The RESET pulse clears all bits of the program counter, except bit 6, which it sets to one. This leaves the program counter set to address (octal) 100. Memory addresses 0 thru 77 are not normally accessed by the program counter.

The remaining set and clear functions (SD0-5, CD6, SD7-13) are tied high (not used).

The program counter outputs P0-P13 go to the Data Bus boards (memory interface), and P0I-P13I go to indicators on the control panel.

2.6 Instruction Decode (97166008—04, 1 page)

The purpose of this board is to decode the six bits (18,19,20,21,22,23) of the command register which comprise the instruction field. These six bits are organized into two groups of three bits each. Bits 18,19,20 are the least significant, and 21,22,23 are the most significant. Each group of three bits can be represented by a single octal digit.

The (9301) decoders C2, E2, D2, B2 and A2 are actually BCD decoders of which only eight outputs are used. This is done in such a way that the three bits on pins 15, 14, and 1 are decoded into eight outputs, with the entire process (at each 9301) enabled by a low input at pin 2. The inputs to pin 2 of the respective 9301's are in turn decoded from bits 21,22,23, by the logic shown at the extreme left.

In terms of the octal digit representing the three most significant bits, the value 4 enables the decoder C2, 0 enables E2, and 1, 2, and 3 enable D2, B2, and A2, respectively. In addition, the signal ML/ must be high (no manual load operation) for any decoding to occur.

The remainder of the circuitry on this board forms a variety of logical combinations of the decoded instructions at the 9301 outputs. Most of these are gating signals for registers within the computer. Some noteworthy cases are discussed below.

SIDX (Suppress Index), formed at D3-8 (zone C2), is an OR function of all the instructions for which indexing does not occur. In such cases, the three index bits (015, 16, 17), are used for purposes other than address modification.

INA1 (formed at F3-8) is high for all instructions in which the indirect address is not used. IDX/ (E7-6) is generated by the logic shown at the extreme lower left of the page, from ML/, SIDX/, and the three index bits (X field). Also INA/.IDX/ is generated there. Indexing has priority over indirection; hence IDX/ is one of the inputs affecting INA1/.

At the top of the page (zone C1) IC1/ is generated (A6-6). This is the Inhibit Carry signal. This signal is used when the adder performs the Exclusive-OR function, and also when the adder is used simply for transfer of data from the buffer register to the A register.

AUG/ (E2-5, zone C3) is the Augmented Instruction signal. This signal runs to other boards, where additional instructions (not contained in bits 18-22 of the command register) are decoded.

2.7 Gate Control #1 (97166006—04, 1 page)

This board contains the logic for forming a variety of gating signals. The inputs are the decoded instructions from the Instruction Decode board, and the control signals from the Control Flip-flop board.

At the lower right are formed the three signals X15A/, X16A/, and X17A/, which go to the Two-bit Slice board to control the index registers. These signals duplicate IX0, IX1, and IX2 (respectively) when the CPU is in a HALT condition, and O15, O16, and O17 when the CPU is in RUN.

At the upper right are the outputs for gating the command register inputs. GCP09/gates bits 0-9, GCP1013/ gates bits 10-13, GCPI/ gates bit 14, GCPXA/ gates bits 15-17, and GCPC/ gates bits 18-23. GMCR/ gates the memory to the command register, and GRO/ gates the contents of the interrupt (R) register to the lowest six bits of the command register (used as address for memory locations (octal) 77 and below).

Other signals generated on this board have the following functions:

RX15	Clears X15 to permit comparison of contents of odd and even numbered X registers
G23CO	Sets iteration counter to decimal 23 (octal 27)--used in multiplication and division
GXFUB/	Gates index register complement to B input of adder
GAE	Gates A register to E register
GCRN/	Gates command register to N bus
PS	Peripheral Select
GAN	Gates A register to N bus
IPC	Increments program counter
STWB/	Store switch register (W) to buffer register
WX/	Write index
LDATX/	Loads OR ADD to index register
GXA	Gates index register to A register
GCPA/	Gates clock pulse to A register
GWA	Gates switch register to A register
IP	Signals peripherals to send interrupt priority signal (on N bus)
IAT	Loads interrupt address into R register
GCPB/	Gates clock pulse to B register
GCPE/	Gates clock pulse to E register
GXUB/	Gates index register to B input of adder
STX	Store X

2.8 Gate Control #2 (97166007—04, 1 page)

This board serves the same general function as Gate Control #1. The two (9301) decoders at the upper left decode the augmented instructions, according to command register bits O10-13, when bits 18-23 are octal 07. Gating signals appearing as outputs from this board have the following functions:

CI	Force carry into arithmetic unit (two's complement)
CYE/	Carry to E0 (divide)
CY-E0	Not Used
DCO	Decrement iteration counter
DSA/	Double shift around
DSN/	Double shift normalized
DTC/	Double two's complement
EXC/	Exchange A register with E register
GAFUA	Gate A register complement to A input of arithmetic unit
GBM	Gate buffer register to memory
GEUA	Gate E register to A input of arithmetic unit
GOCO	Gate operand field (lowest six bits) to iteration counter
GOM/	Gate operand field to memory interface
GOP	Gate operand field to program counter
GPM/	Gate program counter to memory
GSA	Gate sum output from arithmetic unit to A register
GOP1M/	Adds one to operand address (double precision)
LXA/	Load index register from A register
RSR	Reset switch register
RST	Reset state
SA	Shift around
SN	Shift normalized
SST	Set state
SAL	Shift A register left
SAR	Shift A register right
SEL	Shift E register left
SER	Shift E register right
WRM	Write to memory

2.9 Miscellaneous Registers (97166009—04, 1 page)

The principal circuitry on this board is as follows:

State Flip-flops
Indicator Flip-flops
Interrupt (R) Register
K-field Decoders
Branch Enable Logic

The state flip-flops (B1,B2,C1,C2,D1,D2,E1,E2) are a utility storage for the use of the programmer. They are set by the signal SST (Set State), reset by RST (Reset State), and tested by BOS (Branch On State). Which of the eight flip-flops are set or reset is determined by the eight inputs 00-07. A one at any of these inputs sets the corresponding flip-flop during SST, or resets it during RST.

To the right of the state flip-flops are the indicator flip-flops (B7 and D7). These store the four conditions GT (Greater Than), LT (Less Than), EQ (Equal), and BE (Bit Equal), which result from an Add to X (ATX) operation, or a Compare A with Memory operation. In addition, they are set from the four most significant bits of memory, when a BRU (BRanch Unconditional) Indirect command is executed. The logic for setting GT and LT lies between the indicator flip-flops and the state flip-flops on the schematic. The logic for EQ and BE, as well as the timing logic for all four indicators, is located on the Gate Control 2 board.

The indicator flip-flops are also used to store status information from the peripherals. In that case, the information is loaded into the flip-flops from the N (accumulator) bus, in response to an SPU command. The four indicators then take on the following meanings:

GT	peripheral idle
EQ	idle with error
LT	busy
BE	not available

At the extreme right (flip-flops C8 D8, and E8) is the six-bit interrupt (R) register. This register stores the address of the peripheral whose interrupt is being serviced. The interrupt register is loaded in the following manner. The signal IP (from the Gate Control #1 board) at phase time T3, calls for the selection of the peripheral with highest priority among those requesting service. Each of these peripherals then places a one on the (N bus) line of every peripheral having a lower priority than it. At time T5 the peripheral which has not been thus subdued (by a one placed on its line) transmits its address to the CPU via the six least significant bits of the N bus. The signal IAT (also from Gate Control #1) gates this address into the R register. Any non-zero address in the R register causes the output R≠0 of B8-8 to be true. The interrupt is then serviced after the completion of the current instruction.

The lower half of the page contains the logic for generating the Enable Branch signal. There are three instructions which can give rise to a true value for Enable Branch. These Are:

Branch On Indicator (BOI)
Branch on Accumulator Test (BAT)
Branch On State (BOS)

In all three cases, four bits of the command register are used to specify the test on which a possible branch is based. These four bits are called the K-field, and are bits 014, 15, 16, 17 (shown as inputs to this board at the extreme left of the page).

In the case of the indicator test, an affirmative result appears as a low on one of the points (zone A2) C5-8, A5-12, A5-6, or C5-6. This in turn causes E3-8 (Enable Branch) to be high.

For an accumulator test, an affirmative result causes a low on point (zone A2) B5-6, B5-12, B5-8, or C5-12, when the contents of the A register are positive, zero, negative, or odd, respectively. As above, this causes E3-8 (Enable Branch) to be high.

The logic in zone A3 implements the Branch-On-State instruction. In this case, each bit configuration in the K-field represents a command to test one of 16 signals. These signals are:

Eight state flip-flops (SW0-SW7)
Six console switches (CS0-CS5)
Interrupt Enable (IE)
Overflow indicator (OV)

The (9301) decoders (B4 and B3) generate the test signals for the 16 quantities listed above, from the four K-field bits. These are BCD decoders, only eight outputs of which are used, in such a way that a low on pin 2 enables the decoding of the three bits on pins 1, 14, 15. Command register bit 14 goes to pin 2 of B4, and its complement goes to pin 2 of B3.

The 16 outputs from these decoders each test one of the state conditions, at the AND-gate inputs to A1, D4, A2, and D3. The outputs from these AND gates are ORed, as shown, to produce a high at E3-8 (Enable Branch) if any state is tested and found to be true.

2.10 Memory Control (97340201—04, 1 page)

This board contains the logic for generating the A and B memory select signals and the memory access signal, as well as the start of cycle and start write cycle pulses, for memory control purposes. Also located on this board is the iteration counter, which plays an important part in multiplication, division, and shift operations.

At the bottom of the page (zone A2) are the A and B memory select flip-flops, D5-10 and D5-6, respectively, the outputs of which are the signals AMEM and BMEM, which go to the Data Bus boards to control memory data transfer. Several factors influence which of these two flip-flops is set when memory is accessed by the CPU.

Bit 13 in either the command register or the program counter controls which memory is used (bit 13 = 1 selects B memory). Whether the address in the command register or the program counter is used depends on the status of various branch instructions, which is compiled by the logic to the left (zone A3).

In the lower right portion of the page are two flip-flops, B5 and A5, which are the A and B memory write flip-flops. These are set at phase time T1, according to the write request signals from peripherals or CPU, on A or B memory bus bit 18. The outputs of these flip-flops go to the logic group at the extreme left of the page (zone B3). This logic group generates the memory control signals SOC (Start Of Cycle) at time T3, and, when appropriate, SWC (Start Write Cycle) at time T5. The write functions to either memory can be manually inhibited by the panel switches shown.

At gate E6-6 (zone C1) is generated the signal ACCS, which is true when a peripheral has access to memory (i.e., when the CPU is denied access to memory).

The remaining circuitry on this board is the iteration counter (also called the CO counter). This counter (C2 and D2) is instrumental in the control of multiplication, division, and shift operations. It is a six-bit counter, which is loaded from bits 0-5 of the command register when GOCO is true, and is also loaded with the quantity decimal 23 when G23CO is true. The counter actually holds the complement of the quantities mentioned, so that in effect it counts downward from the number loaded in. The count is reduced by one at each (CPS) clock pulse, whenever the signal DCO (decrement counter) is true.

Three special signals (COEQZ at F4-2, COEQ1 at E4-6, and COLEQ5 at B2-8) are formed (zone B1) from the appropriate combinations of iteration counter states. COEQZ is true if the counter contents equal zero, in which case multiplication and division are terminated. COEQ1 true indicates a count of one, which resets control flip-flop TV. COLEQ5 is true when the count is five or less, or if SN/ is low (A23≠A22 for double shift normalize), or if bits 0-5 of the command register are all zero. These conditions all indicate that the shift operation will complete during the current memory cycle. When COLEQ5 is true, TIF is set at the end of the shift operation.

Section 3 Instruction Repertoire

3.0 INTRODUCTION

This section discusses the instruction repertoire of the FST-1, which consists of nine instruction groups, totaling 48 instructions. Accompanying each instruction are examples coded as they would be for the FST-1 assembler.

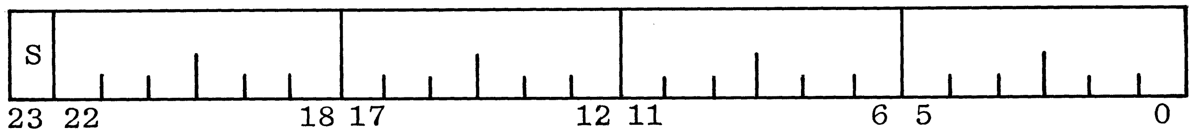
3.1 ABBREVIATIONS AND MNEMONICS

Appendicies A and B contain a list of abbreviations and instruction mnemonics which are used in the description of machine instructions. Any abbreviation which is enclosed in parentheses is a reference to the contents of that particular register or memory location. For example, (M) is a reference to the contents of memory location M; (A) refers to the contents of the accumulator. (A) → M is read, "The contents of A go to memory location M." (M) + (A) → A is read, "The contents of memory location M, plus the contents of A go to A." In the following instruction descriptions, Me is used to refer to the effective memory address, i.e., after both indexing and indirection. Numbers are used to reference individual bits or groups of bits in the registers. For example, A0 represents the "0" bit of the accumulator: A0-7 represents the least significant eight bits of accumulator, etc.

3.2 INSTRUCTION FORMATS

3.2.1 Word Format

The FST-1 CPU word is 24 bits long. The bit positions are numbered from right to left, beginning with 0. Bit 23 is the sign bit. Negative numbers are stored in two's complement form.



3.2.2 Standard Instruction Word Format

A standard instruction word generally has four parts which consist of a 6 bit operation code field, a 3 bit index field, a 1 bit field used to specify direct (0) or indirect (1) address and a 14 bit address field.

If an instruction can be indexed, the instruction format will include an X, (which designates the index to be used). If an instruction can be indirect address modified, the format will include an I. If the instruction can reference memory, then the M in the instruction format is the memory location of the operand.

Operation Code	Index	I	Operand Address
23 18	17 15	14	13 0

3.2.3 Augmented Instruction Word Format

Some instructions do not require a fourteen bit address field, as used in the standard instruction word format. These instructions have operation code 07 and use bits 10 through 13 to 'augment' the 07 operation code (See section 3.8 for further details). An example of an augmented operation code is shown below.

07	Index	I	Augment		Shift Count
23 18	17 15	14	13 10	9 6	5 0

3.3 ASSEMBLER FORMATS

Each of the instruction descriptions which follow is illustrated with an example of the appropriate FST-1 symbolic assembly code.

The FST-1 Assembler Manual should be consulted for the details of instruction formats, conventions, etc. The discussion presented here will clarify the instruction descriptions.

All instructions have an Opcode and most of them also have an Operand, although some others, (such as TCA), do not. "Opcode" is the mnemonic name for the command or Operation Code and "Operand" is the symbolic address of the operand, etc.

There are two basic operand formats: indexable and nonindexable, as shown in the following examples:

(1) Indexable instruction: Symbolic address, index

Examples:

```
STA   BUFFER,5
LDA   TABLE (no index specified)
```

(2) Non-indexable instruction:

(a) index register, operand address
(b) indicator value, operand address
(c) state flip-flop, operand address

Examples:

```
BOS   5,L1
BOS   PASSL,L232
BOI   3,LEQ
LDX   6,5
LDX   PNTRX,TABLE-2
```

Indirect addressing is noted by an asterisk (*) immediately following the opcode mnemonic, as shown below:

```
LDE*  TEMP1,3
MUL*  FACTOR
```

3.4 CYCLES REQUIRED

In each of the instruction descriptions which follow, the number of machine cycles required to execute the instruction is given exclusive of indexing and indirection. Machine cycle times for each instruction are also listed in appendix B. A memory cycle is 1.75 microseconds in duration.

3.5 ARITHMETIC INSTRUCTIONS

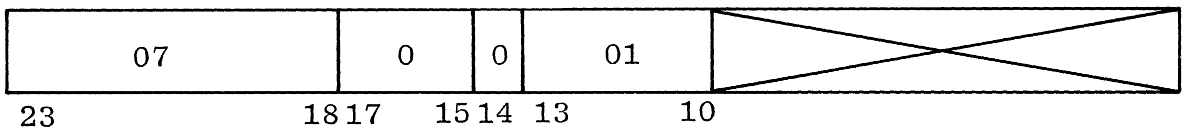
The FST-1 uses the following 10 arithmetic instructions: TCA, DTC, ADD, SUB, DADD, DSUB, MUL, DIV, AOM and SOM.

3.5.1 TCA Two's Complement A Register

Definition: Two's Complement of (A) → A

Cycles Required: 1

Instruction Format (Augmented):



Description:

The contents of the accumulator are two's complemented and placed in the accumulator.

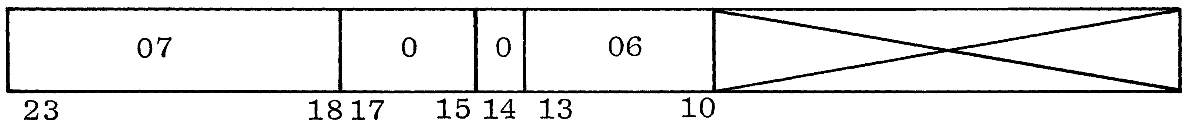
Assembler Format: TCA

3.5.2 DTC Double Two's Complement

Definition: Two's complement of (A and E) → A and E

Cycles Required: 2

Instruction Format (Augmented):



Description:

The contents of A and E are two's complemented and the result is placed in A and E.

Assembler Format: DTC

3.5.3 ADD Addition

Definition: (A) + (Me) →A

Cycles Required: 2

Instruction Format:

20	X	I	M
23 18	17 15	14	13 0

Description:

The contents of memory location, Me, are added algebraically to the contents of the accumulator, with the sum being stored in the accumulator. The contents of memory are not changed. An overflow from the accumulator will set the overflow flag OV, indicating the result is incorrect.

Assembler Format: ADD* TABLE+1,2

3.5.4 SUB Subtraction

Definition: (A) - (Me) → A

Cycles Required: 2

Instruction Format:

22	X	I	M
23 18	17 15	14	13 0

Description:

The contents of memory location, Me, are subtracted algebraically from the contents of the accumulator, with the difference being stored in the accumulator. The contents of memory are not changed. An overflow from the accumulator will set the overflow flag OV, indicating the result is incorrect.

Assembler Format: SUB* TABLE+1,7

3.5.5 DADD Double Addition

Definition: (A and E) + (Me and Me + 1) → A and E, where Me is an even numbered address.

Cycles Required: 4

Instruction Format:

30	X	I	M
23 18	17 15	14	13 0

Description:

The contents of memory locations, Me and Me + 1 are added algebraically to the contents of A and E. Bits 23 of (Me) and (A) are the operand signs. The sum is stored in A and E as a 47 bit signed number with A containing the most significant half of the sum. The sign of the sum is stored in A23. Two's complement is used for negative numbers. The contents of memory are unchanged by the operation. An overflow will set the overflow flag OV, indicating the result is incorrect.

Assembler Format: DADD* TABLE+1

3.5.6 DSUB Double Subtraction

Definition: (A and E) - (Me and Me + 1) → A and E, where Me is an even numbered address.

Cycles Required: 4

Instruction Format:

32	X	I	M
23 18	17 15	14	13 0

Description:

The contents of Me and Me + 1 are subtracted algebraically from the contents of A and E. Bits 23 of (Me) and (A) are the signs of the operands. The difference is stored in A and E as a 47 bit signed number, with A containing the most significant half. The sign of the difference is stored in A23. Two's complement notation is used for negative numbers. The contents of memory are unchanged by the operation. An overflow will set the overflow flag OV, indicating the result is incorrect.

Assembler Format: DSUB* TABLE+1

3.5.7 MUL Multiply

Definition: (Me) x (A) → A and E

Cycles Required: 25

Instruction Format:

34	X	I	M
23 18	17 15	14	13 0

Description:

The contents of memory location, Me, are multiplied by the contents of the accumulator. The product is stored in A and E, with A containing the most significant half. A and Me are assumed to be positive numbers. The contents of memory are not changed.

Assembler Format: MUL* ALPHA

3.5.8 DIV Division

Definition: (A and E)/(Me) → E
Remainder → A

Cycles Required: 25

Instruction Format:

35	X	I	M
23 18	17 15	14	13 0

Description:

The contents of A and E are divided by the contents of memory location Me. The quotient is left in E and the remainder in A. The original contents of A, E and Me are assumed to be positive. The contents of memory are not changed. A divide overflow will occur if (A) $\geq$ (Me). For this condition, the divide is terminated and the overflow flip flop is set. In the event of an overflow, A and E remain shifted left one place.

Assembler Format: DIV* ROGER,2

3.5.9 AOM Add One To Memory

Definition: (Me) + 1 $\rightarrow$ Me

Cycles Required: 4

Instruction Format:

36	X	I	M
23 18	17 15	14	13 0

Description:

The contents of memory location, Me, are incremented by one (1). An overflow condition will cause the OV flag to be set. In the event of an overflow, the result of the operation is incorrect.

Assembler Format: AOM* DETA

3.5.10 SOM Subtract One From Memory

Definition: (Me) - 1 $\rightarrow$ Me

Cycles Required: 4

Instruction Format:

37	X	I	M
23 18	17 15	14	13 0

Description:

The contents of memory location, Me, are decremented by one (1). An overflow condition will cause the OV flag to be set. In the event of an overflow, the result of the operation is incorrect.

Assembler Format: SOM TEM1

3.6 DATA TRANSFER INSTRUCTIONS

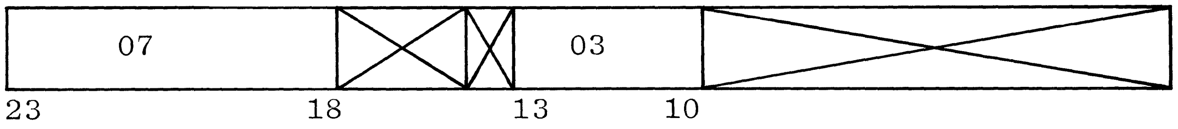
The FST-1 uses the following 8 data transfer instructions: RSR, EXC, STA, STE, LDA, LDE, DLD and DST.

3.6.1 RSR Read Switch Register

Definition: (W) → A

Cycles Required: 1

Instruction Format (Augmented)



Description:

The contents of the console switch register, W, are loaded in the A register.

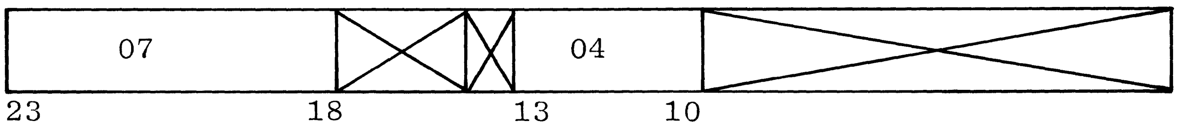
Assembler Format: RSR

3.6.2 EXC Exchange A And E

Definition: (A) → E, (E) → A

Cycles Required: 1

Instruction Format (Augmented):



Description:

The contents of the A register and the contents of the E register are exchanged.

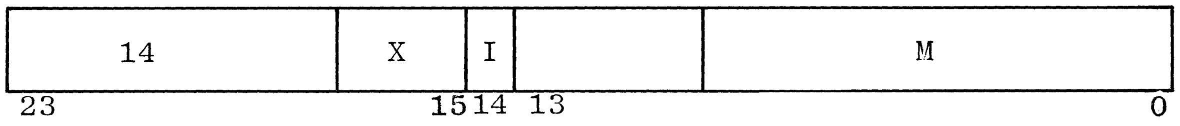
Assembler Format: EXC

3.6.3 STA Store A

Definition: (A) → Me

Cycles Required: 2

Instruction Format:



Description:

The contents of the A register are stored in memory location, Me. The contents of A are not changed.

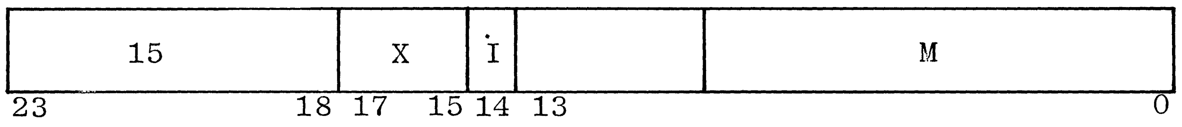
Assembler Format: STA* ALP1,2

3.6.4 STE Store E

Definition: (E) → Me

Cycles Required: 2

Instruction Format:



Description:

The contents of the E register are stored in memory location Me. The contents of E are not changed.

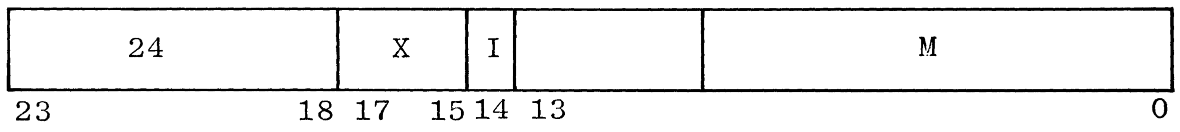
Assembler Format: STE* TEMP5

3.6.5 LDA Load A

Definition: (Me) → A

Cycles Required: 2

Instruction Format:



Description:

The contents of memory location, Me, are copied into the accumulator A. The contents of memory are not changed.

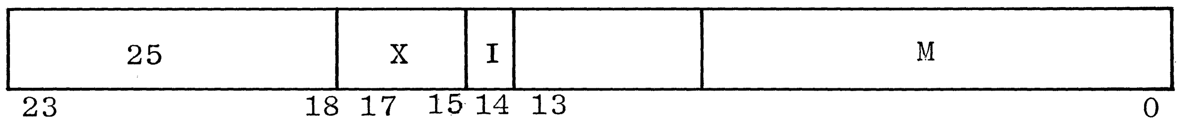
Assembler Format: LDA* TABLE,XPNTR

3.6.6 LDE Load E (The Accumulator Extension)

Definition: (Me) → E

Cycles Required: 2

Instruction Format:



Description:

The contents of memory location, Me, are copied into the accumulator extension E. The contents of memory are not changed.

Assembler Format: LDE* TABLE+1

3.6.7 DLD Double E Load A And E

Definition: (Me and Me + 1) → A and E, where Me is an even numbered address.

Cycles Required: 3

Instruction Format:

31	X	I		M
23 18	17 15	14	13	0

Description:

The contents of memory location, Me and Me + 1, are loaded into A and E respectively. The contents of memory are not changed.

Assembler Format: DLD* TABLE+5,2

3.6.8 DST Double Store

Definition: (A and E) → Me and Me +1, where Me is an even numbered address.

Cycles Required: 3

Instruction Format:

33	X	I	M
23 18	17 15	14	13 0

Description:

The contents of the A and E registers are stored in memory locations, Me and Me + 1 respectively. The contents of A and E are not changed.

Assembler Format: DST TEMP2+2

3.7 INDEX INSTRUCTIONS

The FST-1 uses the following 5 index instructions: LDX, LXA, ATX, STX and LAX.

NOTE

It is standard practice to use the index registers in the FST-1 in adjacent pairs, viz: X7 with X6, X5 with X4, X3 with X2 and X1 with X0. When so used, the odd index is the active, working index while the even index is the limit index for comparison purposes.

3.7.1 LDX Load Index

Definition: Me → Xn

Cycles Required: 1

Instruction Format:

05	X	I	M
23 18	17 15	14	13 0

Description:

The effective address, Me is loaded into the addressed index register. Index address modification does not occur, but a special form of indirect addressing does take place: bits 14-0 are replaced in the command register from memory addressM, bits 17-15 being obtained from the current instruction word and not from memory address M. (See also ATX and STX).

Assembler Format: LDX* 7,200

3.7.2 LXA Load Index From A

Definition: (Ao-13) → Xn

Cycles Required: 1

Instruction Format (Augmented):

07	X	0	00	☒
23 18	17 15	14	13 10	

Description:

The addressed index register, Xn, is loaded from the contents of the accumulator bits, A9 through A13. Index address modification does not occur.

Assembler Format: LXA INDX3

3.7.3 ATX Add To Index

Definition: Me + (X) → X

Cycles Required: 2

Instruction Format:

11	X	I	M
23 – 18	17 – 15	14	13 – 0

Description:

The contents of the addressed index register are added to the effective address (after indirect address modification) and the sum is placed back in the addressed index register. Then the GT, EQ, and LT indicators are set by comparing Xn (the addressed index register) to Xn-1 ('n' must be odd). Index address modification does not occur.

Note that only bits 14-0 are replaced in the command register from memory address M under indirection. (See also LDX and STX).

Assembler Format: ATX* 5,TABLE1

3.7.4 STX Store

Definition: (Xn) → Me

Cycles Required: 2

Instruction Format:

16	X	I	M
23 – 18	17 – 15	14	13 – 0

Description:

The contents of the addressed index register X are stored in memory location Me. The contents of X are unchanged. Bits 14-23 of Me are zeroed. Index address modification does not occur, but a special form of indirect addressing does take place: bits 14-0 are replaced in the command register from memory address M, bits 17-15 being obtained from the current instruction word and not from memory address M. (See also ATX and LDX).

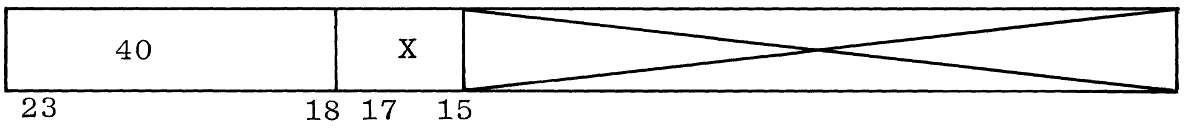
Assembler Format: STX5,TEMP1

3.7.5 LAX Load A From Index

Definition: (Xn) $\rightarrow$ A

Cycles Required: 1

Instruction Format:



Description:

The contents of the specified index register are transferred to A. Bits 23-14 of A are zeroed.

Assembler Format: LAX 2

3.8 SHIFT INSTRUCTIONS

The FST-1 uses the following 9 augmented instructions: DSN, SR, LS, SA, SL, DSR, LDS, DSA and DSL. The execution time depends upon the number of bit positions to be shifted, as shown in the following:

2 cycles for $Je \leq 9$
3 cycles for $9 < Je \leq 14$
4 cycles for $14 < Je \leq 19$
5 cycles for $19 < Je \leq 24$
6 cycles for $24 < Je \leq 29$
7 cycles for $29 < Je \leq 34$
8 cycles for $34 < Je \leq 39$
9 cycles for $44 < Je \leq 49$

Expressed as a formula:

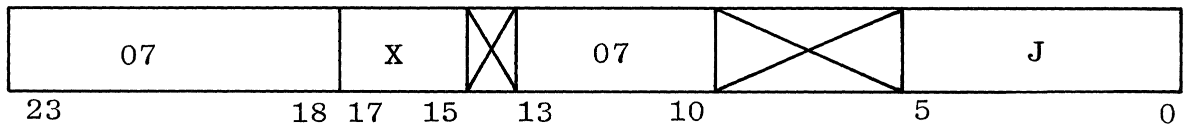
$$T = 2 + [(Je-9)/5]_{\text{integer}} \text{ cycles.}$$

3.8.1 DSN Double Shift Normalize

Definition: Normalize A and E

Cycles Required: $2 + [(J-9)/5]_{\text{integer}}$

Instruction Format:



Description:

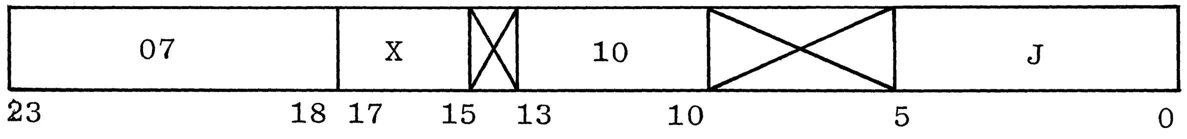
The contents of A and E are shifted left Je bit positions, or until the information in bit position A23 differs from that in A22. E23 shifts into A0 and zeros are entered into E0. At the termination of the shifting, the contents of the shift counter are stored in Index register zero. DSN may use indexing; the contents of X are added to J to obtain the modified shift count, Je.

3.8.2 SR Shift Right

Definition: Shift (A) Right Arithmetical

Cycles Required: $2 + [(J-9)/5]_{\text{integer}}$

Instruction Format:



Description:

The contents of the A register are shifted right Je bit positions. The sign bit, bit 23, of the A register is copied into bit position 22 as the register is shifted. Bits shifted from A0 are lost. SR may use indexing; the contents of Xn are added to J to obtain the modified shift count, Je.

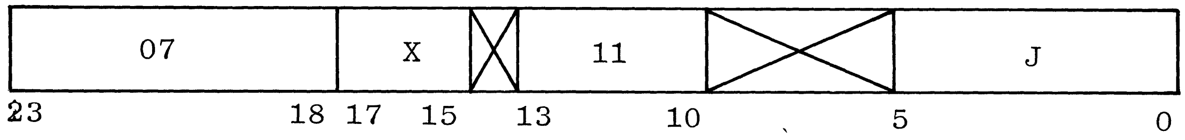
Assembler Format: SR 5

3.8.3 LS Logical Shift

Definition: Shift (A) Right Logical

Cycles Required: $2 + [(J-9)/5]_{\text{integer}}$

Instruction Format:



Description:

The contents of the A register are shifted right Je bit positions, zeros being entered into A from the left (A23). LS may use indexing; the contents of X are added to J to obtain the modified shift count, Je.

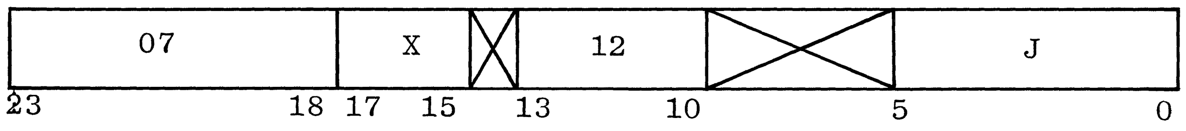
Assembler Format: LS 5

3.8.4 SA Shift Around

Definition: Shift (A) Left Around

Cycles Required: $2 + [(K-9)/5]_{\text{integer}}$

Instruction Format:



Description:

The contents of the A register are shifted left around Je bit positions, with A23 shifting into A0. SA may use indexing; the contents of the X are added to J to obtain the modified shift count, Je.

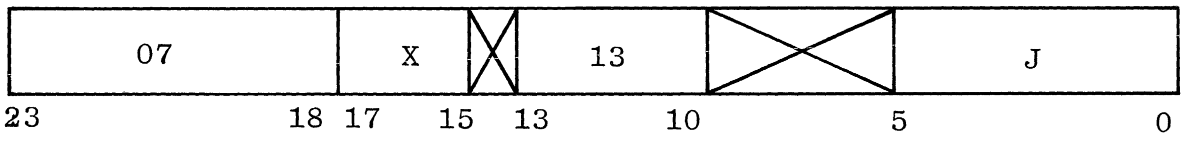
Assembler Format: SA 4,5

3.8.5 SL Shift Left

Definition: Shift (A) Left End Off

Cycles Required: 2 + [(J-9)/5]
integer

Instruction Format:



Description:

The contents of the A register are shifted left Je bit positions, with zeros being entered into A0. SL may use indexing; the contents of X are added to J to obtain the modified shift count, Je.

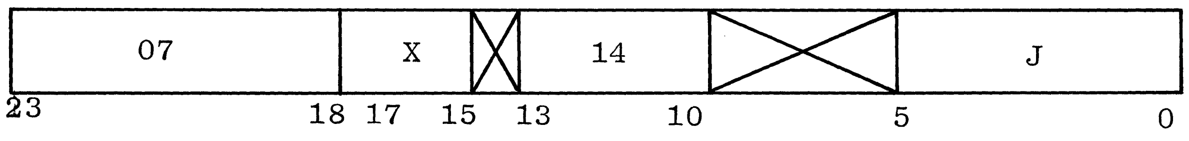
Assembler Format: SL 0,3

3.8.6 DSR Double Shift Right

Definition: Shift A and E Right Arithmetical

Cycles Required: 2 + [(J-9)/5]
integer

Instruction Format:



Description:

The contents of the A and E registers are shifted right (A0 shifting into E23) Je bit positions. The sign of A (A23) does not change during this shift operation and is repeatedly copied into A22 during the shift. DSR may use indexing; the contents of X are added to J to obtain the modified shift count, Je.

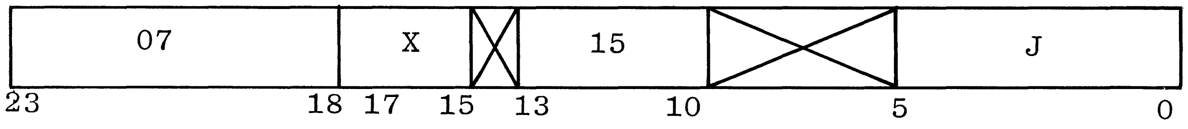
Assembler Format: DSR 25

3.8.7 LDS Logical Double Shift

Definition: Logical Shift A and E Right

Cycles Required: $2 + [(J-9)/5]_{\text{integer}}$

Instruction Format:



Description:

The contents of A and E are shifted right Je bit positions. Zeros are entered into A23. In addition, A0 is shifted into E23, while bits shifted out of E0 are lost. LDS may use indexing; the contents of Xn are added to J to obtain the modified shift count, Je.

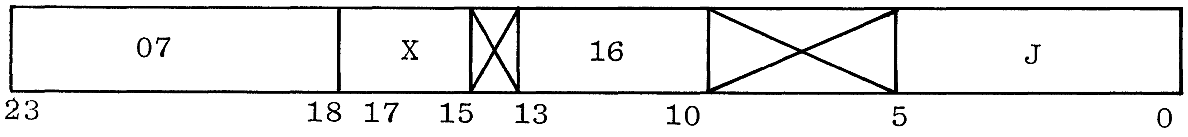
Assembler Format: LDS 0,5

3.8.8 DSA Double Shift Around

Definition: Shift A and E Around Left

Cycles Required: $2 + [(J-9)/5]_{\text{integer}}$

Instruction Format:



Description:

The contents of A and E are shifted left around Je positions, (A23 going to E0 and E23 going to A0). DSA may use indexing; the contents of X are added to J to obtain the modified shift count, Je.

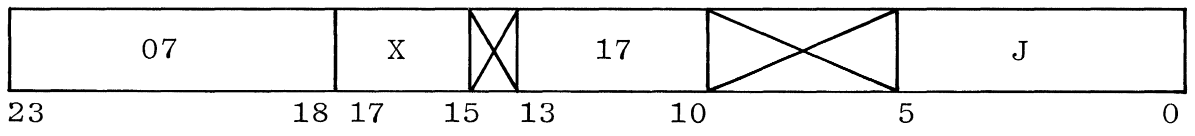
Assembler Format: DSA 24

3.8.9 DSL Double Shift Left

Definition: Shift A and E Left

Cycles Required: $2 + [(J-9)/5]_{\text{integer}}$

Instruction Format:



Description:

The contents of A and E are shifted left Je bit positions; E23 is shifted into A0. Zeros are entered into E0 and the bits shifted out of A23 are lost. DSL may use indexing; the contents of X are added to J to obtain the modified shift count, Je.

Assembler Format: DSL 20

3.9 LOGICAL INSTRUCTIONS

The FST-1 uses the following 4 logical instructions; RUM, EOR, AND and OR.

3.9.1 RUM Replace Under Mask

Definition: $(Me) \wedge (E) \vee (A) \wedge (\overline{E}) \rightarrow A$
on a bit by bit basis.

Cycles Required: 2

Instruction Format:

17	X	I	M
23 18	17 15	14	13 0

Description:

The contents of Me are masked into A under the control of E. For each "1" bit in E, the corresponding bit in A is replaced by the corresponding bit in Me. Neither (Me) or (E) change.

Assembler Format: RUM* 2020

3.9.2 EOR Exclusive OR

Definition: $(A) \vee (Me) \rightarrow A$

Cycles Required: 2

Opcode Format:

21	X	I	M
23 18	17 15	14	13 0

Description:

The contents of Me are "Exclusively ORed," with the contents of A on a bit by bit basis, and the results stored in A.

Assembler Format: EOR TEMP1+1

3.9.3 AND Logical And

Definition: $(A) \wedge (Me) \rightarrow A$

Cycles Required: 2

Instruction Format:

26	X	I	M
23 18	17 15	14	13 0

Description:

The contents of Me and the A register are "ANDed" on a bit by bit basis and the results stored in A.

Assembler format: AND TEMP1

3.9.4 OR Logical Or (Inclusive OR)

Definition: (A)   (M) → A

Cycles Required: 2

Instruction Format:

27	X	I	M
23 18	17 15	14	13 0

Description:

The contents of Me and the A register are "ORed" on a bit by bit basis and the results stored in A.

Assembler Format: OR* 1000

3.10 STATE CONTROL INSTRUCTIONS

This section consists of 2 instructions: "SET STATE" and "RESET STATE", both of which are augmented instructions. The state flip flops affected by these instructions are defined by Ce, the least significant 10 bits of the instruction, modified by the contents of X. The ten state flip flops which are affected by these instructions are: SW0, SW1, SW2, SW3, SW4, SW5, SW6, and SW7, the interrupt enable flip flop IE, and the overflow indicator OV. The individual controls for these indicators are the set state and reset state instruction bits, 00 through 09, respectively. If the effective address of the set state (or reset state) instruction has a logical one in the least significant bit, bit 00, SW0 will be set (or reset) by the instruction. If 00 is a logical zero, SW0 will not be changed. SW1 will be set (reset) if a logical one exists in bit 01 of the effective address of the instruction. Any number of the state flip flops can be set (or reset) with one instruction execution, as shown on the following page.

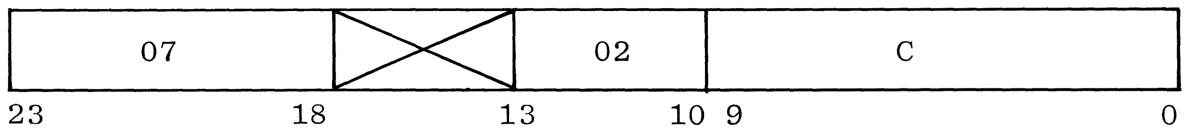
Operand Address Bit	State Flip Flop Affected
0_0	SW0
0_7	SW1
0_2	SW2
0_3	SW3
0_4	SW4
0_5	SW5
0_6	SW6
0_7	SW7
0_8	IE
0_9	OV

3.10.1 SST Set State

Definition: Set States Defined by C

Cycles Required: 1

Instruction Format:



Description:

Execution of the SET STATE instruction will cause any of ten state flip flops to be set.

Assembler Format: SST, SW0 SW1, SW4, OV

NOTE

> A special assembler mnemonic exists for setting bit 08 (IE). This is IEN for Interrupt Enable.

Assembler Format: IEN

3.10.2 RST Reset State

Definition: Reset States Defined by C

Cycles Required: 1

Instruction Format:

07	╳	05	C

Description:

The execution of the Reset State instruction will cause the state flip flops addressed to be reset.

Assembler Format: RST, SW2, SW3, IE, OV

NOTE

> A special assembler mnemonic exists for resetting bit 08 (IE). This is IDA for Interrupt Disable.

Assembler Format: IDA

3.11 COMPARE INSTRUCTION

3.11.1 CAM Compare A With Memory

Definition: The contents of A, (A), are compared with the contents of Me, (Me). The indicators GT, EQ, LT and BE are set accordingly.

Cycles Required: 2

Instruction Format:

23	X	I	M
23 18	17 15	14	13 0

Description:

The contents of A are compared with the contents of memory location Me. The greater than (GT), equal (EQ), less than (LT), or bit equal (BE) indicators are set in accordance with the outcome of the comparison as described below:

1. If (A) > (Me) the GT indicator is set.
2. If (A) = (Me) the EQ indicator is set.
3. If (A) < (Me) the LT indicator is set.
4. BE is set if a logical one exists in any corresponding bit positions of both A and Me. For example, if the fifth bit of A is a one, and the fifth bit position of Me is also a one, BE will be set when the comparison is complete.

The contents of Me are not changed.

Assembler Format: CAM TEMP5

3.12 TRANSFER OF CONTROL INSTRUCTIONS

The FST-1 uses 7 instructions which effect transfer of control (or branching). They are: BAH, BRU, BAT, BOI, BOS, BSM and BSZ.

3.12.1 BAH Branch And Halt

Definition: Branch to Me and Halt

Cycles Required: 1

Instruction Format:

00	X	I	M
23 18	17 15	14	13 0

Description:

Program Control is transferred to Me, after which program execution is halted. The next instruction, which will be executed if the start switch is actuated, is displayed in the command register indicators.

Assembler Format: BAH START2

3.12.2 BRU Branch Unconditionally

Definition: Branch Unconditionally to Me

Cycles Required: 1

Instruction Format:

01	X	I	M
23 18	17 15	14	13 0

Description:

The BRU instruction will transfer program control unconditionally to Me. BRU can be indexed and indirect address modified.

An Indirect Address modification of BRU will set the five indicators OV, GT, LT, EQ, and BE from bit positions 23, 22, 21, 20 and 19 of the memory location containing the effective address word. For example, if bit position 23 of the memory location containing the Me for BRU contains a one-bit, OV will be set during execution of the BRU instruction. Bit 22, containing a one, will cause GT to be set, etc.

An indirect BRU is generally used as a return branch for either a BSM ("branch, store return at location M") or a BSZ ("branch store return at location zero") instruction. Note that this restores the five indicators to the states which existed when either a BSM or BSZ instruction was executed.

Assembler Format: BRU* STATE+5

3.12.3 BAT Branch A Register Test

Definition: Branch to M on A Register Test

Cycles Required: 1

Instruction Format:

02	K	M
23 18	17 14	13 0

Description:

The BAT instruction will transfer program control to M, dependent upon the contents of the accumulator. The accumulator contents are tested for positive, zero, negative or odd states.

BAT can neither be indexed nor address modified. The K-field (bits 17-14) specifies the state of A to be tested. If bit 17 is a one, program control will be transferred to M, providing the contents of A are positive (A23-0). If bit 16 is a one, program control will be transferred to M, providing the contents of A are zero, etc. Combinations of states are allowed. For example, if both bits 17 and 14 are ones, program control will be transferred to M if A is positive or if A is odd (A0=1).

NOTE

Zero is an exclusive state and is neither positive nor negative.

Assembler Format: BAT K,TEST2

NOTE

Seven special assembler mnemonics exist to aid the programmer. These are: BP, BPZ, BZ, BNZ, BN, BNEZ and BO for K = 10, 14, 4, 6, 2, 12 and 1, respectively, (i.e., Branch Positive, Positive or Zero, Zero Negative or Zero, Negative, Not Equal to Zero and Odd, respectively).

Assembler Format: BPZ TEST2

3.12.4 BOI Branch On Indicator

Definition: Branch to M if tested Indicator(s) set

Cycles Required: 1

Instruction Format:

03	K	M
23 18	17 14	13 0

Description:

The BOI instruction will transfer program control to M, dependent upon the state of the four indicators GT, EQ, LT, or BE. BOI can neither be indexed nor indirect address modified. The value, K, is defined by bits 17-14 of the BOI instruction word. Bit 17 tests the state of the GT indicator, while bits 16, 15 and 14, respectively, test the states of the EQ, LT and BE indicators. If one or more of the tests is true, program control will be transferred to M. For example, if bits 17 or 16 are set in the BOI instruction word, then program control will be transferred to M, if either GT or EQ is set.

Assembler Format: BOI K,TEST2

NOTE

Seven special assembler mnemonics exist to aid the programmer. These are: BG, BGE, BE, BLE, BL, BNE and BBC for K = 10, 14, 4, 6, 2, 12 and 1, respectively, (i.e., Branch Greater, Greater or Equal, Equal, Less Than or Equal, Less Than, Not Equal and Bit Compare, respectively).

Assembler Format: BGE TEST2

3.12.5 BOS Branch On State

Definition: Branch to M is State K Set

Cycles Required: 1

Instruction Format:

04	K	M
23 18	17 14	13 0

Description:

Program control is transferred to M, providing the switch or indicator defined by K is set. BOS can neither be indexed nor indirect address modified. Bits 17, 16, 15 and 14 of the BOS instruction word are decoded into sixteen values of K. The table below defines the appropriate switch or indicator tested for each value of K (expressed octally):

K_8	State Tested	
0	Switch Flip Flop	0
1	Switch Flip Flop	1
2	Switch Flip Flop	2
3	Switch Flip Flop	3
4	Switch Flip Flop	4
5	Switch Flip Flop	5
6	Switch Flip Flop	6
7	Switch Flip Flop	7
10	Interrupt Enable	IE
11	Overflow Indicator	OV (note: after testing, OV is reset.)
12	Console Switch	CS0
13	Console Switch	CS1
14	Console Switch	CS2
15	Console Switch	CS3
16	Console Switch	CS4
17	Console Switch	CS5

Assembler Format: BOS K,ALPHA

3.12.6 BSM Branch Store Return At M

Definition: Branch to Me + 1, Store Return at Me

Cycles Required: 2

Instruction Format:

12	X	I	M
23 18	17 15	14	13 0

Description:

Program control is unconditionally transferred to Me + 1. The contents of the Program Counter, (current program address + 1) are stored in Me, bits 0 - 13. The states of the five indicators OV, BT, EQ, LT, and BE, are stored in memory location Me in bit positions 23, 22, 21, 20, and 19, respectively. These states are restored to the indicators when an indirect BRU instruction is used as a subroutine exit (see BRU description).

Assembler Format: BSM PRTCH

3.12.7 BSZ Branch Store Return At Zero

Definition: Branch to Me, Store Return at Memory Location Zero

Cycles Required: 2

Instruction Format:

13	X	I	M
23 18	17 15	14	13 0

Description:

Program control is unconditionally transferred to Me. The contents of the Program Counter, (current program address + 1) are stored in memory location zero, bits 0 - 13.

The states of the five indicators OV, GT, EQ, LT, and BE, are stored in memory location zero, in bit positions 23, 22, 21, 20 and 19, respectively. These states are restored to the indicators when an indirect BRU instruction is used as a subroutine exit (see BRU description).

Assembler Format: BSZ* TEST3

3.13 INPUT/OUTPUT INSTRUCTIONS

This section consists of 1 multifunction instruction: SPU.

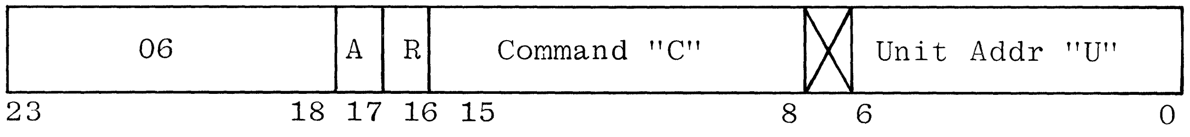
The FST-1 uses only one multifunction I/O instruction, SPU.

3.13.1 SPU Select Peripheral Unit

Definition: Select Peripheral Unit "U"

Cycles Required: 1

Instruction Format:



The Select Peripheral Unit is a multifunction instruction. These functions are:

(1) the addressing of a peripheral unit for selection;

(2) the transfer of a command to the addressed unit;

(3) the transfer of up to 24 bits of information in either direction between the addressed unit and the CPU's accumulator;

(4) the transfer of the unit's status to the CPU.

"U" defines the unit to be selected by the SPU command. The seven bits in this field allow the selection of up to 128 unique units.

"C" defines the command to the addressed peripheral unit. During the SPU execution, this command field is gated to the peripheral unit, where it is decoded and used to initiate a peripheral operation. (For a description of the commands for each peripheral unit, refer to the sections describing the particular peripheral unit.)

The "A" and "R" bits define a transfer between the addressed peripheral and the CPU accumulator. If the A bit is a "1", there will be an information transfer. If R = 0, the transfer will be from the CPU accumulator to the peripheral unit; if R = 1, the transfer will be from the peripheral unit to the CPU accumulator. If the A bit is a "0", no transfer will occur. During each SPU execution, the addressed peripheral will send status to the CPU. This status is stored in the GT, EQ, LT and BE indicators. Refer to the section devoted to peripheral controllers for interpretation of indicators following an SPU command.

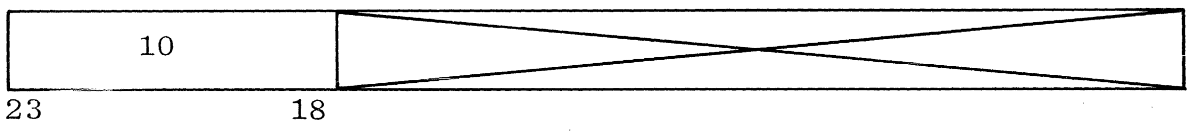
3.14 NO OPERATION INSTRUCTION

3.14.1 NOP No Operation

Definition: No operation

Cycles Required: 1

Instruction Format:



Description:

No operation of any kind will occur on the instruction. Its main use is in debugging so as to provide spare instruction slots in assembly programs.

Assembler Format: NOP

Appendix A
FST—1 Abbreviations

ABS	Accumulator Bus System
AIS	Accumulator Interface System
AR	(Memory) Address Register
A-Register (A)	Accumulator (24 bits)
ARL	Access Request Line
AU	Arithmetic Unit
BE	Bit Equal Indicator
B-Register (B)	Memory Buffer Register (24 bits)
CP	Card Punch
CR	Card Reader
CR	Command Register (24 bits)
CRC	Card Reader Controller
CRS	Card Reader System
DCB	Data Control Block
DCW	Data Control Word
DMA	Direct Memory Access
EQ	'Equal' Indicator
E-Register (E)	Extension Register (24 bits)
FCS	Fairchild Computer System
GT	'Greater Than' Indicator
I	Indirect Address Indicator
IAM	Indirect Address Modification
IE	Interrupt Enable Flip-Flop
IP	Interrupt Priority
IR	Interrupt Required Flip-Flop
J	Shift Constant
K	Value Substituted for I and X on Conditional Branches
LT	'Less Than' Indicator
M	Memory Location
MAG	Memory Access Gained Flip-Flop
MAR	Memory Address Register
MIS	Memory Interface System
O	Operand Address
OC	Instruction Operation Code
OV	'Overflow' Indicator
P-Register (P)	Program Counter (14 bits)
R-Register (R)	Interrupt Address Director Register (6 bits)
SN	Nth Output of the AU
TOL	Tape Object Loader
Ua	"A" Input to AU
Ub	"B" input to AU
W-Register (W)	Console Switch Register (24 bits)
X	Index Address
X-Register	Index Registers X0----X7 (14 bits)

Appendix B Instruction Mnemonics

A.1 (OPCODES SORTED BY ASCENDING ALPHA OPCODE)

OPCODE	MNEMONIC	CODE DESCRIPTION	CYCLES
	ABS	ABSOLUTE PROGRAM LOCATOR	
20000000	ADD	ADD	2
26000000	AND	LOGICAL AND	2
36000000	AOM	ADD ONE TO MEMORY	4
06403400	ARD	ALTERNATE READ	1
06613400	ARDS	ALTERNATE READ STATUS	1
06422400	ASPAC	ALTERNATE SPACE	1
11000000	ATX	ADD TO INDEX	2
07000000	AUG	AUGMENT	
06423400	AWRIT	ALTERNATE WRITE	1
00000000	BAH	BRANCH AND HALT	1
02000000	BAT	BRANCH ON A-REGISTER TEST	1
03040000	BBC	BRANCH BIT COMPARE	1
03200000	BE	BRANCH IF EQUAL	1
03400000	BG	BRANCH IF GREATER	1
03600000	BGE	BRANCH IF GREATER OR EQUAL	1
03100000	BL	BRANCH IF LESS	1
03300000	BLE	BRANCH IF LESS OR EQUAL	1
02100000	BN	BRANCH IF NEGATIVE	1
03500000	BNE	BRANCH NOT EQUAL	1
02500000	BNEZ	BRANCH IF NOT EQUAL TO ZERO	1
02300000	BNZ	BRANCH IF NEGATIVE OR ZERO	1
02040000	BO	BRANCH IF ODD	1
03000000	BOI	BRANCH ON INDICATOR	1
04000000	BOS	BRANCH ON STATE	1
04440000	BOV	BRANCH ON OVERFLOW	1
02400000	BP	BRANCH IF POSITIVE	1
02600000	BPZ	BRANCH IF POSITIVE OR ZERO	1
01000000	BRU	BRANCH UNCONDITIONAL	1
12000000	BSM	BRANCH STORE RETURN AT M	2
	BSS	BLOCK STORAGE SIZE	
13000000	BSZ	BRANCH STORE RETURN AT ZERO	2
02200000	BZ	BRANCH IF ZERO	1
12000000	CALL	SUBROUTINE CALL	
23000000	CAM	COMPARE A WITH MEMORY	2
30000000	DADD	DOUBLE ADD	3
	DATA	DATA DEFINITION	
35000000	DIV	DIVIDE	26
31000000	DLD	DOUBLE LOAD	3
07034000	DSA	DOUBLE SHIFT AROUND	

A.1 (OPCODES SORTED BY ASCENDING ALPHA OPCODE) (Continued)

OPCODE	MNEMONIC	CODE DESCRIPTION	CYCLES
07036000	DSL	DOUBLE SHIFT LEFT	
07030000	DSR	DOUBLE SHIFT RIGHT	
33000000	DST	DOUBLE STORE	3
32000000	DSUB	DOUBLE SUBTRACT	3
07014000	DTC	DOUBLE TWO'S COMPLEMENT	2
	END	PROGRAM TERMINATOR	
21000000	EOR	EXCLUSIVE OR	2
	EQU	EQUIVALENCE	
06010000	ETST	ERROR TEST	1
07010000	EXC	EXCHANGE A AND E	1
06051500	FSKIPB	SKIP FILE BACKWARD	1
06041500	FSKIPF	SKIP FILE FORWARD	1
07012400	IDA	INTERRUPT DISABLE	1
07004400	IEN	INTERRUPT ENABLE	1
	INSEQ	CHECK SEQUENCE NUMBERS	1
13000000	LAX	LOAD A FROM INDEX	1
24000000	LDA	LOAD A-REGISTER	2
25000000	LDE	LOAD E-REGISTER	2
07032000	LDS	LOGICAL DOUBLE SHIFT	
05000000	LDX	LOAD INDEX	2
	LIST	PRODUCE ASSEMBLY LISTING	
07022000	LS	LOGICAL SHIFT A	
07000000	LXA	LOAD INDEX FROM A	1
34000000	MUL	MULTIPLY	25
	NOLIST	NO ASSEMBLY LISTING	
	NOOBJ	NO OBJECT PROGRAM	
10000000	NOP	NO OPERATION	1
	NOSEQ	STOP SEQUENCE CHECK	
	OBJECT	PRODUCE OBJECT PROGRAM	
27000000	OR	OR (INCLUSIVE)	2
	ORG	ORIGINATION CONTROL	
	PAGE	PAGINATION CONTROL	
06001000	PCOMP	PRIORITY COMPLETE	1
06011000	POFF	PRIORITY OFF	1
06013000	PON	PRIORITY ON	1
00000000	PROC	SUBROUTINE ENTRY POINT	
00000000	PZE	POSITIVE ZERO (ENTRY PT)	
06401400	RD	READ	1

A.1 (OPCODES SORTED BY ASCENDING ALPHA OPCODE) (Continued)

OPCODE	MNEMONIC	CODE DESCRIPTION	CYCLES
06611400	RDS	READ STATUS	1
06501500	RDT	READ (MAGNETIC) TAPE	1
06601400	RDTT	READ TELETYPE	1
06611700	REWC	READ EXCESS WORD COUNT	1
06000500	REWIND	REWIND TAPE	1
06011500	RSKIPB	SKIP RECORD BACKWARD	1
06001500	RSKIPF	SKIP RECORD FORWARD	1
07006000	RSR	READ SWITCH REGISTER	1
07012000	RST	RESET STATE	1
17000000	RUM	REPLACE UNDER MASK	2
07024000	SA	SHIFT A AROUND LEFT	
06461500	SKWR	SKIP AND WRITE	1
07026000	SL	SHIFT A LEFT	
37000000	SOM	SUBTRACT ONE FROM MEMORY	4
06420400	SPAC	SPACE	1
06000000	SPU	SELECT PERIPHERAL UNIT	1
07020000	SR	SHIFT A RIGHT	
07004000	SST	SET STATE	1
14000000	STA	STORE A-REGISTER	2
15000000	STE	STORE E-REGISTER	2
06000000	STST	STATUS TEST	1
16000000	STX	STORE INDEX	2
22000000	SUB	SUBTRACT	2
	SYM	PRODUCE SYMBOL TABLE	
07002000	TCA	TWO'S COMPLEMENT A	1
06000400	TOF	TOP-OF-FORM	1
	TPASS	TWO PASS ASSEMBLY	
06421400	WRIT	WRITE	1
06061500	WRITM	WRITE TAPE MARK	1

A.2 (OPCODES SORTED BY ASCENDING OCTAL OPCODE)

OPCODE	CODE	CODE DESCRIPTION	CYCLES
	ABS	ABSOLUTE PROGRAM LOCATOR	
	BSS	BLOCK STORAGE SIZE	
	DATA	DATA DEFINITION	
	END	PROGRAM TERMINATOR	
	EQU	EQUIVALENCE	
	INSEQ	START SEQUENCE CHECK	
	LIST	PRODUCE ASSEMBLY LISTING	
	NO SEQ	STOP SEQUENCE CHECK	
	NOLIST	NO ASSEMBLY LISTING	
	NOOBJ	NO OBJECT PROGRAM	
	OBJECT	PRODUCE OBJECT PROGRAM	
	ORG	ORIGINATION CONTROL	
	PAGE	PAGINATION CONTROL	
	SYM	PRODUCE SYMBOL TABLE	
	TPASS	TWO PASS ASSEMBLY	
00000000	BAH	BRANCH AND HALT	1
00000000	PROC	SUBROUTINE ENTRY POINT	
00000000	PZE	POSITIVE ZERO (ENTRY PT)	
01000000	BRU	BRANCH UNCONDITIONAL	1
02000000	BAT	BRANCH ON A-REGISTER TEST	1
02040000	BO	BRANCH IF ODD	1
02100000	BN	BRANCH IF NEGATIVE	1
02200000	BZ	BRANCH IF ZERO	1
02300000	BNZ	BRANCH IF NEGATIVE OR ZERO	1
02400000	BP	BRANCH IF POSITIVE	1
02500000	BNEZ	BRANCH IF NOT EQUAL TO ZERO	1
02600000	BPZ	BRANCH IF POSITIVE OR ZERO	1
03000000	BOI	BRANCH ON INDICATOR	1
03040000	BBC	BRANCH BIT COMPARE	1
03100000	BL	BRANCH IF LESS	1
03200000	BE	BRANCH IF EQUAL	1
03300000	BLE	BRANCH IF LESS OR EQUAL	1
03400000	BG	BRANCH IF GREATER	1
03500000	BNE	BRANCH NOT EQUAL	1
03600000	BGE	BRANCH IF GREATER OR EQUAL	1
04000000	BOS	BRANCH ON STATE	1
04440000	BOV	BRANCH ON OVERFLOW	1
05000000	LDX	LOAD INDEX	1
06000000	SPU	SELECT PERIPHERAL UNIT	1
06000000	STST	STATUS TEST	1
06000400	TOF	TOP-OF-FORM	1
06000500	REWIND	REWIND TAPE	1
06001000	PCOMP	PRIORITY COMPLETE	1
06001500	RSKIPF	SKIP RECORD FORWARD	1
06010000	ETST	ERROR	1

A.2 (OPCODES SORTED BY ASCENDING OCTAL OPCODE) (Continued)

OPCODE	CODE	CODE DESCRIPTION	CYCLES
06011000	POFF	PRIORITY OFF	1
06011500	RSKIPB	SKIP RECORD BACKWARD	1
06013000	PON	PRIORITY ON	1
06041500	FSKIPF	SKIP FILE FORWARD	1
06051500	FSKIPB	SKIP FILE BACKWARD	1
06061500	WRITM	WRITE TAPE MARK	1
06401400	RD	READ	1
06403400	ARD	ALTERNATE READ	1
06420400	SPAC	SPACE	1
06421400	WRIT	WRITE	1
06422400	ASPAC	ALTERNATE SPACE	1
06423400	AWRIT	ALTERNATE WRITE	1
06461500	SKWR	SKIP AND WRITE	1
06501500	RDT	READ (MAGNETIC) TAPE	1
06601400	RDTT	READ TELETYPE	1
06611400	RDS	READ STATUS	1
06611700	REWC	READ EXCESS WORD COUNT	1
06613400	ARDS	ALTERNATE READ STATUS	1
07000000	AUG	AUGMENT	
07000000	LXA	LOAD INDEX FROM A	1
07002000	TCA	TWO'S COMPLEMENT A	1
07004000	SST	SET STATE	1
07004400	IEN	INTERRUPT ENABLE	1
07006000	RSR	READ SWITCH REGISTER	1
07010000	EXC	EXCHANGE A AND E	1
07012000	RST	RESET STATE	1
07012400	IDA	INTERRUPT DISABLE	1
07014000	DTC	DOUBLE TWO'S COMPLEMENT	2
07016000	DSN	DOUBLE SHIFT NORMALIZED	
07020000	SR	SHIFT A RIGHT	
07022000	LS	LOGICAL SHIFT A	
07024000	SA	SHIFT A AROUND LEFT	
07026000	SL	SHIFT A LEFT	
07030000	DSR	DOUBLE SHIFT RIGHT	
07032000	LDS	LOGICAL DOUBLE SHIFT	
07034000	DSA	DOUBLE SHIFT AROUND	
07036000	DSL	DOUBLE SHIFT LEFT	
10000000	NOP	NO OPERATION	1
11000000	ATX	ADD TO INDEX	2
12000000	BSM	BRANCH STORE RETURN AT M	2
12000000	CALL	SUBROUTINE CALL	
13000000	BSZ	BRANCH STORE RETURN AT ZERO	2
14000000	STA	STORE A-REGISTER	2

A.2 (OPCODES SORTED BY ASCENDING OCTAL OPCODE) (Continued)

OPCODE	CODE	CODE DESCRIPTION	CYCLES
15000000	STE	STORE E-REGISTER	2
16000000	STX	STORE INDEX	2
17000000	RUM	REPLACE UNDER MASK	2
20000000	ADD	ADD	2
21000000	EOR	EXCLUSIVE OR	2
22000000	SUB	SUBTRACT	2
23000000	CAM	COMPARE A WITH MEMORY	2
24000000	LDA	LOAD A-REGISTER	2
25000000	LDE	LOAD E-REGISTER	2
26000000	AND	LOGICAL AND	2
27000000	OR	OR (INCLUSIVE)	2
30000000	DADD	DOUBLE ADD	3
31000000	DLD	DOUBLE LOAD	3
32000000	DSUB	DOUBLE SUBTRACT	3
33000000	DST	DOUBLE STORE	3
34000000	MUL	MULTIPLY	25
35000000	DIV	DIVIDE	26
36000000	AOM	ADD ONE TO MEMORY	4
37000000	SOM	SUBTRACT ONE FROM MEMORY	4
40000000	LAX	LOAD A FROM INDEX	1

Section 4
FST—1 Instruction Execution And Timing

4.1 INTRODUCTION

This section describes the execution for each of the FST-1 instructions. A timing diagram and data flow diagram for each instruction form the basis for each of the instruction description. Since the logic design for the FST-1 CPU was derived from these diagrams, they constitute the most important form of information necessary to the understanding of the CPU operation.

4.2 PHASE TIMER

The Phase Timer for the FST-1 consists of a five (5) flip flop (T1, T2, T3, T4, and T5) shift register contained on the CPU clock board. Only one of the flip flops in this timer can be on at any one time. The Phase Timer is shifted every 350n sec which allows it to cycle every 1.75 usec or once per memory cycle.

The system clock as illustrated in Figure II-4-1 is used to shift the Phase Timer. The system clock consists of negative going pulses approximately 90n sec in duration which occur every 350 nanoseconds.

The Phase Timer divides the memory cycle into five unique times which are used to synchronize address and data transfers to and from the memory system. Figure II-4-1 illustrates the relationship between the Phase Timer and the timing diagrams which are used in the illustration of each instruction timing.

During the T2 phase time, the CPU system will attempt to get access to one of the core memory banks. If it is not successful, the clocks to the CPU are turned off and the CPU waits until the next T2 time to request memory access again. If access is gained during T2 time, the CPU will present an address to the memory during T3 time and will signal the memory whether it will read from or write into the addressed memory location.

For a read memory operation, the CPU will gate the data from memory to its own registers during the T1 phase time and execute the transfer with the clock occurring at the end of the T1 phase time. For a write to memory operation, the CPU will gate the data to be written into memory onto the appropriate memory bus during the T5 phase time.

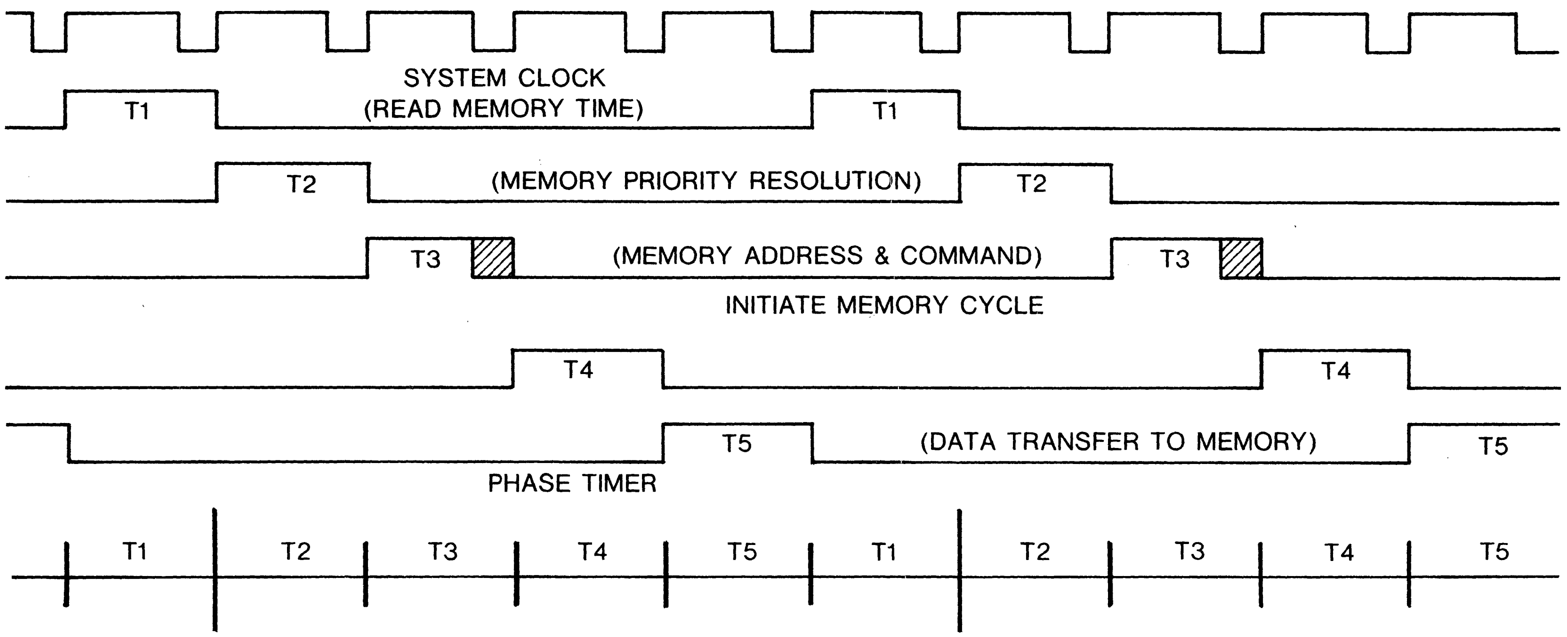


FIGURE II-4-1. SYSTEM CLOCK & PHASE TIMER

4.3 TIMING DIAGRAMS

Timing diagrams are used to illustrate the sequence of events required in the execution of the FST-1 instructions. Figure II-4-1 illustrates the relationship between the timing diagram and the occurrence of the phase time signals. Figure II-4-2 illustrates how the timing diagram is used to describe the relationship between the occurrence of control signals and the phase timer.

In the upper part of Figure II-4-2 there are four signals shown on the Operand Fetch timing diagrams. The nomenclature in this diagram is to be interpreted as follows: The two signals TOF and GMB are present during all 5 phase times of the operand fetch. The signal GOM is present during T3 phase time only and GCPB is present only during T1 phase time. (The term "present" is synonymous with the terms energized, in the "one" state, etc.)

The system clocks cause the transitions from one phase time to another. The signal GOM, which is shown present during T3 will actually become energized after the clock occurring during T2 and will overlap the clock occurring at the end of T3.

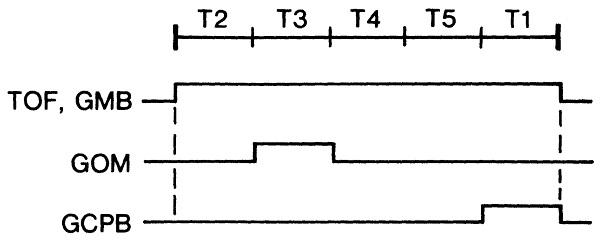
4.4 OPERAND AND INSTRUCTION FETCH CYCLES

The Operand Fetch Cycle is common to most FST-1 instructions and the Instruction Fetch is a part of every instruction. To simplify the explanation of the instruction timing, the signals necessary to accomplish the two operations have been separated and illustrated in II-4-2.

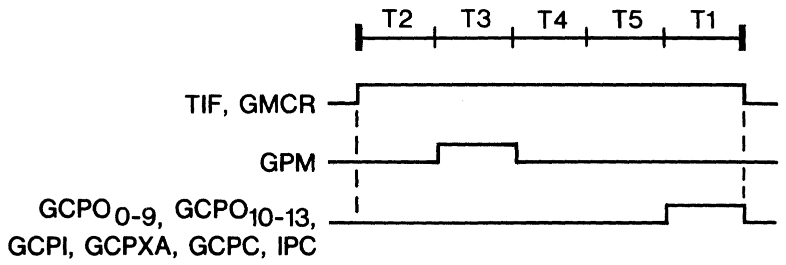
4.4.1 Instruction Fetch Cycle

The function of the Instruction Fetch Cycle is to obtain the next instruction to be executed from memory and set it into the CPU's Command Register (CR).

The program (P) counter normally contains the address of the memory location containing the instruction which is to be executed following the execution of the current instruction. (The term, "current instruction," refers to the instruction occupying the Command Register. To fetch this instruction from memory, the CPU must gain memory access at time T2 of the Instruction Fetch Cycle, transfer the contents of the program counter to memory at time T3 with a read command, and gate the contents of the memory bus to the CR at T1 time. (See Figure II-4-2.)



OPERAND FETCH CYCLE



INSTRUCTION FETCH CYCLE

FIGURE II-4-2. OPERAND AND INSTRUCTION FETCH CYCLE

The control flip-flop TIF (Time of Instruction Fetch) is set throughout the Instruction Fetch cycle. The GPM (Gate P to Memory) signal causes a transfer of the P counter contents to memory at time T3. At time T1 of the Instruction Fetch Cycle all 24-bits of the addressed memory location are transfered onto the memory bus and into the Command Register by signals GCPO 0-9 (Gate a Clock Pulse to Operand address bits 0 thru 9), GCPO 10-13 (Gate a Clock Pulse to Operand address bits 10 thru 13), GCPI (Gate a Clock Pulse to the Indirect bit), GCPXA (Gate a Clock Pulse to the INDEX bits), GCPC (Gate a Clock Pulse to the Command bits). This partitioning of the clock pulses to the command register allows control of the several fields in other operations.

Signal IPC (Increment the Program Counter) increases the program counter by one during time T1, setting up the program counter for the next Instruction Fetch Cycle.

4.4.2 Operand Fetch Cycle

The function of the Operand Fetch Cycle is to retrieve a particular piece of data (an operand) from the FST-1 memory and transfer it via the buffer register (B) into the CPU's memory. To do this, it is necessary for the CPU to request memory access during T2, transfer an address to memory during T3 and read the contents of the memory bus at T1.

The flip flop TOF is always set during the Time of an Operand Fetch. If the CPU gains memory access at time T2, the signal GOM (Gate Operand address register to Memory) will be present during time T3. This signal causes the address portion of the command register, identifying the desired operand location to be presented to memory. At the end of the time T3, this address will be loaded into the memory's address register and a read memory cycle initiated.

Throughout the Operand Fetch cycle, signal GMB (Gate Memory to B register) prepares the input to the B register for the transfer of data. The data (operand) is then gated onto the memory output bus and into the B register during time T1 by signal GCPB (Gate Clock Pulse to B register), completing the operand fetch cycle.

If the instruction being executed requires only one operand fetch, TOF is reset at the end of the T1 phase of the cycle. If two operand fetches are required, the above operations are executed twice and TOF is reset at the end of the second operand fetch cycle. Only one operand can be fetched at a time, but an operand can be double length.

4.4.3 Address Modification

Operand addresses may be modified by indexing in which the contents of a specified index register are added to the operand address before the memory fetch, and by indirection in which the memory fetch produces not data, but another address which may in turn be subject to indexing and indirection. The order of modification is first index, then indirection.

4.4.3.1 Indexing

Index register modification of an address is available to all commands with the exception of SPU (Select Peripheral Unit), BAT (Branch on Accumulator Test), BOS (Branch On State), BOI (Branch On Indicator), LXA (Load Index from Accumulator), LAX (Load Accumulator from Index), ATX (Add to Index), STX (Store Index), and LDX (Load Index).

An indexing operation is specified and directed by bits 17, 16, and 15 of the command register word. These bits indicate the address of the specific index register to be used (1 through 7). If bits 17, 16, and 15 are a "0," indexing has not been specified. When any one or more of these three bits is a "1," signal IDX (Index) is present and acts to suppress the execution of any command for which index modification is valid until the modification occurs (see Figure II-4-3).

Throughout the TOF cycle, the contents of the addressed index register are gated to the B input of the adder by control signal GXUB (Gate Index to Adder Input B), Figure II-3-3. Simultaneously, the operand field of the Command register (bits 0 through 13) is gated to the A input of the adder by the control signal GOUA (Gate Operand field to Adder Input A), the resulting adder output (sum of A and B adder inputs) is gated to the input of the operand address field portion of the command register by signal GSO (Gate Sum to O). At time T5 the adder output has had time to settle and the sum of the two adder inputs (operand address plus index register contents) is clocked into the operand address field of the command register. This is accomplished by signals GCP0-9 (Gate Clock Pulse to Operand field, 0 through 9) and GCP10-13 (Gate Clock Pulse to Operand field, bits 10 through 13) as shown in Figure II-4-3. At time T1 of the memory cycle, signal GCPXA (Gate Clock Pulse to Index Address) clears the index address bits in the command register, terminating signal IDX and the indexing function. Note that as an IDX signal results from at least one of the index bits (bit 17, 16, or 15) being a "1," it is therefore possible to address only index registers X1 through X7 and not X0.

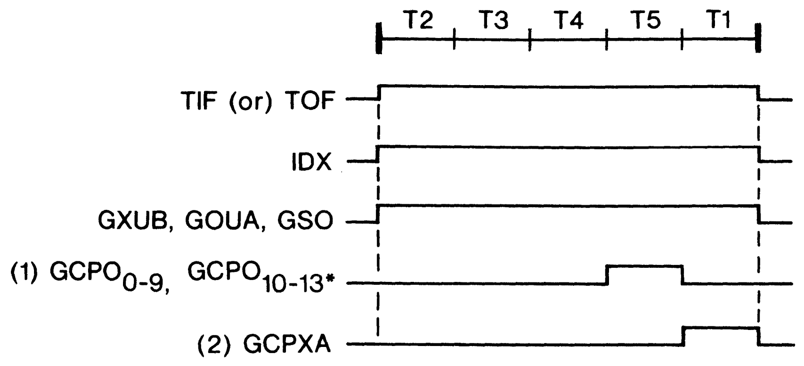
4.4.4 Indirect Addressing

Indirect addressing may be used to modify the address of all FST-1 Commands with the exception of SPU, BAT, BOS, and BOI. The state of bit 14 (the Indirect bit) of the command register determines if an indirect address function is to be performed. If bit 14 is a "0" an indirect addressing function does not occur. When bit 14 is a "1" indirect addressing will always occur, with the four exceptions listed above.

(See Figure II-4-4). When the indirect bit is set, signal INA (Indirect Address) inhibits the execution of any command for which indirect addressing is valid until the indirect cycle is complete. The INA signal is itself inhibited by an IDX (Index) signal, so that indirection must follow an indexing function if both functions are desired. During an indirect address modification cycle (a TOF as shown in Figure II-4-4), signals INA (Indirect Address) and GMCR (Gate Memory contents to Command Register) are always present. At time T3 of the memory cycle, signal GOM (Gate Operand Field to Memory) gates the operand field bits (bits 0 through 13) to memory. A READ command is also sent to memory during the indirect address cycle and, during time T1, data is read from the addressed memory location and gated into the command register by signal GMCR (Gate Memory to Command Register). Signals GCP0-9, GCP10-13, GCPI (Gate Clock Pulse to Indirect bit), CCPXA (Gate Clock Pulse to Index bits), clock the data from memory into the command register.

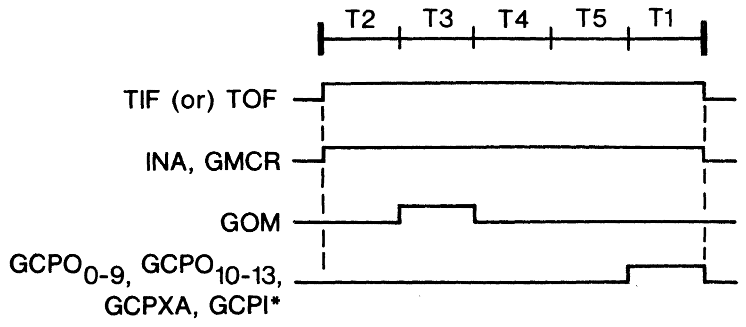
In this manner the index address (bits 17, 16, and 15), indirect bit (bit 14), and operand address (bits 13 through 0) are replaced in the command register. Note that the original command (bits 23 through 18) is not altered.

If any index bit is a "1" through replacement during indirection, another index modification as described will occur. If the indirect bit of the data fetched from memory is a "1," another indirect address modification will follow, as described above. There is no limit to the number of subsequent indirection memory cycles, and it is therefore, possible to have a closed loop of indirect addresses from which there is no exit. Subsequent indirection will be terminated when the indirect bit fetched from memory is a "0." An indirect loop may be terminated by lifting the SMC (Single Memory Cycle) switch on the central processor front panel, and entering a non-indirect instruction.



(1) Gates the sum to O (operand field of command register)
(2) Resets index bits in command register (CR)
*GCPO 10-13 does not occur for augmented instructions

FIGURE II-4-3. INDEXING ADDRESS MODIFICATION TIMING DIAGRAM



*Clock M19 through M23 to status register if instruction is a BRU indirect
(M19 $\rightarrow$ BE, M20 $\rightarrow$ LT, M21 $\rightarrow$ EO, M22 $\rightarrow$ GT, M23 $\rightarrow$ OV)

FIGURE II-4-4. INDIRECT ADDRESS MODIFICATION TIMING DIAGRAM

4.5 INSTRUCTION EXECUTION AND TIMING

The execution sequence for each instruction in the FST-1 repertoire is contained in the following paragraphs. The flow charts show the data paths between the major registers used in the execution of each instruction and the gating signals controlling the timing. The following descriptions do not include the timing for indirect or index address modification.

4.5.1 Add Instruction

Figures II-4-5 and II-4-6 are the flow and timing diagrams for the ADD instruction. The execution of the ADD instruction performs the binary addition of the contents of the memory location specified by the operand address portion of the Command Register to the contents of the Accumulator register. If the sum exceeds the limits of the accumulator register, the OV (overflow flip-flop is set.

The ADD instruction starts with an Operand Fetch Cycle (TOF). This is illustrated on the ADD timing diagram (Figure II-4-6). The addition is performed by energizing the signals GAUA, GBUB and GSA. These signals are enabled during the ADD instruction under control of the flip flop TEX1 (Time of Execution 1). GAUA gates the contents of the Accumulator to the arithmetic Unit's A input. GBUB gates the contents of the B register to the arithmetic Unit's B input. GSA gates the Sum output of the arithmetic unit to the Accumulator.

The carry is allowed to propagate through the Arithmetic Unit (AU) until the T5 phase of TEX 1. During T5, the signal GCPA (Gate a Clock Pulse to A) is energized. GCPA will allow A to receive one clock pulse which will enter the Sum into the A register. Also during T5, if the sign of A and B are alike and the sign of the sum is different, the OV flip flop is set signaling an overflow from the A register.

An Instruction Fetch Cycle (TIF) overlaps the operations required to execute the binary addition. This is possible since the data paths needed for an instruction fetch are independent of those used during the addition.

4.5.2 Sub Instruction

Figures II-4-7 and II-4-8 are the flow and timing diagrams for the SUB instruction. The SUB instruction subtracts the contents of the memory location specified by the operand address from the contents of A. The difference is stored in A. The SUB

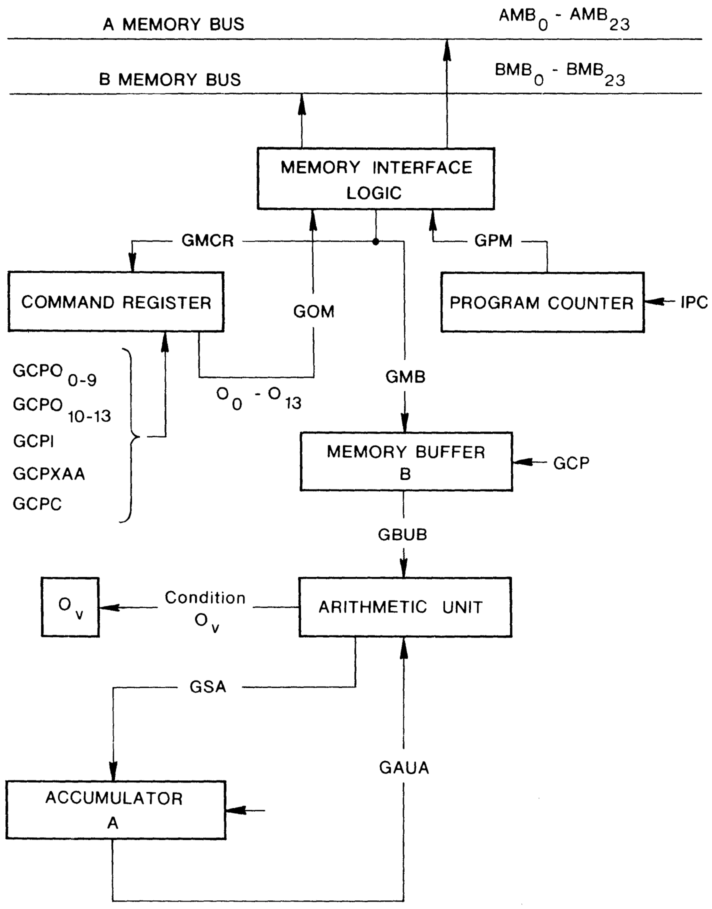


FIGURE II-4-5. ADD INSTRUCTION FLOW DIAGRAM

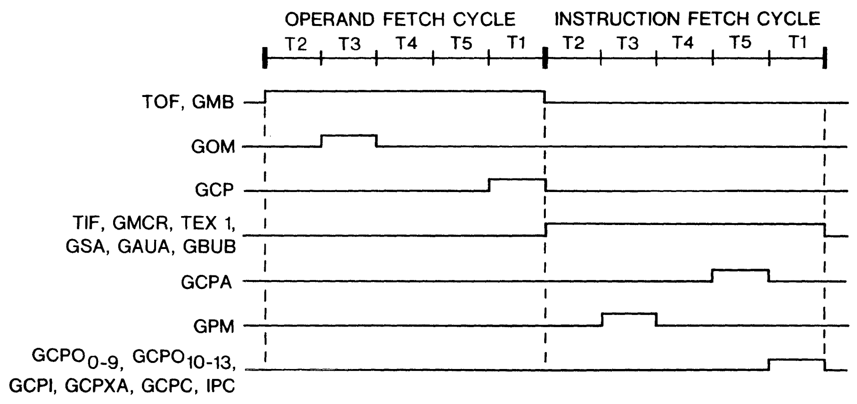


FIGURE II-4-6. ADD INSTRUCTION TIMING DIAGRAM

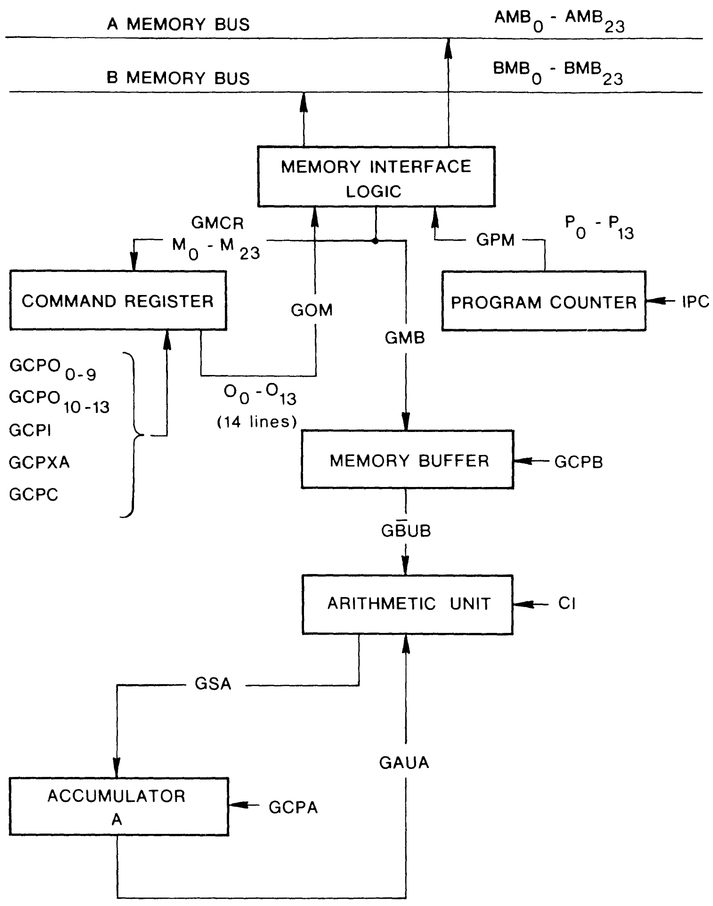


FIGURE II-4-7. SUB INSTRUCTION FLOW DIAGRAM

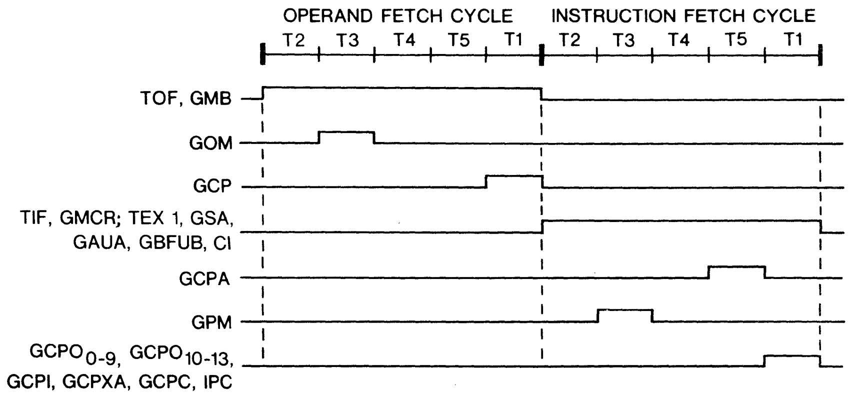

OPERAND FETCH CYCLE
INSTRUCTION FETCH CYCLE
T2 T3 T4 T5 T1 T2 T3 T4 T5 T1
TOF, GMB
GOM
GCP
TIF, GMCR; TEX 1, GSA, GAUA, GBFUB, CI
GCPA
GPM
GCPO0-9, GCPO10-13, GCPI, GCPXA, GCPC, IPC

FIGURE II-4-8. SUB INSTRUCTION TIMING DIAGRAM

instruction, with two exceptions, uses the same control signals and timing as ADD. The first exception is that GBUB is replaced by the signal GBFUB (Gate the complement of B to the arithmetic Unit's B input). The second exception is that the CI (Carry In) signal is present during TIF. Signal CI enters a carry into the least significant bit position of the AU. Together, these two signals present the two's complement of the B register contents to the adder. Addition of a two's complement is equivalent to subtraction.

4.5.3 EOR Instruction

Figures II-4-9 and II-4-10 are the flow and timing diagrams for the EOR instruction. The EOR instruction forms the exclusive OR of the contents of the memory location specified by the operand address and the contents of (A) Accumulator. The results of the EOR are stored in A. The exclusive OR function is obtained by inhibiting carries in the adder unit.

The EOR timing, control, and data paths are identical with those of the ADD instruction except that the IC (Inhibit Carry) signal is energized. The next instruction fetch overlaps the execution, hence TIF is true concurrent with TEX1.

4.5.4 CAM Instruction

Figures II-4-11 and II-4-12 are the flow and timing diagrams for the CAM instruction. The CAM (Compare A and Memory) instruction, compares the contents of A with the contents of the memory location specified by the operand address. The comparison is made by subtraction and is the appropriate setting of the four (4) indicators GT (Greater Than), EQ The result obtained by the CAM instruction is the appropriate setting of the four (4) indicators GT (Greater Than), EQ (EQual), LT (Less Than) and BE (Bit Equal).

During T5, the output of the AU (Arithmetic Unit) determines which of the indicators is to be set. A special set of gates within the adder unit detects a zero sum for setting the EQ indicator. If the sum is non-zero, the condition of the sum sign bit sets either the GT or the LT indicator. Another set of gates forms an AND of the A and B adder input values for setting the BE indicator if any pair of corresponding bits are ones. The appropriate indicator is set with the clock pulse occurring at the end of the T5 phase. GCPA does not occur during CAM so that the A register is not changed. The next instruction fetch overlaps the execution, hence TIF is true concurrent with TEX1.

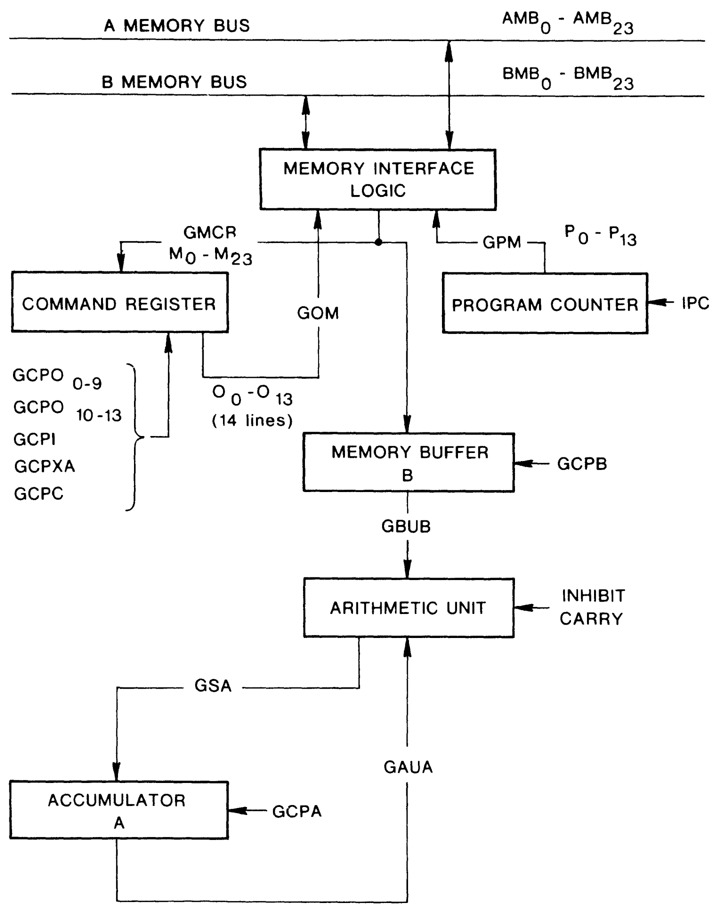


FIGURE II-4-9. EOR INSTRUCTION FLOW DIAGRAM

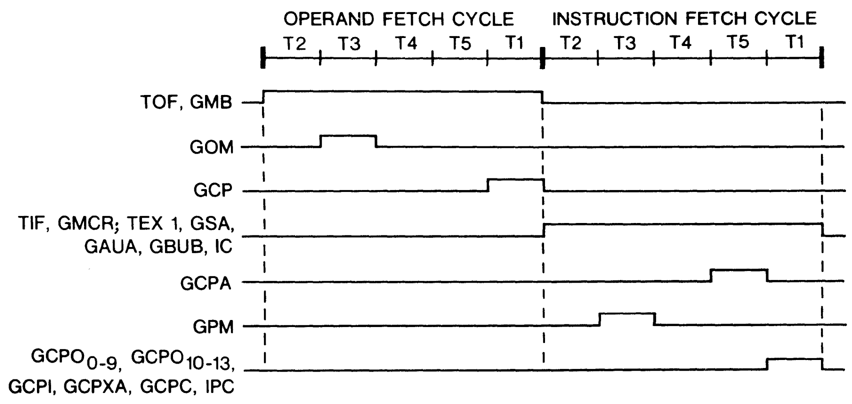


FIGURE II-4-10. EOR INSTRUCTION TIMING DIAGRAM

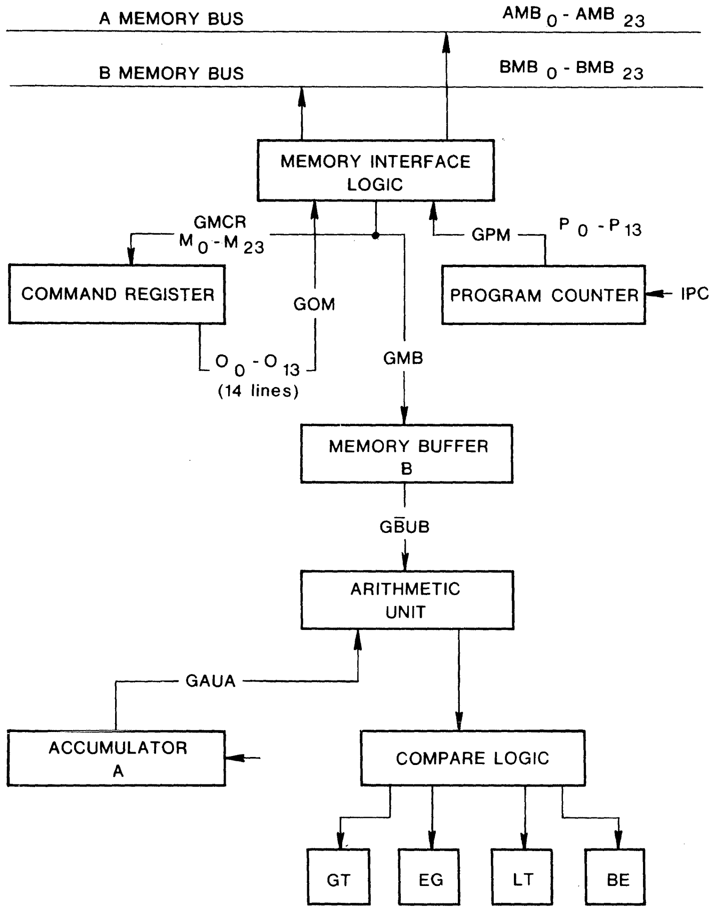


FIGURE II-4-11. CAM INSTRUCTION FLOW DIAGRAM

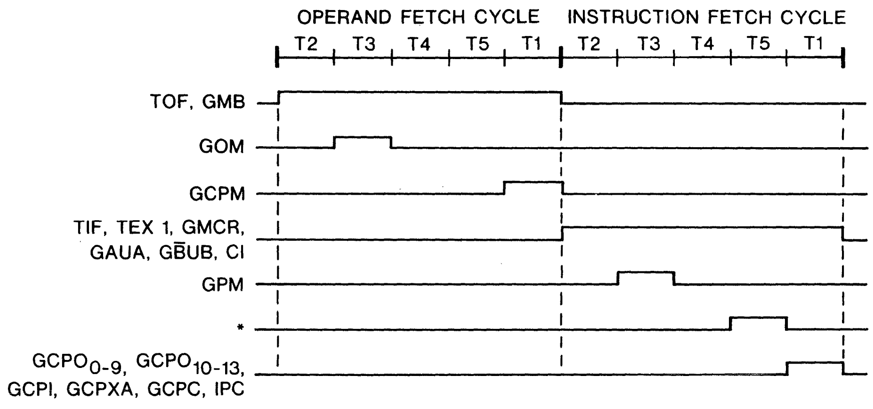

OPERAND FETCH CYCLE
INSTRUCTION FETCH CYCLE
T2 T3 T4 T5 T1 T2 T3 T4 T5 T1
TOF, GMB
GOM
GCPM
TIF, TEX 1, GMCR, GAUA, GB̄UB, CI
GPM
*
GCPO0-9, GCPO10-13, GCPI, GCPXA, GCPC, IPC
*Set/reset indicators depending on outcome of comparison

FIGURE II-4-12. CAM INSTRUCTION TIMING DIAGRAM

4.5.5 LDA Instruction

Figures II-4-13 and II-4-14 are the flow and timing diagrams for the LDA instruction. The LDA (Load A) instruction transfers the contents of the memory location specified by the operand address into A. The LDA operation consists of adding the contents of the operand to zero and placing the sum in A. The LDA instruction uses the same data paths and timing as the ADD instruction, with the exception that the GAUA signal is not energized for LDA. With the B register furnishing the only input to the AU, the sum output of the AU equals the contents of B. GSA in effect gates B to A. GCPA at T5 clocks the operand from AU into A. The next instruction fetch overlaps the execution, hence TIF is true concurrent with TEX1.

4.5.6 LDE Instruction

Figures II-4-15 and II-4-16 are the flow and timing diagrams for the LDE instruction. The execution of an LDE (LOAD E) instruction results in the contents of the memory location specified by the operand address being transferred into the E register. The operand fetch cycle for LDE places the operand in the B register. During TEX1, the contents of B are gated to the AU with GBUB as in LDA. The contents of B are added to zero and the sum is gated to E with GSE (Gate Sum to E). At T5, GCPE (Gate a Clock Pulse to E) into E.

4.5.7 RUM, AND and OR Instruction

These instructions are identical in timing and differ only in the gating structure at the input to A. (Refer to Figures II-4-17 thru II-4-20). Each of these instructions starts with an operand fetch cycle to load an operand into the B register. The RUM instruction causes a selective replacement of data bits in the accumulator. Those bit positions of the extension register which contain ones correspond with the accumulator bit positions which undergo replacement. The corresponding bit position of the buffer register holds the value which replaces the bit in the accumulator. Accumulator bit positions for which the corresponding E-register bit is zero remain unchanged. GBUB, gates the contents of B through the AU.

The Rum gating logic at the input to A is illustrated in Figure II-4-17. GCPA is energized at T5 clocking the results into A. The next instruction fetch overlaps the execution, hence TIF is true concurrent with TEX1.

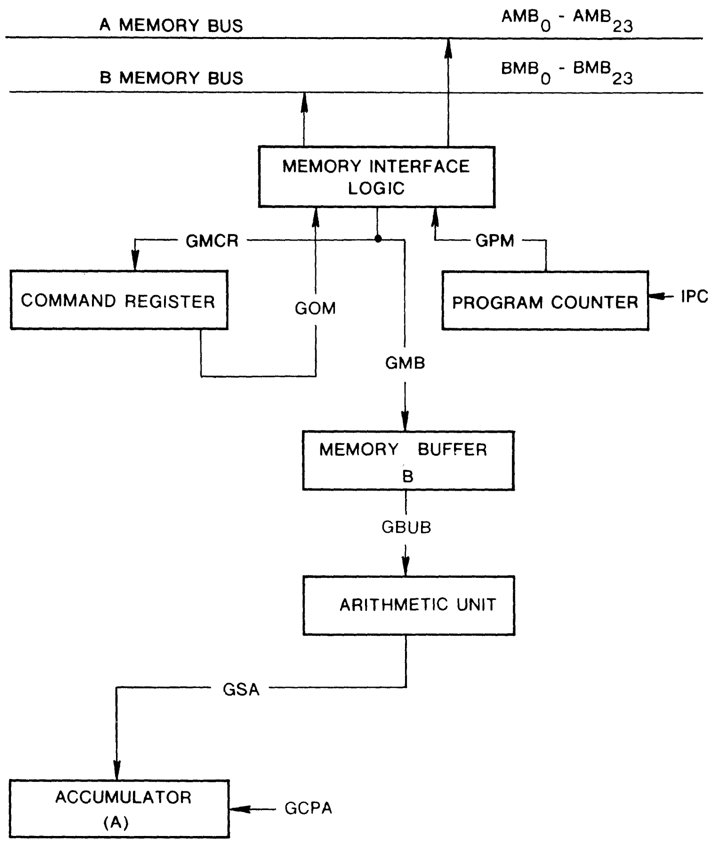


FIGURE II-4-13. LDA INSTRUCTION FLOW DIAGRAM

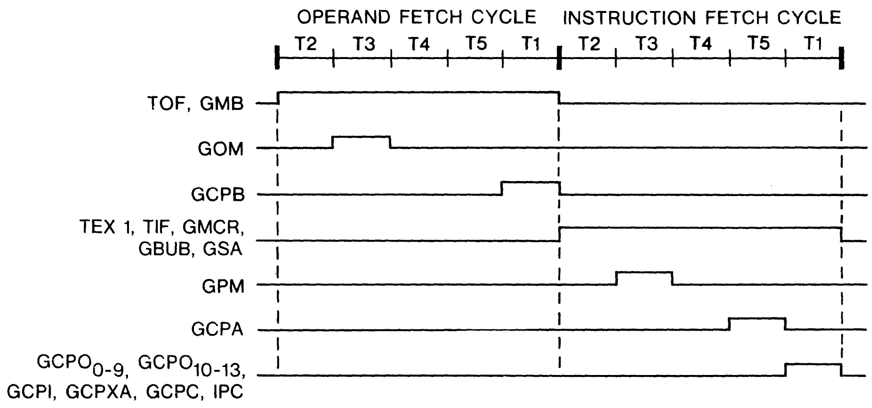


FIGURE II-4-14. LDA INSTRUCTION TIMING DIAGRAM

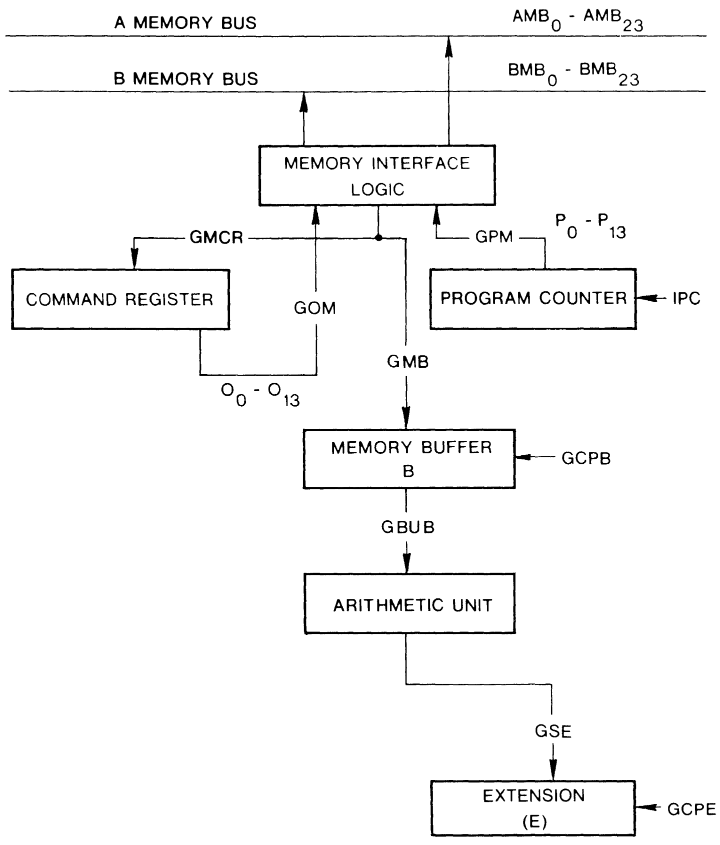


FIGURE II-4-15. LDE INSTRUCTION FLOW DIAGRAM

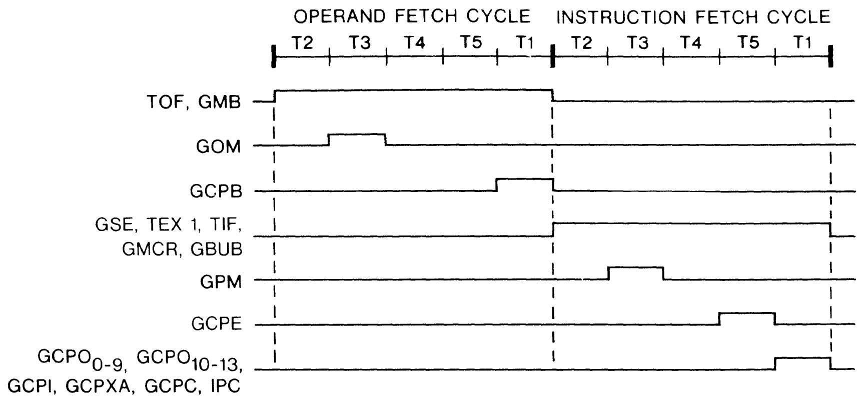


FIGURE II-4-16. LDE INSTRUCTION TIMING DIAGRAM

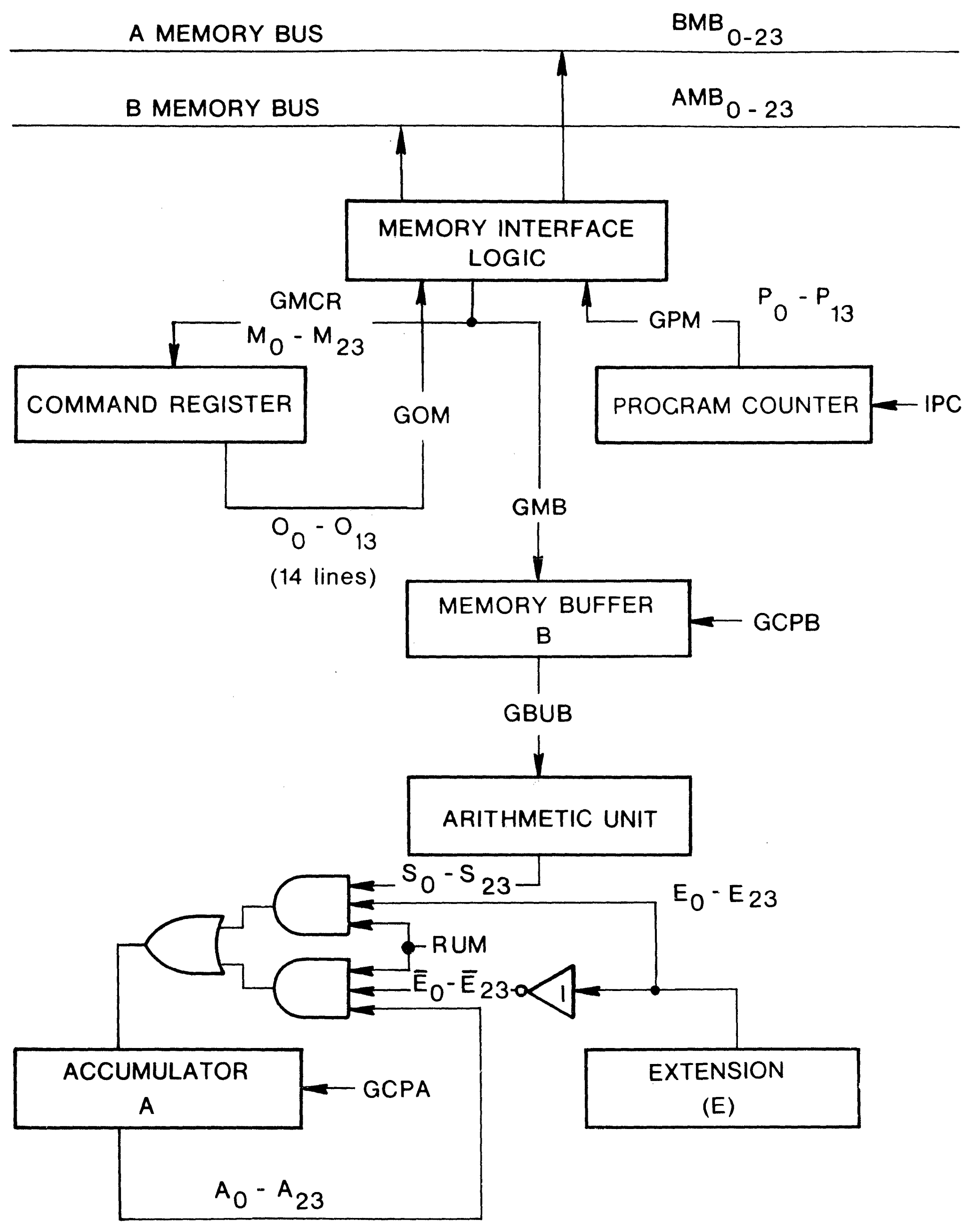


FIGURE II-4-17. RUM INSTRUCTION FLOW DIAGRAM

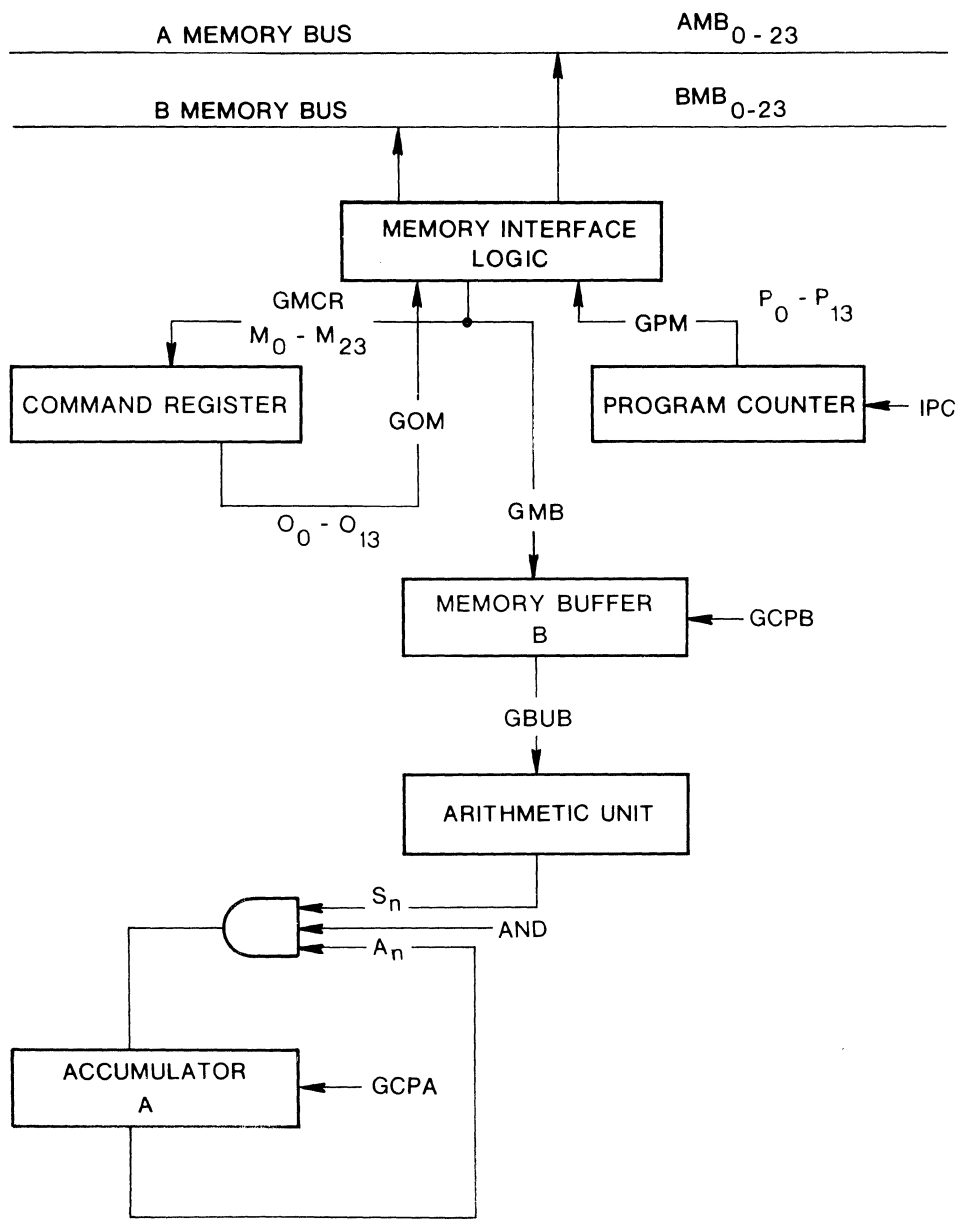


FIGURE II-4-18. AND INSTRUCTION FLOW DIAGRAM

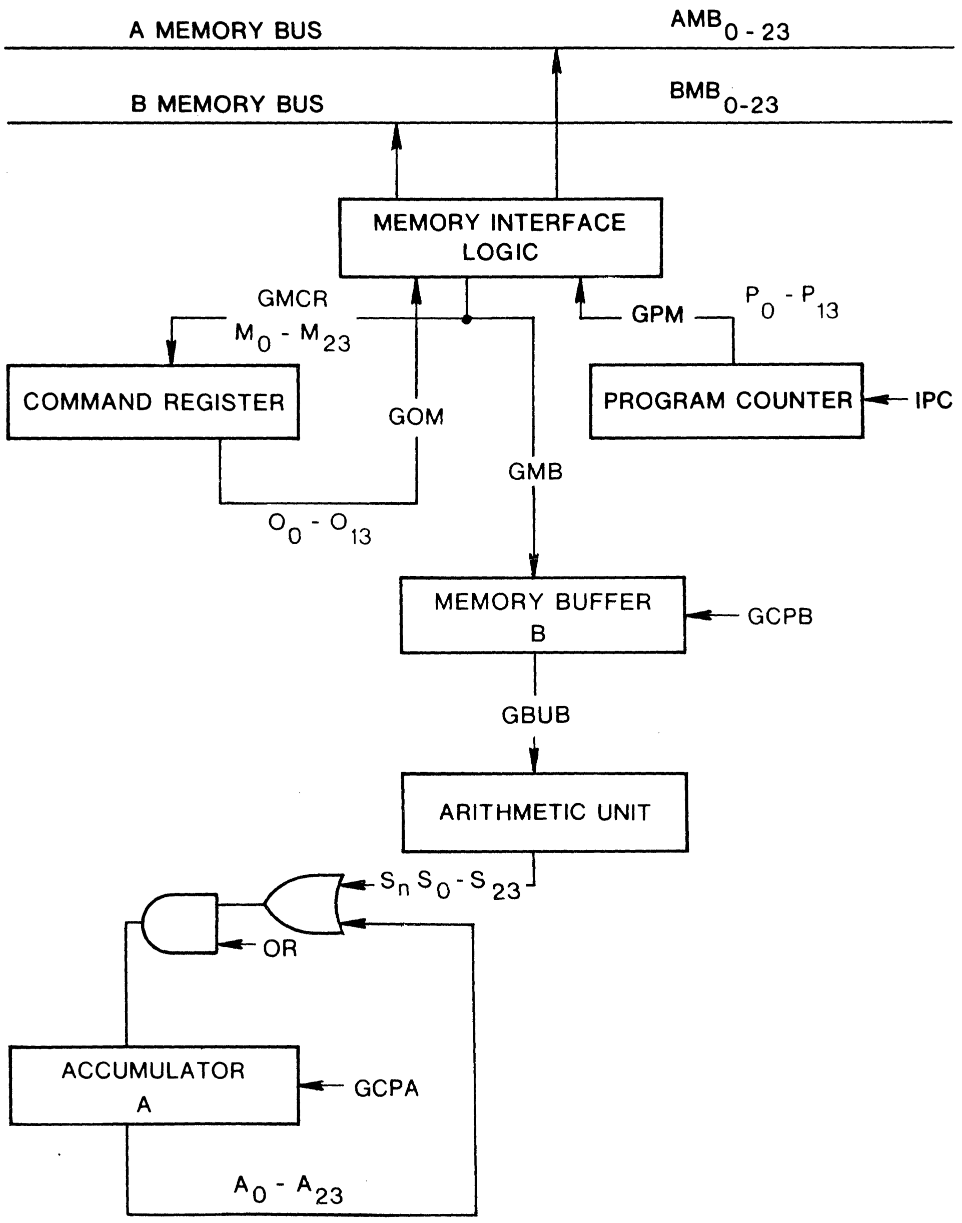


FIGURE II-4-19. OR INSTRUCTION FLOW DIAGRAM

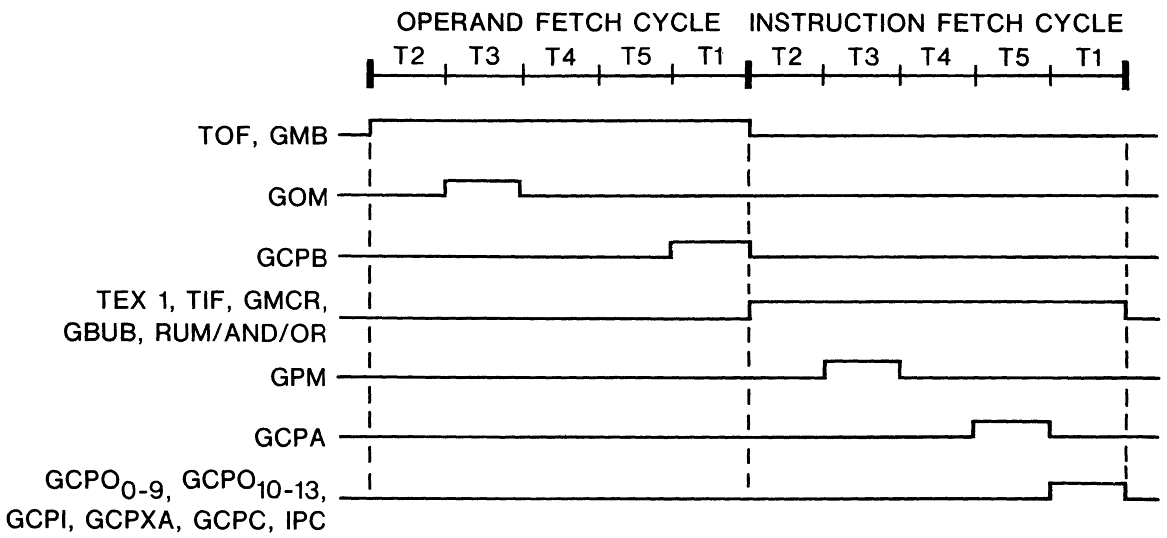


FIGURE II-4-20. RUM/AND/OR INSTRUCTION TIMING DIAGRAM

FAIRCHILD SYSTEMS TECHNOLOGY

The AND instruction replaces each bit in A with the results of the logical AND of each bit position in A with the corresponding bit of the operand in B. During execution (TEX1), GBUB gates the contents of B through the AU to the AND logic at the input to A; (see Figure II-4-18). At T5, GCPA is energized clocking the results from the AND logic into A. The next instruction fetch overlaps the execution.

The OR instruction replaces each bit in A with the results obtained by performing the logical OR of each bit position in A with the corresponding bit of the operand in B. During execution, GBUB gates the contents of B through the AU to the OR logic (see Figure II-4-19). At T5, GCPA is energized clocking the results from the OR logic into A. The next instruction fetch overlaps the execution.

4.5.8 ATX Instruction

Figures II-4-21 and II-4-22 are the flow and timing diagrams for the ATX instruction. The ATX instruction adds the contents of the command register operand address field to the contents of the addressed index register. A comparison of the resulting sum with the contents of the index register addressed by changing bit 15 of the Command Register to zero sets one of the indicators GT, EQ, LT. Although ATX instruction requires no operand fetch, it uses TOF to define the first memory cycle of the instruction's execution. During this cycle the contents of the addressed index register are gated to the AU by the signal GXUB (Gate indeX to Arithmetic Unit's B-input). The contents of the command register operand address field are gated to the AU with GOUA (Gate the Operand address to the arithmetic Unit's A-input). The sum of the contents of X and O is formed on the AU's outputs, S0 through S13, and gated to the command register with GSO (Gate Sum to O). The sum is entered into the command register with the signals GCP0-9 and GCP10-13 which occur at T5. The SUM is gated from the command register to the addressed index register with LDATX (Load Address to Index) and, written into X at T1 with signals WX (Write X).

Also at T1 of the first memory cycle, index bit 15 in the command register is cleared. The signal RX15 (Reset X15) gates X17 and X16 to themselves, while a "0" is gated to X15. GCPXA leaves X17 and X16 unchanged, while X15 is cleared.

During the TEX1 part of ATX, the complement of the addressed index register is gated to the AU with GXFUB (Gate X complement to UB) and the contents of the CR operand address field are gated to the AU with GOUA. The CI signal is also energized. The difference between the contents of the command register address

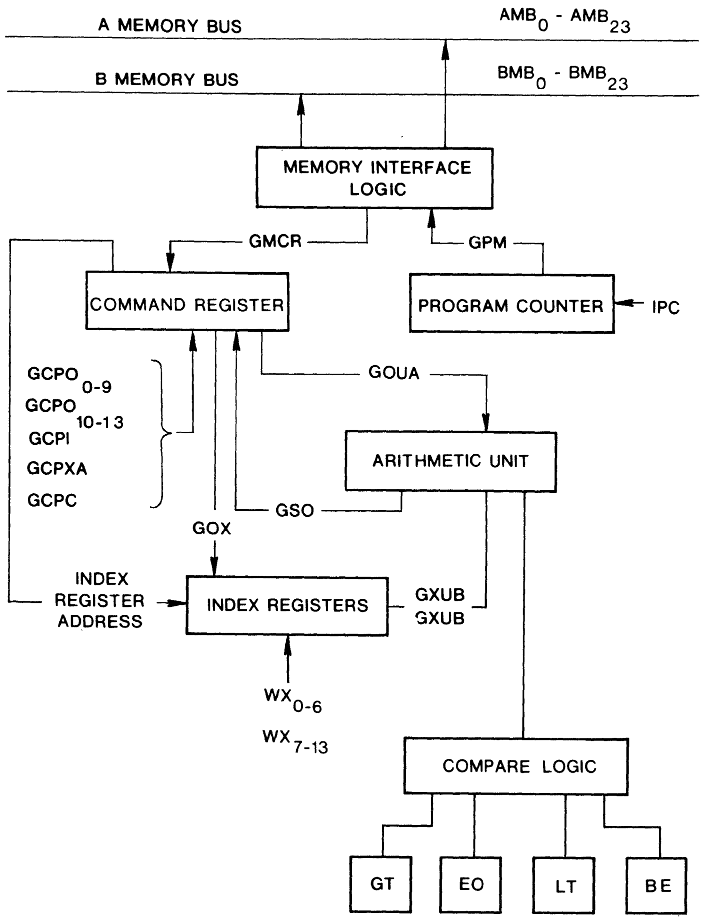


FIGURE II-4-21. ATX INSTRUCTION FLOW DIAGRAM

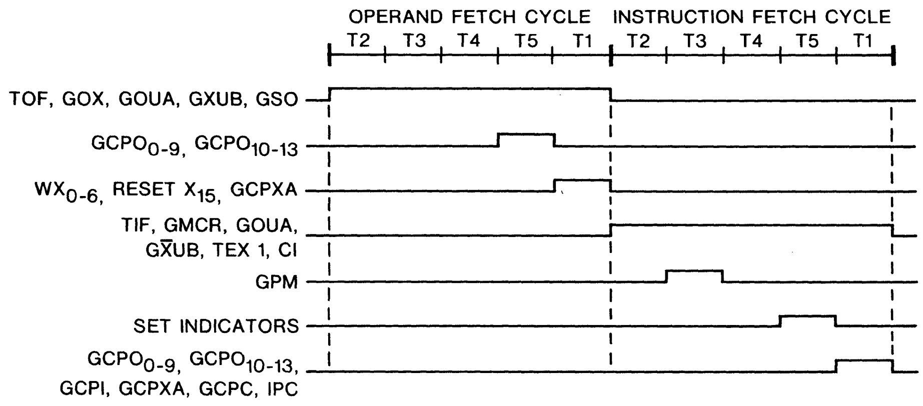


FIGURE II-4-22. ATX INSTRUCTION TIMING DIAGRAM

field and the contents of the even-address index register is used by the compare logic to set the indicators GT, EQ and LT.

A normal Instruction Fetch Cycle occurs during the second memory cycle of ATX.

4.5.9 DTC Instruction

Figures II-4-23 and II-4-24 are the flow and timing diagrams for the DTC instruction. The DTC instruction replaces the contents of the accumulator and extension registers with the two's complement of the original contents. The accumulator and extension registers together represent a 48-bit operand. The accumulator is the more significant half.

The double two's complement is formed by first complementing E, storing the carry condition out of E23 in the carry flip-flop CY and then by complementing A using CY as the carry in (CI). DTC uses two memory cycles for execution. During the first memory cycle with TOF set, the signal GEFUA gates the one's complement of E to the A input of the AU. The signal CI is energized entering a carry into the least significant bit position of the AU. The two's complement of E is formed on the sum output of the AU and is gated to E with the signal GSE. At T5 of this memory cycle, the AU output is clocked into E with the GCPE signal, and, the state of the carry out of the AU is entered into CY.

During the second memory cycle, the normal instruction fetch operation is overlapped by the complementing of the contents of A. With TIF set, the signal GAFUA gates the one's complement of A to A-input of the AU. The state of CY is gated through CI into the least significant bit position of the AU. The two's complement of the more significant half of the double word is formed on the sum output of the AU and is gated to A with GSA. At T5 of this memory cycle, the output of the AU is clocked into A with the signal GCPA.

4.5.10 BSM Instruction

Figures II-4-25 and II-4-26 are the flow and timing diagrams for the BSM instruction. The BSM instruction stores the contents of the program counter (PC) and the states of the indicators OV, GT, EQ, LT and BE at the memory location specified by the operand address, and then loads the operand address, plus 1, into the PC. The subsequent instruction fetch occurs at the new location held by the program counter.

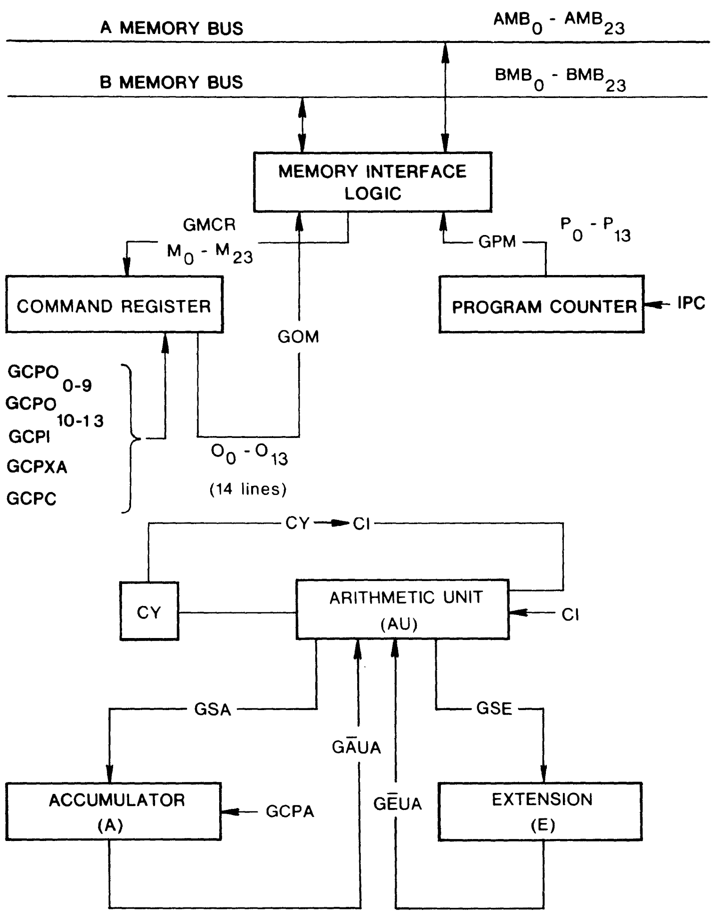


FIGURE II-4-23. DTC INSTRUCTION FLOW DIAGRAM

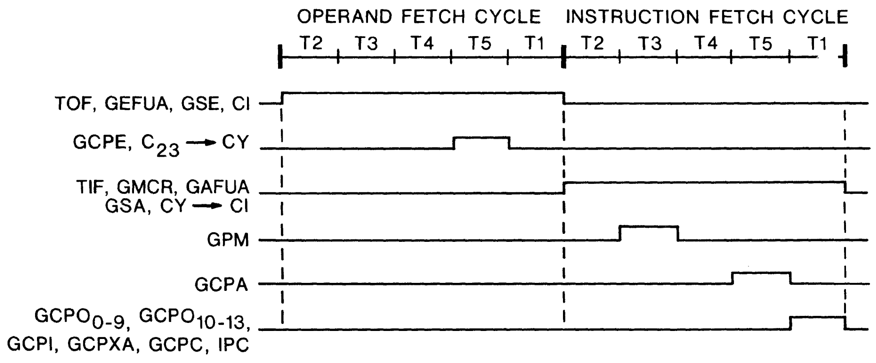


FIGURE II-4-24. DTC INSTRUCTION TIMING DIAGRAM

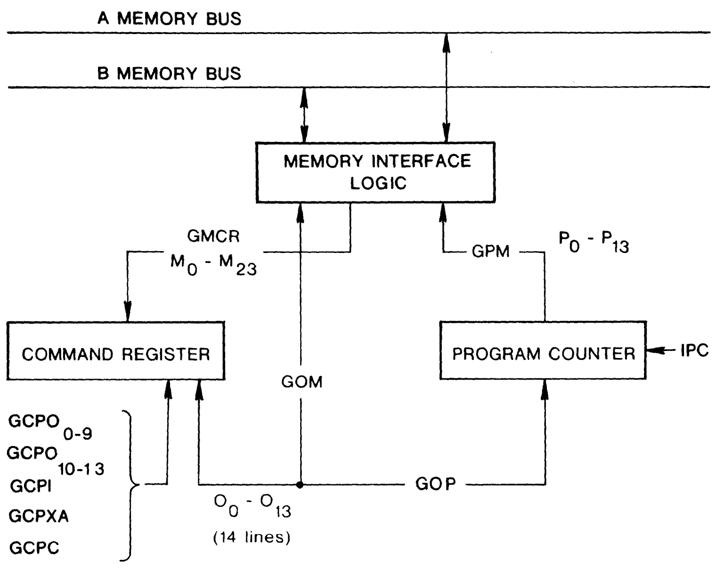


FIGURE II-4-25. BSM INSTRUCTION FLOW DIAGRAM

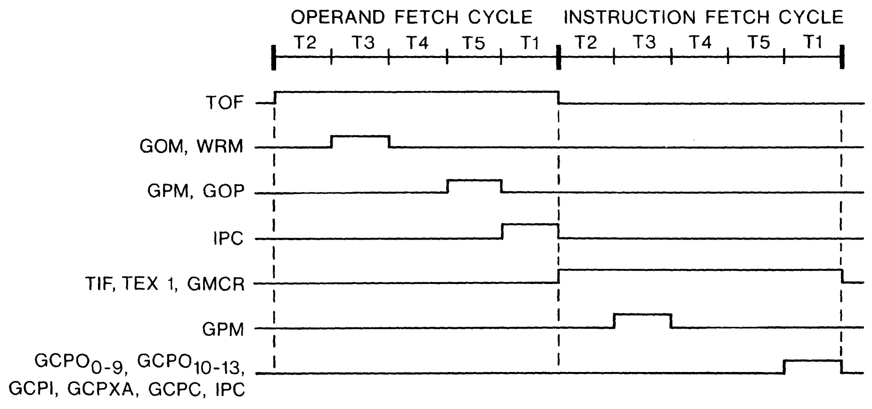


FIGURE II-4-26. BSM INSTRUCTION TIMING DIAGRAM

At T3 of the first memory cycle, the command register address field is transferred to memory by GOM (Gate Operand Address to Memory) and the signal WRM (Write to Memory) is energized. At T5, the contents of the PC and the indicators OV, GT, EQ, LT, and BE are gated to memory with GPM (Gate Program Counter to Memory). Simultaneously, the contents of the command register address field are gated to the program counter with the signal GOP (Gate Operand Address to Program Counter). Since program control is to be transferred to the operand address plus 1, the new contents of the PC are incremented at T1 of the TOF cycle with the signal IPC. The second half of the BSM instruction is a normal Instruction Fetch Cycle.

4.5.11 BSZ Instruction

Figures II-4-27 and II-4-28 are the flow and timing diagrams for the BSZ instruction. The BSZ instruction stores the contents of the PC and the state of the indicators GT, EQ, LT, and BE in memory location zero and transfers program control to the memory location specified by the operand address. BSZ differs from BSM only in the address transferred to memory and the incrementing of the program counter.

At T3, no address is transferred to memory. Since WRM (Write to Memory) is energized at T3, a write memory cycle is initiated for memory location zero. GOP (Gate Operand Address to Program Counter) stores the contents of the command register in P with the clock of T5. The Instruction Fetch Cycle obtains the next instruction from th memory address in the pc, the original BSZ operand address.

4.5.12 STA Instruction

Figures II-4-29 and II-4-30 are the flow and timing diagrams for the STA instruction. The Store A instruction stores the accumulator contents in the memory location specified by the STA's operand address. The data path from A to memory is through the adder unit into the Buffer (B) register. From B, the data is transferred to memory.

At T3 of the first memory cycle, the operand address is gated to memory by the signal GOM. WRM causes this to be "Write to Memory" memory cycle. Signal GAUA is also present causing the contents of A to be transferred to the sum output of the adder unit. The signal GSB (Gate Sum to B) gates the sum from the accumulator to the input of the buffer register. At T4, GCPB (Gate Clock Pulse to B) looses B. At T5, the new contents of B

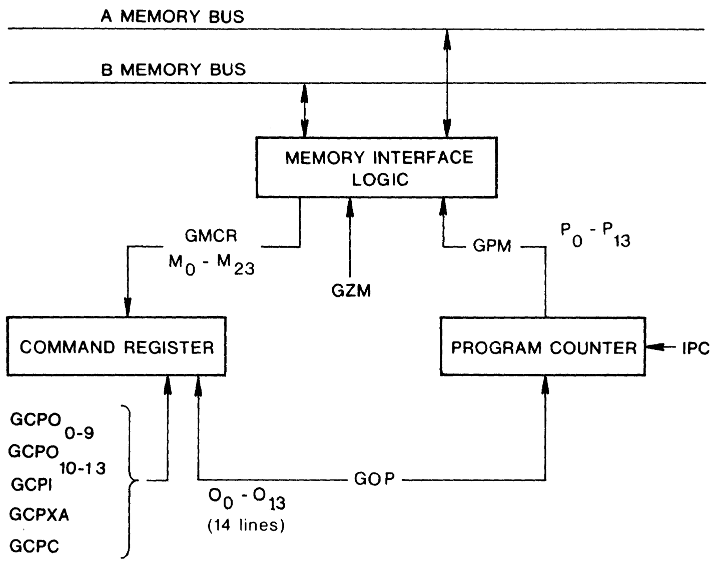


FIGURE II-4-27. BSZ INSTRUCTION FLOW DIAGRAM

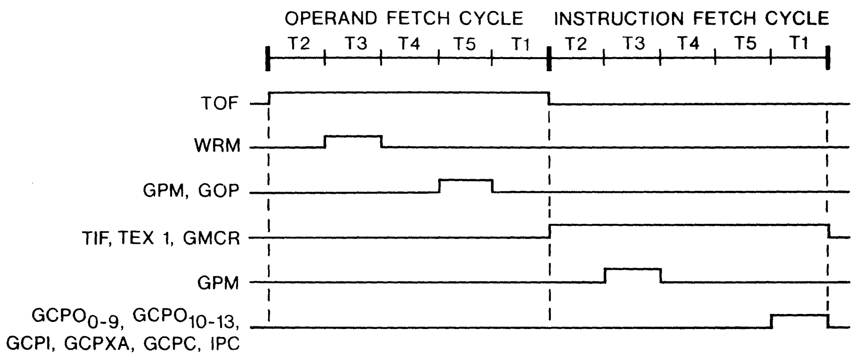


FIGURE II-4-28. BSZ INSTRUCTION TIMING DIAGRAM

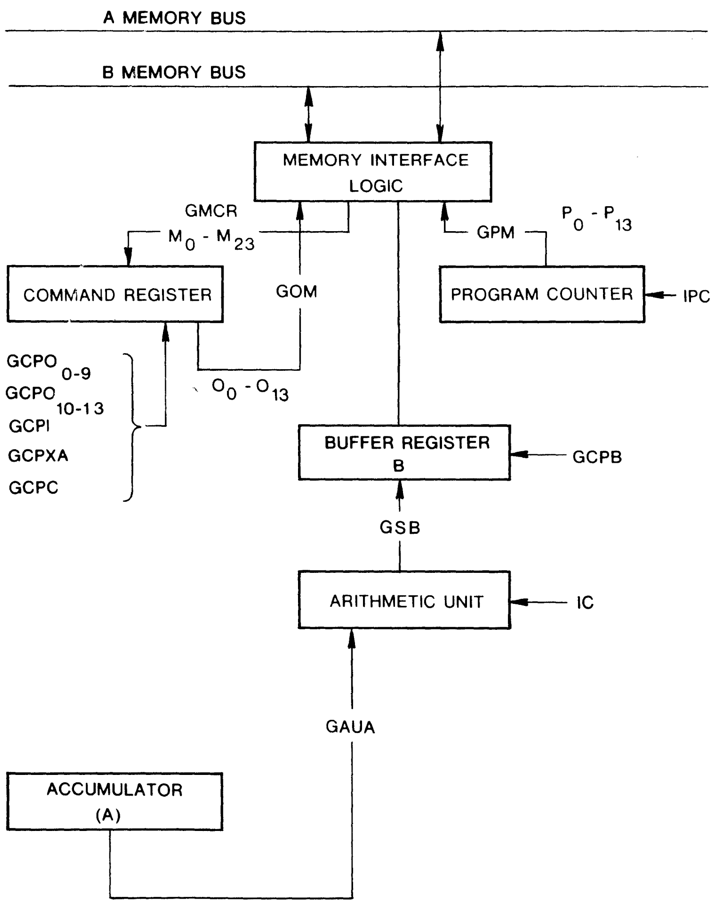


FIGURE II-4-29. STA INSTRUCTION FLOW DIAGRAM

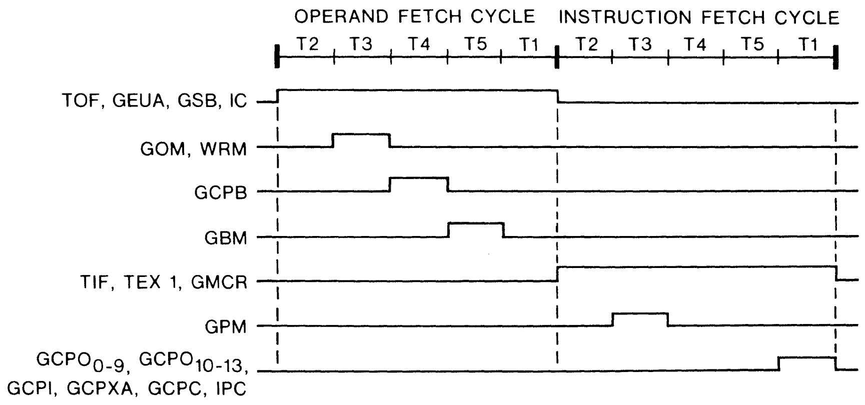


FIGURE II-4-30. STA INSTRUCTION TIMING DIAGRAM

are gated to memory. The second memory cycle of STA is a normal instruction fetch cycle.

4.5.13 STE Instruction

Figures II-4-31 and II-4-32 are the flow and timing diagrams for the STE instruction. The gating paths for STE differ from STA in that GEUA is energized during the first memory cycle rather than GAUA. Otherwise, the timing is the same.

4.5.14 STX Instruction

Figures II-4-33 and II-4-34 are the flow and timing diagrams for the STX instruction. The Store Index Register instruction stores the addressed index register contents in the memory location specified by the operand address. The instruction requires two memory cycles, one for storing the index register and one for the Instruction Fetch Cycle. At T3 of the first memory cycle the operand address is gated to memory with the signal GOM. The WRM signal at T3 causes a write-to-memory cycle, during which GXUB presents the contents of the addressed index register to the B-input of the adder. The adder output (Index register contents) is gated to the command register with the signal GSO. At T4 of the Instruction Fetch Cycle, the two signals GCP0-9 and GCP10-13 clock the contents of the index register into the Command Register bits 0 through 13. Signal GOM presents the new contents of the CR to memory at T5. The second memory cycle of STX is a normal instruction fetch cycle.

4.5.15 TCA Instruction

Figures II-4-35 and II-4-36 are the flow and timing diagrams for the TCA instruction. TCA is a single memory cycle instruction. The two's complementation of A occurs simultaneously with the instruction fetch cycle. The two's complement of A is formed by adding one to the one's complement of A. This is accomplished by energizing GAFUA which gates the one's complement of A to the adder. A carry is inserted into the least significant bit position of the adder (CI) resulting in one being added to the complemented A-input. By T5 of the TCA execution, the worst case carry has had time to propagate through the adder. GCPA is energized at T5 clocking the sum output of the adder (two's complement of A) into A.

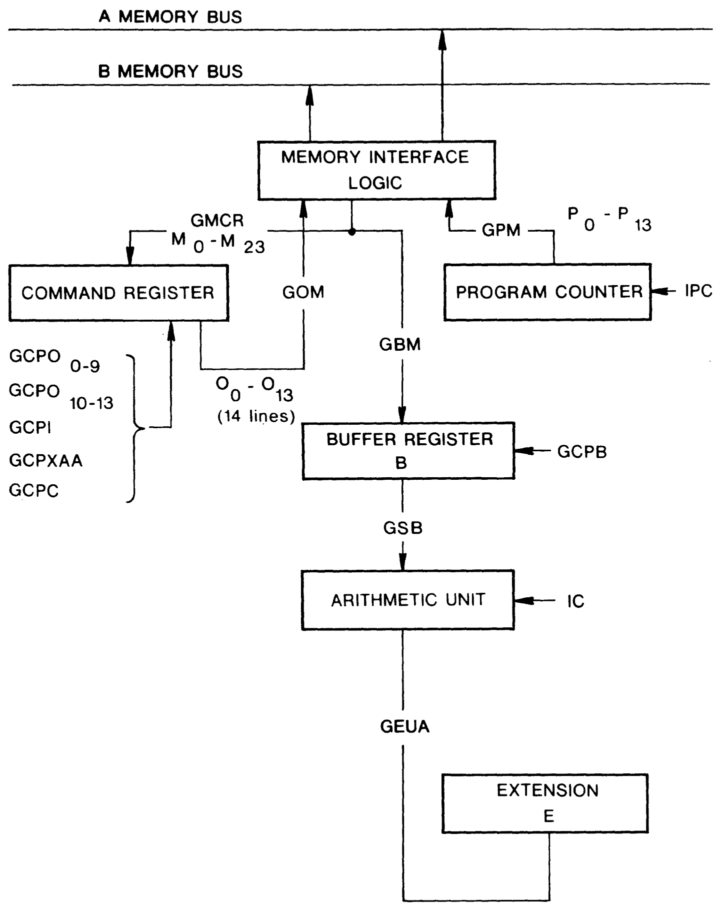


FIGURE II-4-31. STE INSTRUCTION FLOW DIAGRAM

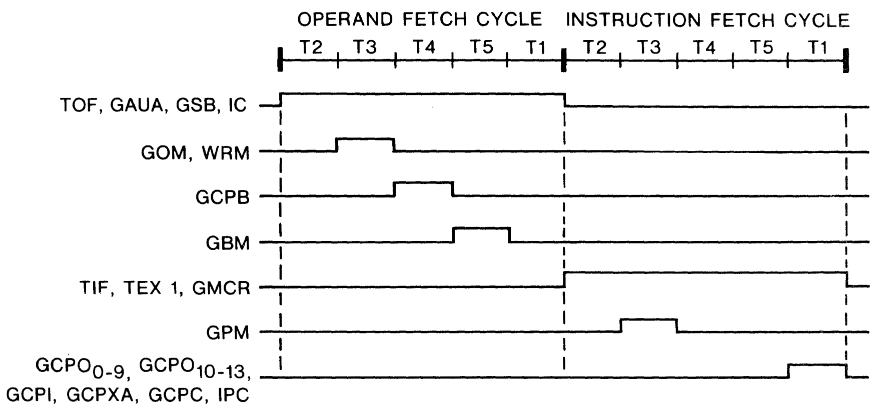


FIGURE II-4-32. STE INSTRUCTION TIMING DIAGRAM

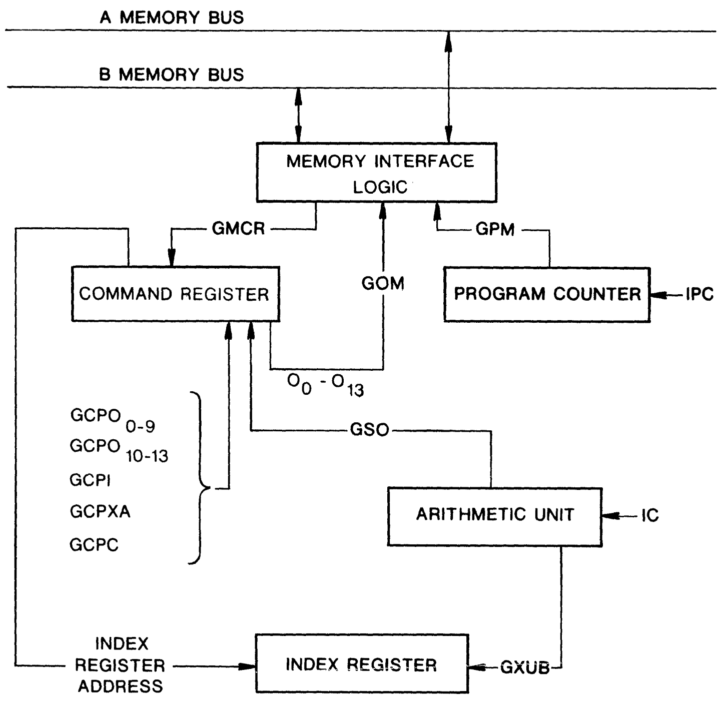


FIGURE II-4-33. STX INSTRUCTION FLOW DIAGRAM

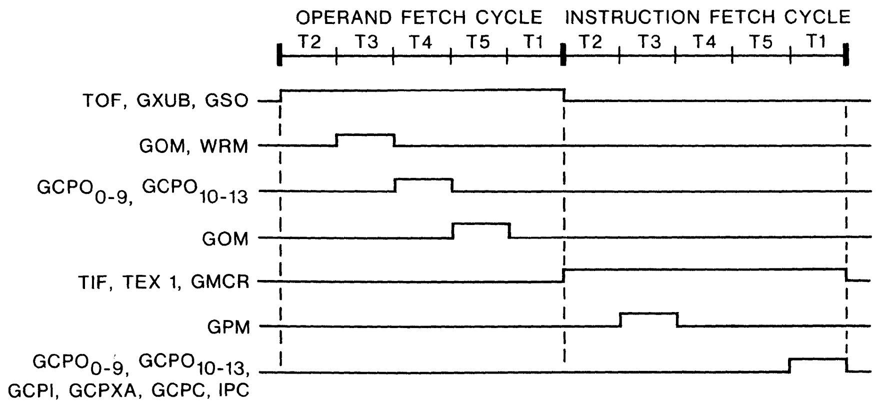

OPERAND FETCH CYCLE
INSTRUCTION FETCH CYCLE
T2 T3 T4 T5 T1 T2 T3 T4 T5 T1
TOF, GXUB, GSO
GOM, WRM
$GCPO_{0-9}$, $GCPO_{10-13}$
GOM
TIF, TEX 1, GMCR
GPM
$GCPO_{0-9}$, $GCPO_{10-13}$, GCPI, GCPXA, GCPC, IPC

FIGURE II-4-34. STX INSTRUCTION TIMING DIAGRAM

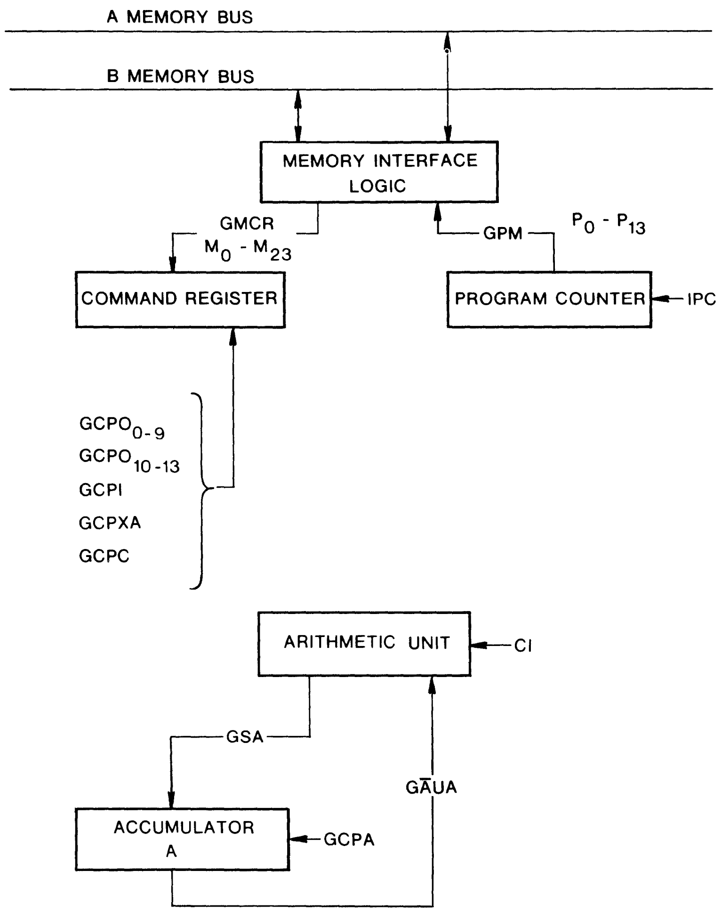


FIGURE II-4-35. TCA INSTRUCTION FLOW DIAGRAM

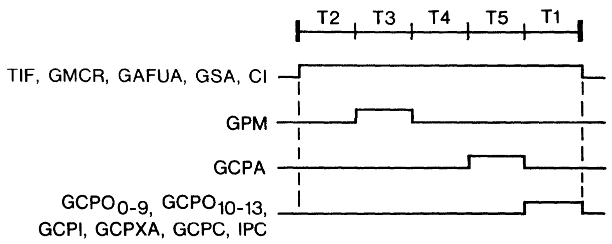


FIGURE II-4-36. TCA INSTRUCTION TIMING DIAGRAM

4.5.16 LDX Instruction

Figures II-4-37 and II-4-38 are the flow and timing diagrams for the LDX instruction. The Load Index register instruction loads the addressed index register with the contents of bits 0 through 13 of the Command Register. The LDX instruction is a single memory cycle instruction. X17-X15 (index address) select the index register to be loaded. The signal GOX (Gate O to Index) gates the contents of the operand address field to the input of the index register. The signal WX effects the transfer from the command register to the index register.

4.5.17 LXA Instruction

Figures II-4-39 and II-4-40 are the flow and timing diagrams for the LXA instruction. The LXA instruction copies the least significant 14 bits of the accumulator into the addressed index register, and requires one memory cycle for execution. The A to X data transfer overlaps the next instruction fetch cycle. The decoded command signal LXA actually controls the gating of A to X. The contents of A0 through A13 are loaded into X at T5 with the write index signal WX.

4.5.18 RSR Instruction

Figures II-4-41 and II-4-42 are the flow and timing diagrams for the RSR instruction. The RSR instruction loads the switch register (W) contents into the accumulator. The output of the switch register is gated to A with the signal GWA (Gate W to A). The transfer into A occurs with the clock of T5 gated to A with the signal GCPA. Loading of the accumulator overlaps the next instruction fetch.

4.5.19 EXC Instruction

Figures II-4-43 and II-4-44 are the flow and timing diagrams for the EXC instruction. The Exchange A and E instruction exchanges the contents of the A and E register. EXC requires only one memory cycle for execution, and the exchange of the contents of A and E overlap the next instruction fetch cycle. The transfer of A to E is a direct transfer. A is Gated to E (GAE) with the signal EXC, the decoded exchange command. The contents of E are transferred to A via the AU. E is gated to AU with GEUA. The output of the Adder is gated to A with GSA. At T5 of EXC, GCPA and GCPE are energized gating a clock pulse to A and E and causing the transfer of the contents of A to E, and the contents of E to A.

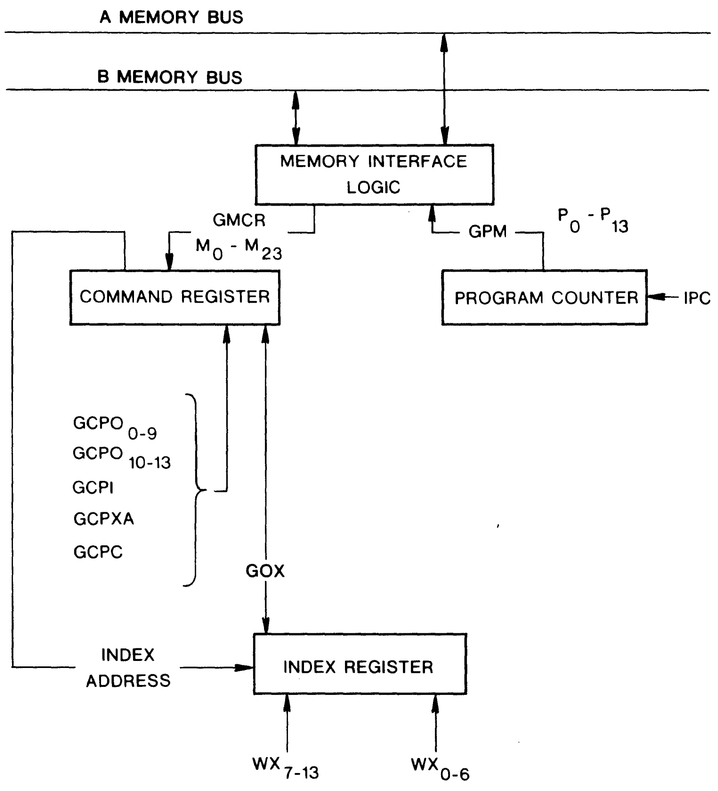


FIGURE II-4-37. LDX INSTRUCTION FLOW DIAGRAM

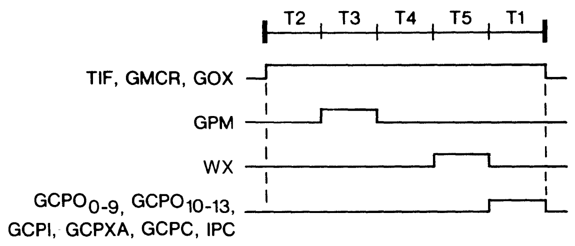


FIGURE II-4-38. LDX INSTRUCTION TIMING DIAGRAM

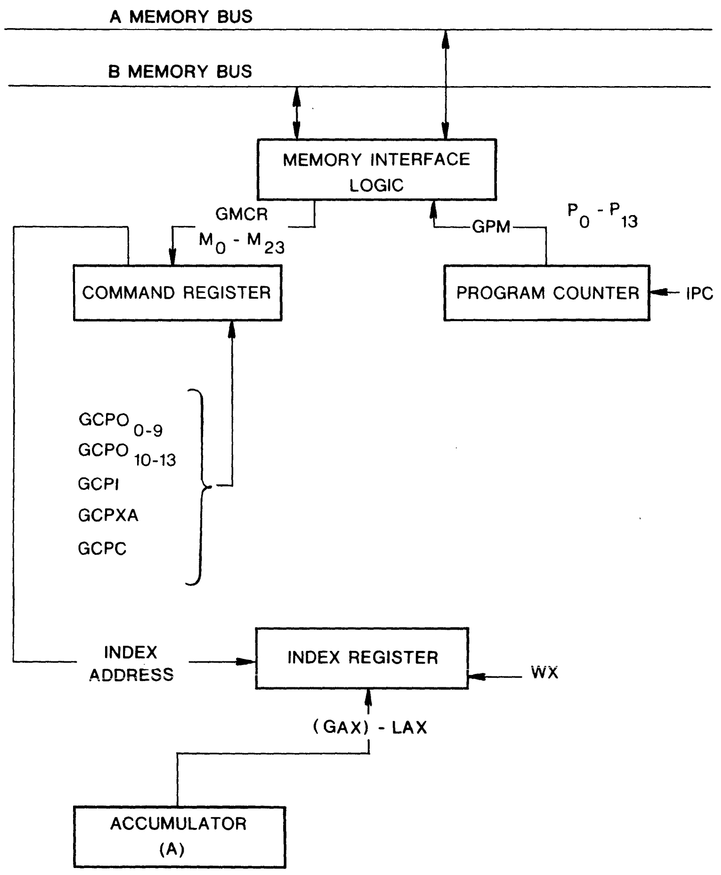


FIGURE II-4-39. LXA INSTRUCTION FLOW DIAGRAM

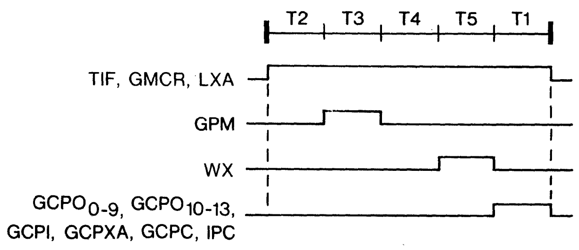


FIGURE II-4-40. LXA INSTRUCTION TIMING DIAGRAM

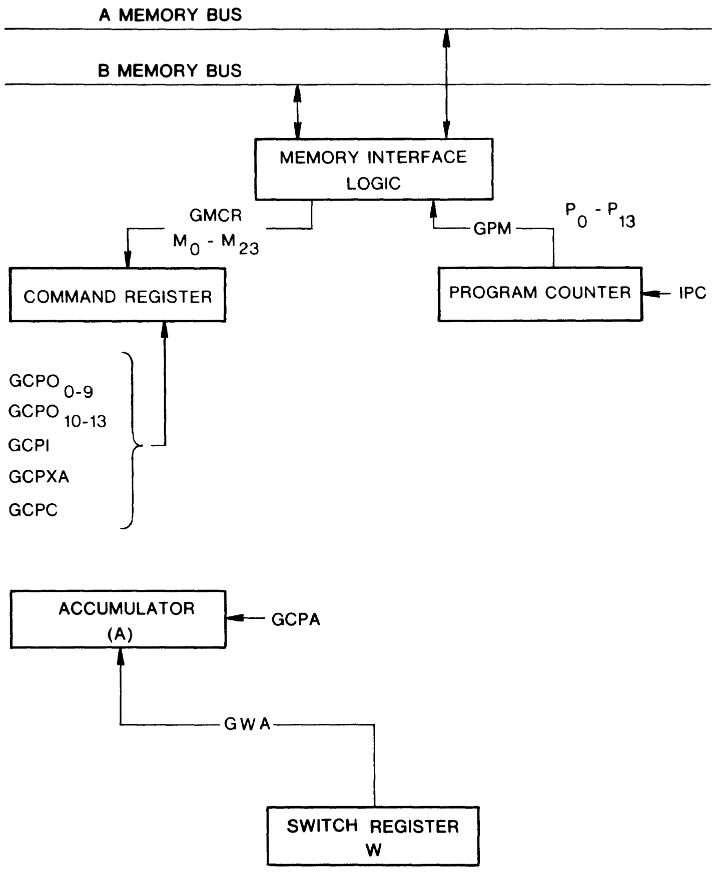


FIGURE II-4-41. RSR INSTRUCTION FLOW DIAGRAM

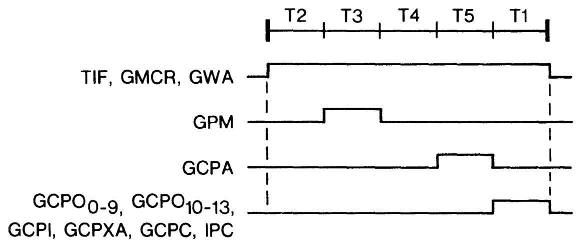


FIGURE II-4-42. RSR INSTRUCTION TIMING DIAGRAM

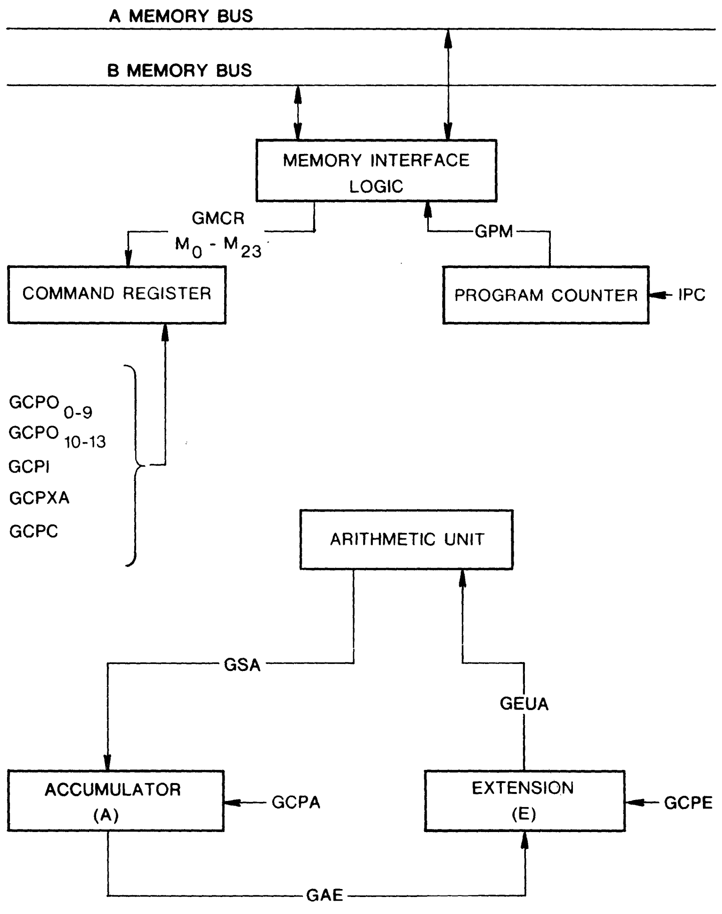


FIGURE II-4-43. EXC INSTRUCTION FLOW DIAGRAM

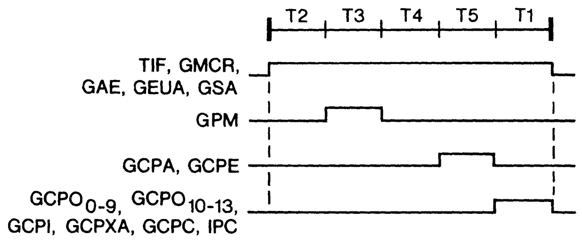


FIGURE II-4-44. EXC INSTRUCTION TIMING DIAGRAM

4.5.20 RST Instruction

Figures II-4-45 and II-4-46 are the flow and timing diagrams for the RST instruction. The RST instruction is used for controlling the resetting of the program switches, and the IE (Interrupt Enable) and OV (OVerflow) flip-flops. The RST instruction requires one memory cycle for execution. At T5 of RST, the outputs of the operand address register, bits 000 through 09, are gated to the switch register flip-flops, SW0 through SW7, and the Interrupt Enable and the Overflow flip-flops, as illustrated in Figure II-3-45. If 000 is a 1 during RST, SW0 is reset with the clock at T5. If 001 is a 1, SW1 is reset with the clock at T5, etc. The next instruction fetch occurs concurrently.

4.5.21 SST Instruction

Figures II-4-47 and II-4-48 are the flow and timing diagrams for the SST instruction. Instruction SST is used for controlling the setting of the program switch register (SW), and the IE (Interrupt Enable) and OV (OVerflow) flip-flops. At time T5, 00 is gated to the set input of SW0, 01 is gated to the set input of SW1 etc., with the exception of the IE flip-flop. 08 is gated to IE at T2 time instead of T5 as illustrated in figure II-3-47. If 00 is a 1 at T5 of SST, SW1 is set with the clock at T5. If 0 is a 1 at T5 of SST, SW1 is set with the clock at T5. If 0 is a 1, SW is set with the clock at T5 and so on up to IE. IE is set by the clock at T2 if 08 is a 1.

4.5.22 SPU Instruction

Figures II-4-49 and II-4-50 are the flow and timing diagrams for the SPU instruction. The SPU (Select Peripheral Unit) instruction presents the least significant 18 bits of the SPU instruction word to the system peripherals via the accumulator bus during the time the Peripheral Select (PS) synchronization signal is energized. If the SPU instruction specifies an input or output accumulator transfer, the transfer occurs at T1.

The SPU instruction requires one memory cycle for execution. At T4, the least significant 18 bits of the command register are gated to the accumulator bus with the signal GCRN (Gate Command Register to the accumulator bus N), and the PS signal is energized. These 18 bits define direction of information transfer, if any, specify action to be taken (command), and define which peripheral device is to respond.

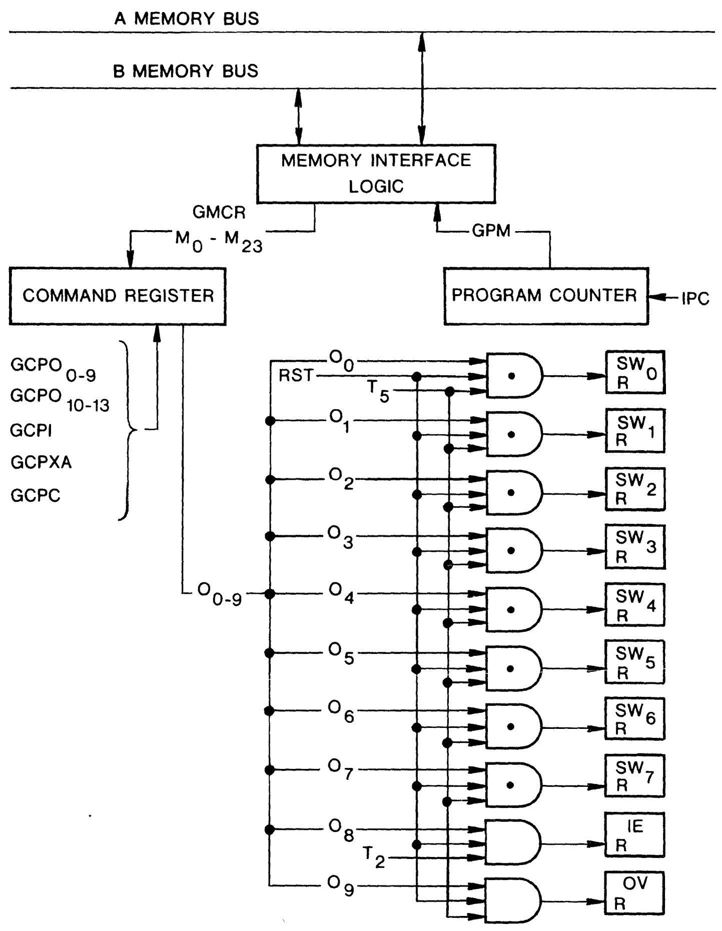


FIGURE II-4-45. RST INSTRUCTION FLOW DIAGRAM

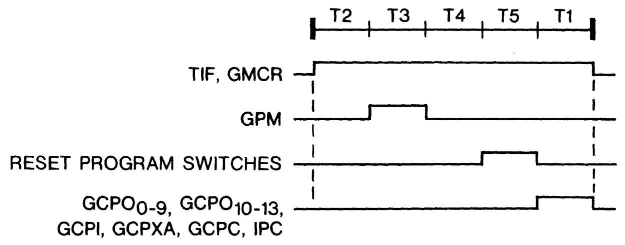


FIGURE II-4-46. RST INSTRUCTION TIMING DIAGRAM

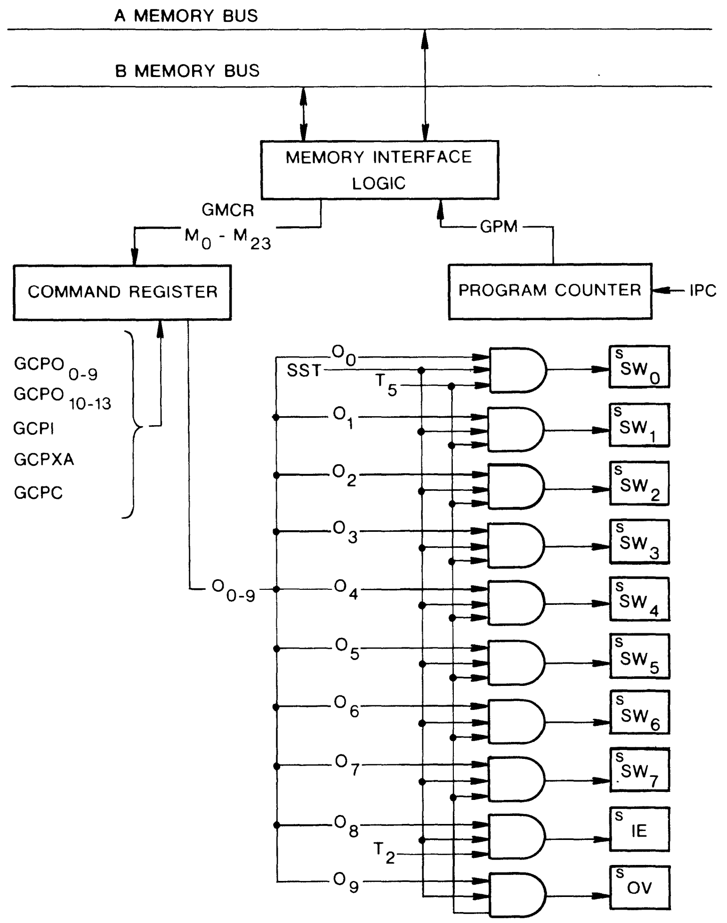


FIGURE II-4-47. SST INSTRUCTION FLOW DIAGRAM

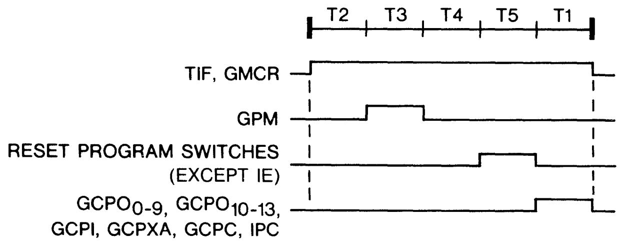

T2
T3
T4
T5
T1
TIF, GMCR
GPM
RESET PROGRAM SWITCHES (EXCEPT IE)
$GCPO_{0-9}$, $GCPO_{10-13}$, GCPI, GCPXA, GCPC, IPC

FIGURE II-4-48. SST INSTRUCTION TIMING DIAGRAM

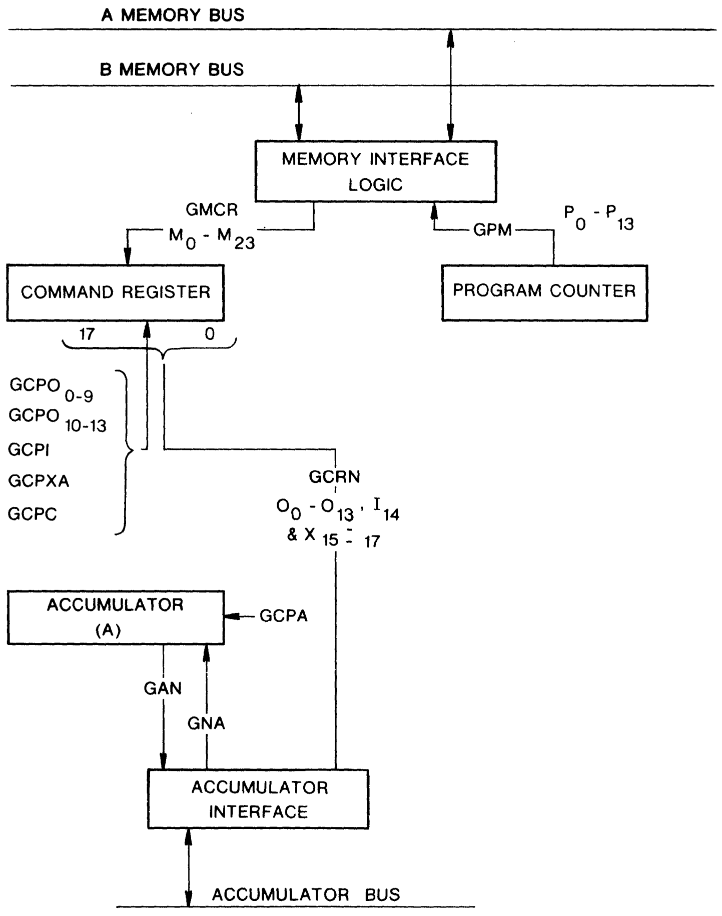


FIGURE II-4-49. SPU INSTRUCTION FLOW DIAGRAM

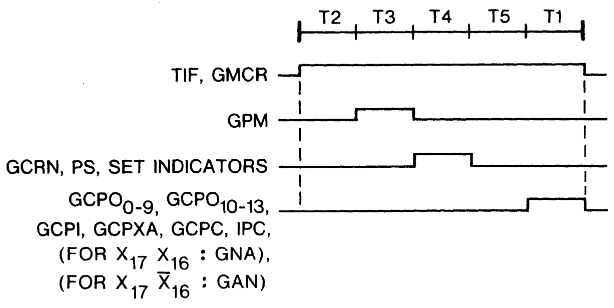


FIGURE II-4-50. SPU INSTRUCTION TIMING DIAGRAM

The addressed peripheral gates its status to the CPU on the 4 most significant bit lines of the accumulator bus. During T4, the peripheral status from the accumulator bus is entered into the GT, EQ, LT, and BE indicators. After the SPU execution, these indicators take on the following meaning:

Bit 23	GT = Peripheral IDLE
Bit 22	EQ = Peripheral IDLE with an ERROR
Bit 21	LT = Peripheral BUSY
Bit 20	BE = Peripheral NOT AVAILABLE

If bit 17 is a "1," there will be an information transfer at the following T1 time. The direction will be from the peripheral device if bit 16 is a "1" or to the peripheral device if bit 16 is a "0." In the latter case, the signal GAN (Gate Accumulator to N) will occur, while in the former case the signal GNA (Gate N to A) will occur. The accumulator is loaded by GCPA. The next instruction fetch overlaps the execution of the SPU instruction.

4.5.23 BRU Instruction

Figures II-4-51 and II-4-52 are the flow and timing diagrams for the BRU instruction. An unconditional branch instruction transfers the branch instruction's operand address to memory during the instruction fetch cycle instead of the program counter contents. The operand address is then transferred to the program counter and the program counter incremented by one. This results in fetching the next instruction from the memory location specified by the branch operand address, and setting the program counter so it points to the instruction in the next memory location. BRU is executed as a normal instruction fetch cycle, except that GOM instead of GPM is energized at T3. At T4 of BRU, the contents of the operand address field are gated to P with the signal GOP (Gate O to PC). If a BRU is executed indirectly, the indicators OV, GT, EQ, LT, and BE are set according to bits 23, 22, 21, 20, and 19, respectively, at the indirect word. (Reference the indirect address transfer timing description.)

4.5.24 BAH Instruction

Figures II-4-53 and II-4-54 are the flow and timing diagrams for the BAH instruction. BAH is executed with the same set of control signals as BRU, the difference in the two commands being that HLT is set at T1 of BAH, stopping the clocks to the CPU. The START pushbutton on the CPU control panel must be pressed to resume the program. An indirect BAH does not restore the GT, EQ, LT and BE indicators.

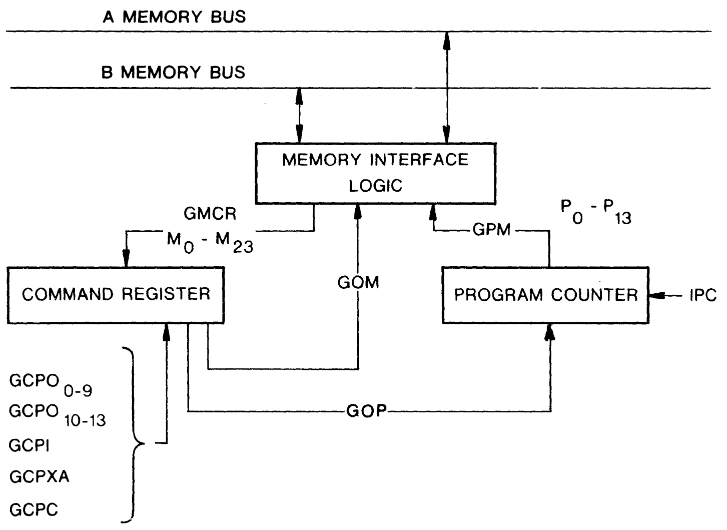


FIGURE II-4-51. BRU INSTRUCTION FLOW DIAGRAM

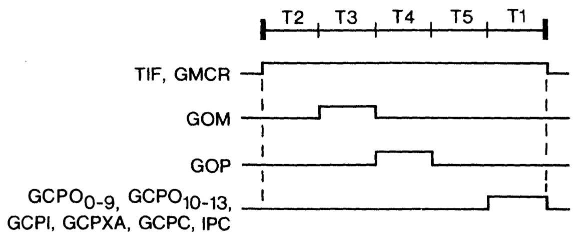


FIGURE II-4-52. BRU INSTRUCTION TIMING DIAGRAM

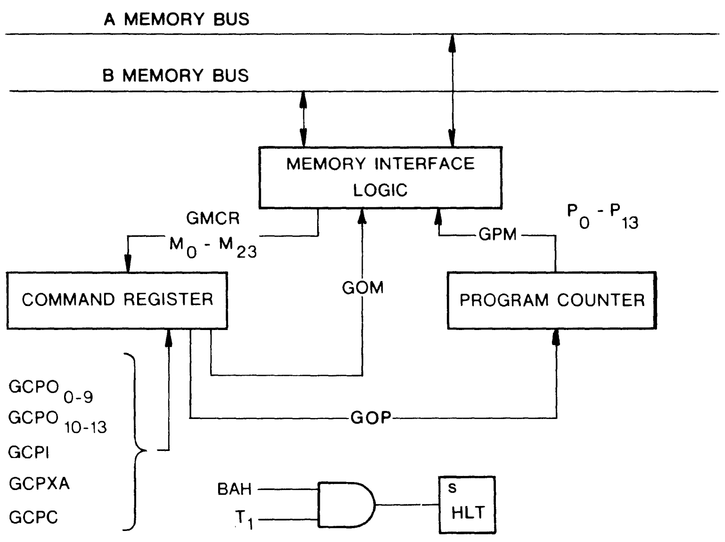


FIGURE II-4-53. BAH INSTRUCTION FLOW DIAGRAM

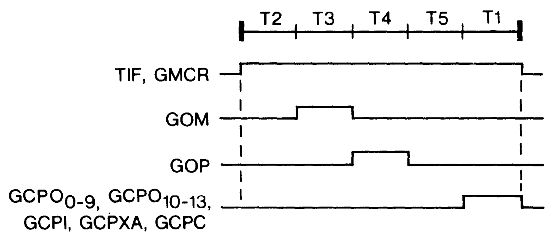


FIGURE II-4-54. BAH INSTRUCTION TIMING DIAGRAM

4.5.25 BAT Instruction

Figures II-4-55 and II-4-56 are the flow and timing diagrams for the BAT instruction. The BAT instruction will cause a branch if a particular state of the A register exists. The state to be tested by BAT is defined by bits 14-17. The states of the A register tested are: positive, zero, negative, and odd. The signal which is generated to determine whether or not the branch will be executed is EB (Execute Branch). If EB is true GOM occurs at T3, causing a branch to occur as described in the explanation of BRU. If EB is false, GPM occurs at T3 and an instruction fetch cycle transpires. During BAT, signals GAUA and IC are energized gating the contents of A through the AU. The signal ZERO is energized when the output of the AU is equal to zero.

EB is true if:

1) A is positive and not zero and X17 is a one.

2) A is zero and X16 is a one.

3) A is negative and X15 is a one.

4) A is an odd binary number and I14 is a one.

4.5.26 BOS Instruction

Figures II-4-57 and II-4-58 are the flow and timing diagrams for the BOS instruction. The BOS instruction is a conditional branch depending on the state of any one of eight program switches (SW0 through SW7), six console switches (CS0 through CS5), or the two indicators IE and OV. The switch or indicator to be tested is specified by bits 15-17 of the BOS instruction word. These four bits are decoded into their sixteen possible states to form the signals ST0, through ST15 which are used in the EB (Enable Branch) logic to select the branch conditions.

Signal GOM occurs at T3 and GOP at T4 if EB = 1 during BOS. This results in a program branch. If EB = 0, a normal instruction fetch cycle is executed using the Program Counter. A BOS instruction which specifies the OV (overflow) indicator will clear OV after executing the branch.

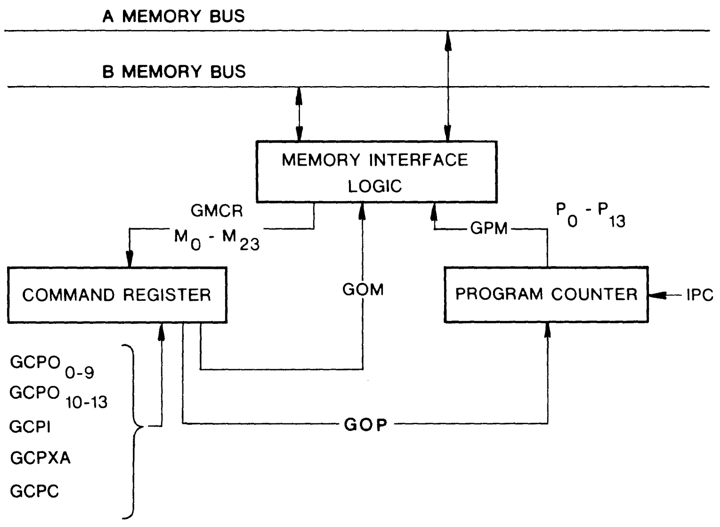


FIGURE II-4-55. BAT INSTRUCTION FLOW DIAGRAM

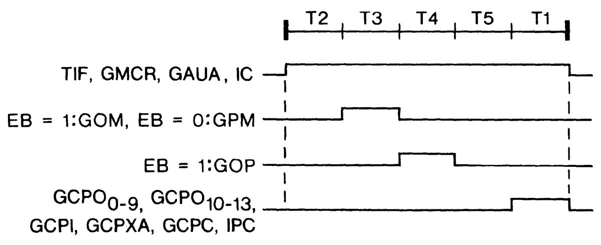


FIGURE II-4-56. BAT INSTRUCTION TIMING DIAGRAM

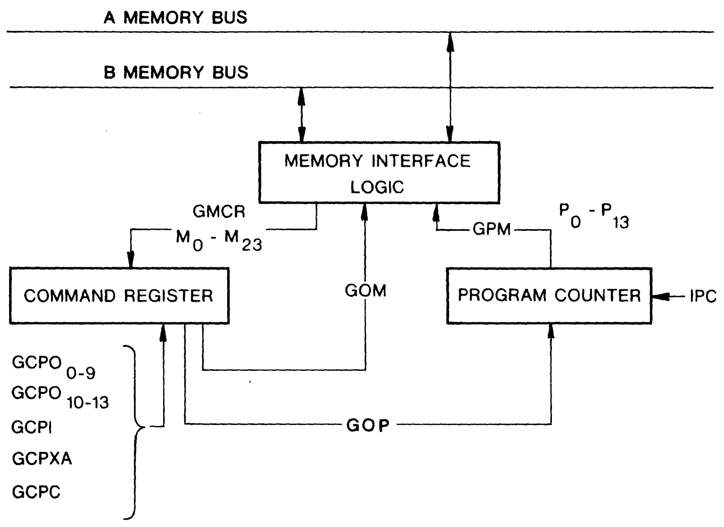


FIGURE II-4-57. BOS INSTRUCTION FLOW DIAGRAM

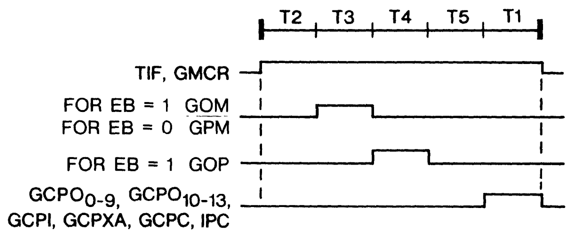


FIGURE II-4-58. BOS INSTRUCTION TIMING DIAGRAM

4.5.27 BOI Instruction

Figures II-4-59 and II-4-60 are the flow and timing diagrams for the BOI instruction. The BOI instruction initiates a program branch dependent upon the states of GT, EQ, LT and BE. The indicators to be tested by BOI are specified by bits 15 thru 17 of the BOI instruction. X17 specifies GT, X16 specifies EQ, X15 specifies LT, and I14 specifies BE. If one or more of the specified indicators is true, then EB (Enable Branch) is true which allows GOM (Gate O to Memory) at T3 and GOP (Gate O to P counter) at T4. Otherwise, a normal instruction fetch cycle ensues.

4.5.28 SHIFT Instructions

Figures II-4-61 and II-4-62 are the flow and timing diagrams for the shift instruction. The FST-1 shift instructions shift the A and E registers independently or together either right, left, or around. A separate operation code specifies each of the possible shift operations.

One special instruction, DSN (Double Shift Normalize) is available for normalizing, i.e., the contents of the A and E registers. During DSN A and E are shifted left until the shift count goes to zero, or until the value in A and E is normalized. The residual of the count after shifting is completed and stored in index register zero.

There are nine (9) different shift instructions available to the FST-1 programmer. The timing for all of these instructions is basically the same. Each shift instruction starts with a pseudo operand fetch cycle. At T2 of the TOF memory cycle, the shift count in the least significant six bits of the operand register is transferred to the shift counter (CO) with GOCO (Gate O to Counter). The TV (Timing for Variable length shift) flip-flop is also set at T2 if the shift count is not zero.

A different combination of the shift gates is enabled for each instruction as illustrated in Figure II-4-63. If the shift count is zero, TV is not set and no shifting occurs. The shift count is decremented in CO with the signal DCO (Decrement CO) with the clock of each phase time as long as TV is set. When the shift counter is decremented to a count of 1, the signal CO = 1 is energized causing TV to be reset. TV reset ends the shift operation.

At the end of the first memory cycle, TOF is reset. The next instruction fetch cycle is not initiated until the shift counter contains a value of less than six (6). At T1 of the first memory

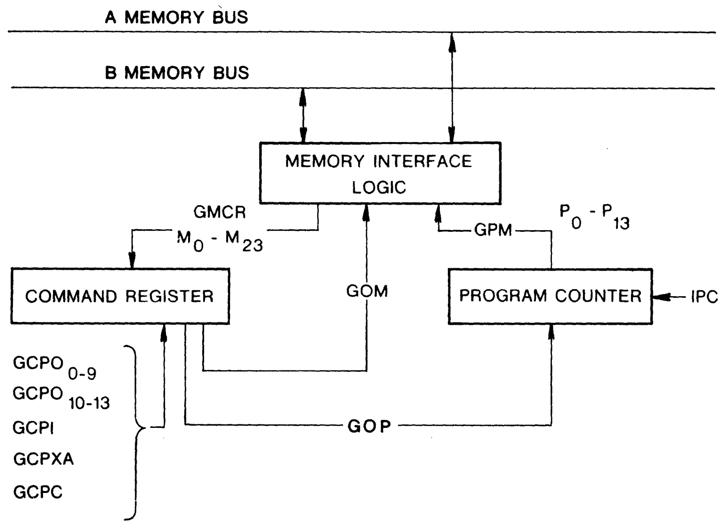


FIGURE II-4-59. BOI INSTRUCTION FLOW DIAGRAM

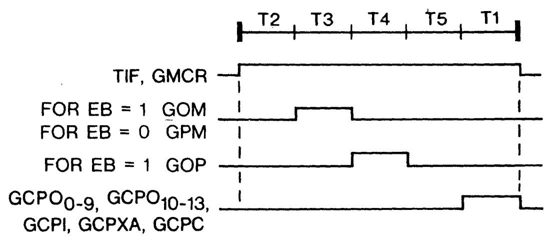


FIGURE II-4-60. BOI INSTRUCTION TIMING DIAGRAM

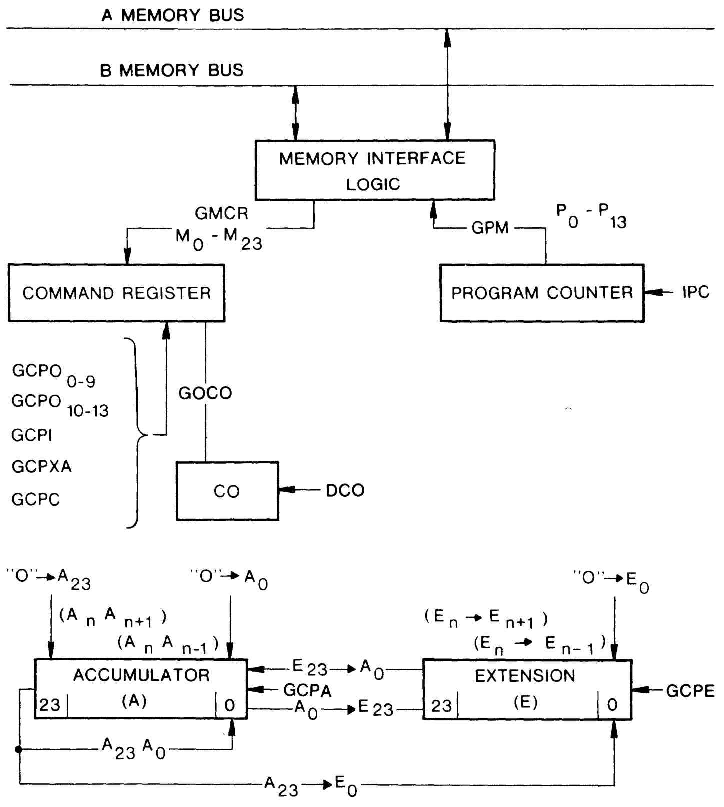


FIGURE II-4-61. SHIFT INSTRUCTION FLOW DIAGRAM

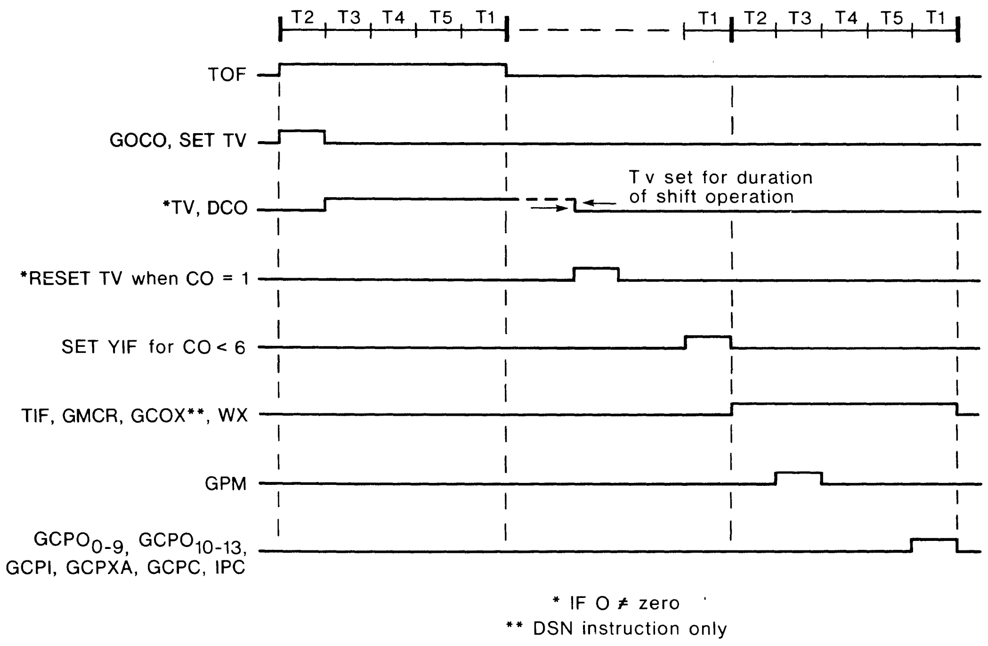
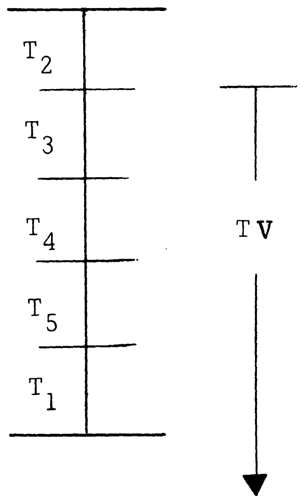


FIGURE II-4-62. SHIFT INSTRUCTION TIMING DIAGRAM



SHIFT RIGHT	LOGICAL SHIFT	SHIFT AROUND	SHIFT LEFT	DOUBLE SHIFT RIGHT	LOGICAL DOUBLE SHIFT	DOUBLE SHIFT AROUND	DOUBLE SHIFT LEFT	DOUBLE SHIFT NORMAL
$A_{23} \rightarrow A_{23}$	"0" $\rightarrow A_{23}$	$A_{23} \rightarrow A_0$	"0" $\rightarrow A_0$	$A_{23} \rightarrow A_{23}$	"0" $\rightarrow A_{23}$	$A_{23} \rightarrow A_{23}$	$A_{22} \rightarrow A_{23}$	$A_{23} \rightarrow A_{23}$
$A_n \rightarrow A_{n-1}$	$A_n \rightarrow A_{n-1}$	$A_n \rightarrow A_{n+1}$	$A_n \rightarrow A_{n+1}$	$A_n \rightarrow A_{n-1}$	$A_n \rightarrow A_{n-1}$	$A_n \rightarrow A_{n+1}$	$A_n \rightarrow A_{n+1}$	$A_n \rightarrow A_{n+1}$
GCPA	GCPA	$A_{22} \rightarrow A_{23}$	$A_{22} \rightarrow A_{23}$	$A_0 \rightarrow E_{23}$	$A_0 \rightarrow E_{23}$	$E_{22} \rightarrow A_0$	$E_{23} \rightarrow A_0$	$E_{23} \rightarrow A_0$
		GCPA	GCPA	$E_n \rightarrow E_{n-1}$	$E_n \rightarrow E_{n-1}$	$E_n \rightarrow E_{n+1}$	$E_n \rightarrow E_{n+1}$	$E_n \rightarrow E_{n+1}$
				GCPA	GCPA	$A_{23} \rightarrow E_0$	"0" $\rightarrow E_0$	"0" $\rightarrow E_0$
				GCPE	GCPE	GCPA	GCPA	GCPA
						GCPE	GCPE	GCPE

FIGURE II-4-63. SHIFT INSTRUCTION GATING TABLE

FAIRCHILD SYSTEMS TECHNOLOGY

cycle of any shift where CO = 6, TIF is set and a normal instruction fetch cycle is initiated. By the time the instruction fetch cycle is completed, the shift counter will have run out completing the shift operation before the end of the instruction fetch cycle.

The decoded shift instructions SR, LS, etc. are used to control the gating signals between registers. TV is used to control the gating of clocks to the registers to be shifted.

The DSN instruction differs in that the shifting of the A and E registers is terminated either when the shift counter runs out or when A23 and A22. SN (Shift Normalize) must be energized for clocks to be gated to the A and E register.

SN is energized during DSN as long as A23 and A22 are not in the same state. SN is energized during all other instructions. The gating of clock, for DSN is controlled by an AND of TV and SN. The decoded signal DSN is used to gate the contents of CO to the input of the index registers (GCOX). WX (Write Index) is energized during the instruction fetch to write the remainder of the count in CO into index register zero.

4.5.29 AOM Instruction

Figures II-4-64 and II-4-65 are the flow and timing diagrams for the AOM instruction. The Add One to Memory instruction takes place over four memory cycles. The first cycle is an operand fetch, the second cycle is an operand modify, the third cycle is an operand store, and the fourth cycle is a normal instruction fetch.

The operand is fetched to the B register using GOM (Gate Operand to Memory) to send the operand address and GMB (Gate Memory to B) and GCPB (Gate Clock Pulse to B) to load the Buffer register with data at T1.

The data in B is modified by gating it to the arithmetic unit with GBUB (Gate B to Adder input B) and forcing a one into the arithmetic unit with CI (Carry Input). The resultant sum is returned to the B register by GSB (Gate Sum to B) where it replaces the original operand at T5 of the second memory cycle. The control flip-flop TEX1 (Time of Execution 1) identifies this phase of the execution. If a change of sign results from the addition of two like-signed operands, OV (Overflow) is set. TEX2 (Time of Execution 2) identifies the store phase of the AOM instruction. WRM (Write Memory) at time T3 causes a clear write

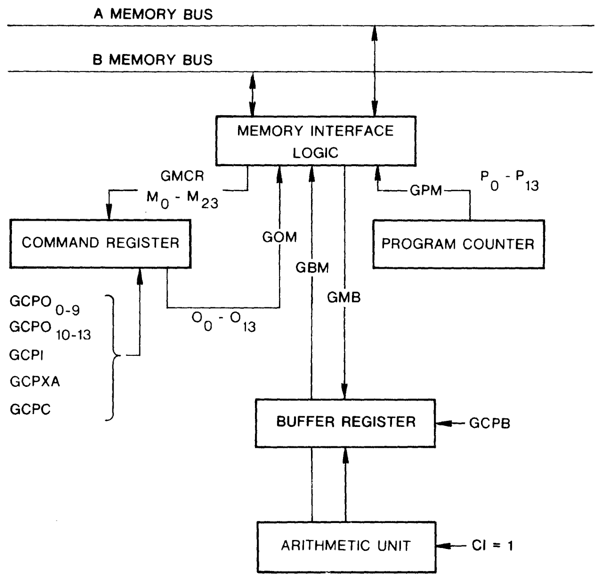


FIGURE II-4-64. AOM INSTRUCTION FLOW DIAGRAM

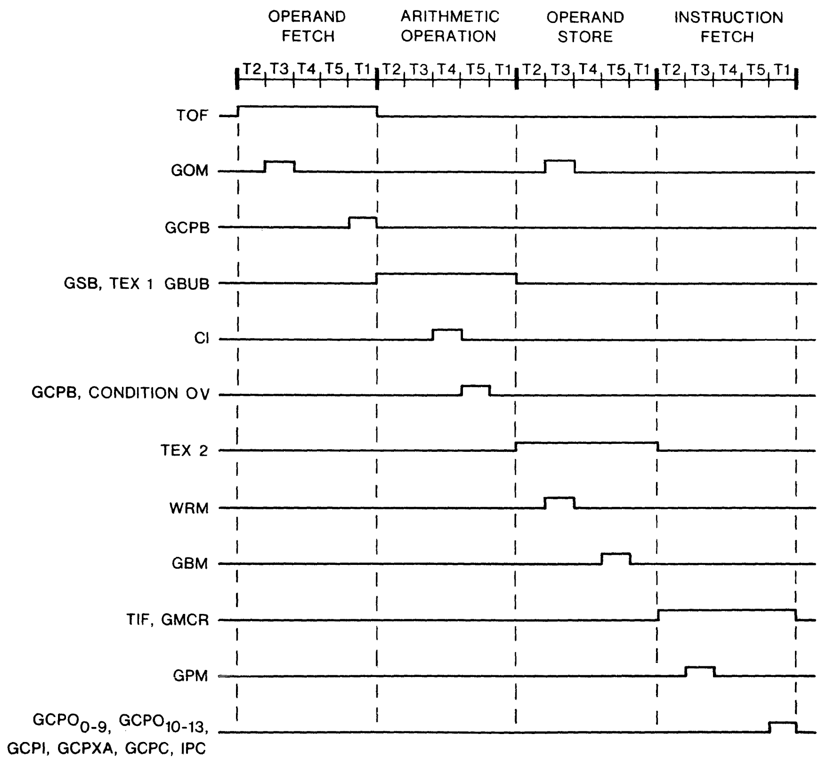


FIGURE II-4-65. AOM INSTRUCTION TIMING DIAGRAM

memory operation which accepts data gated by GBM (Gate B to Memory at T5). TIF identifies the fourth memory cycle of AOM as a normal instruction fetch.

4.5.30 SOM Instruction

Figures II-4-66 and II-4-67 are the flow and timing diagrams for the SOM instruction. Subtract One from Memory differs from AOM only in the modification memory cycle. The signal GMI (Gate Minus 1 to Adder input A) is energized rather than CI. This signal injects ones into all 24 stages of the adder. This value is minus one in two's complement arithmetic. The output of the adder is the operand value reduced by one. If a change of sign results from two like-signed operands OV (Overflow) is set. The remainder of SOM is identical with AOM.

4.5.31 DADD Instruction

Figures II-4-68 and II-4-69 are the flow and timing diagrams for the DADD instruction. Double Add is a double precision addition using the E (Extension) register to increase the accumulator to 48 bits. Double precision operands are stored in consecutive memory locations. The more significant half resides in any even-address memory location while the less significant half resides in the next, odd-address memory location. Double addition is a two step process in which the less significant half-word is added to the E register after which the more significant half-word is added to the A register considering any carry from the first part.

During the first memory cycle of DADD, TOF (Time of Operand Fetch) is set. In addition to the signals GMB (Gate Memory to Buffer) and GCPB (Gate Clock Pulse to Buffer) which are used to load data returning from memory, GOM (Gate Operand address to Memory) and GOPIM (Gate Operand plus 1 to Memory) are used to fetch the less significant operand word from the odd numbered memory address. GOPIM forces the least significant bit of the operand address (which must be zero for any even address) to a one.

The second memory cycle is characterized by both TOF and TEX1 (Time of Execution 1) being set. Since only GOM is true this subsequent operand fetch is the more significant half from the even numbered memory address.

TEX1 enables the addition of the contents of B to the contents of E by enabling GBUB (Gate B to adder input B), GEUA (Gate E to adder input A), and GSE (Gate Sum to E). Any carry produced by

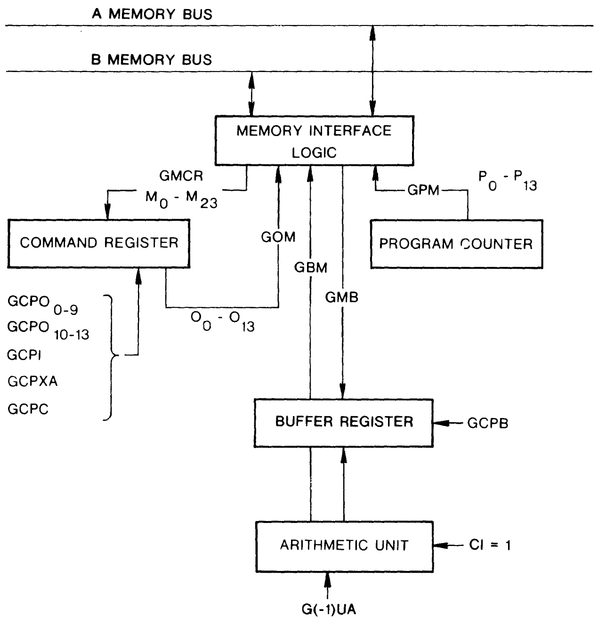


FIGURE II-4-66. SOM INSTRUCTION FLOW DIAGRAM

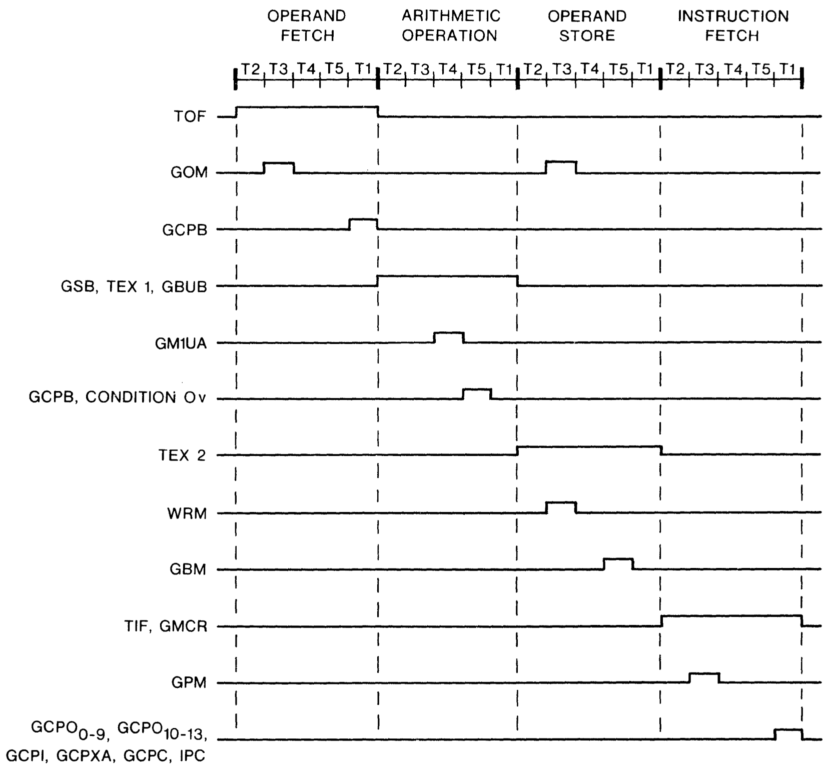


FIGURE II-4-67. SOM INSTRUCTION TIMING DIAGRAM

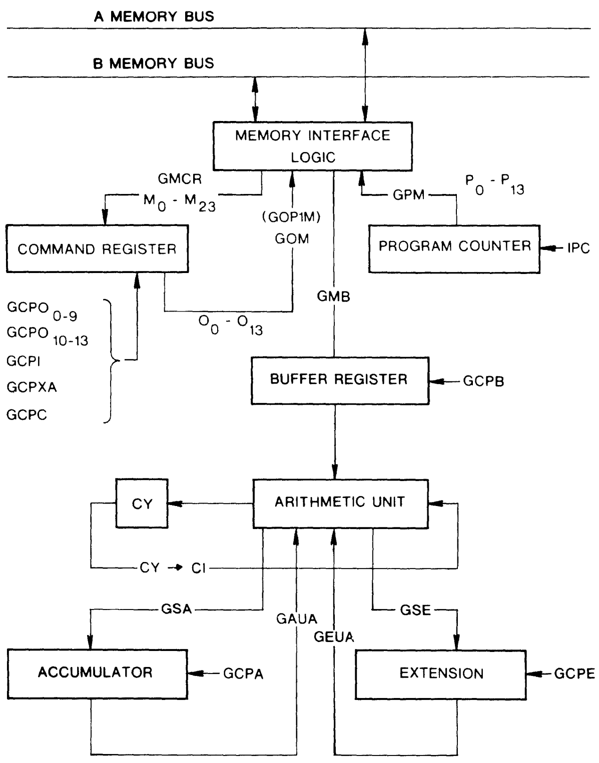


FIGURE II-4-68. DADD INSTRUCTION FLOW DIAGRAM

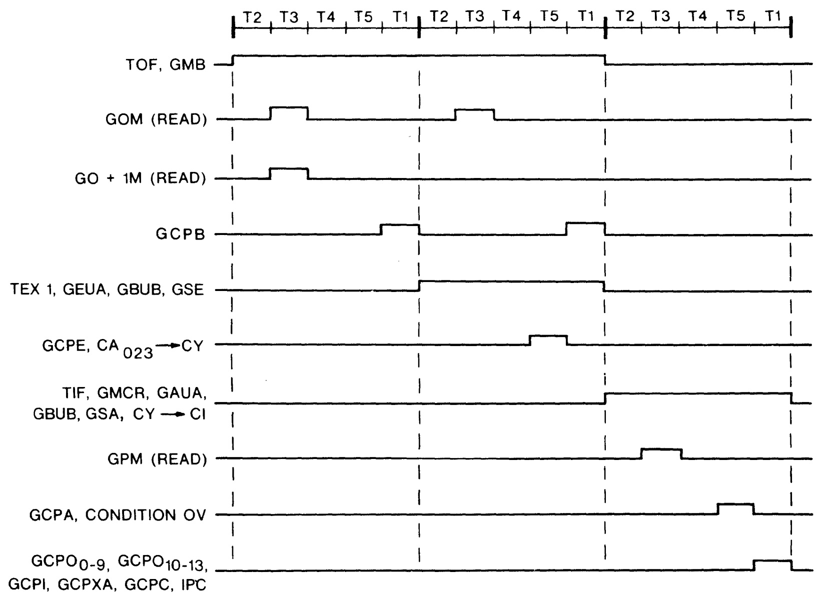

T2 T3 T4 T5 T1 T2 T3 T4 T5 T1 T2 T3 T4 T5 T1
TOF, GMB
GOM (READ)
GO + 1M (READ)
GCPB
TEX 1, GEUA, GBUB, GSE
GCPE, CA 023 → CY
TIF, GMCR, GAUA, GBUB, GSA, CY → CI
GPM (READ)
GCPA, CONDITION OV
GCPO0-9, GCPO10-13, GCPI, GCPXA, GCPC, IPC

FIGURE II-4-69. DADD INSTRUCTION TIMING DIAGRAM

the most significant bit of the adder is directed to CY (Carry flip-flop). At T5, GCPE (Gate Clock Pulse to E) loads the least significant half sum into E and stores any carry in CY.

During the third memory cycle TIF (Time of Instruction Fetch) is set, TOF clears, and TEX1 clears. The normal instruction fetch overlaps the addition of the more significant half of the double precision operand. The new contents of B are added to A by enabling GAUA (Gate A to adder input A), GBUB (Gate B to adder input B) and GSA (Gate Sum to A). The state of CY is gated to the adder via CI (Carry In). At T5, GCPA (Gate Clock Pulse to A) loads the more significant half Sum into A and sets OV (Overflow) if a change of sign results.

4.5.32 DSUB Instruction

Figures II-4-70 and II-4-71 are the flow and timing diagrams for the DSUB instruction. Double Subtract is a double precision addition using the E (Extension) register to increase the accumulator to 48 bits. Double precision operands are stored in consecutive memory locations. The more significant half resides in any even-address memory location while the less significant half resides in the next, odd-address memory location. Double Subtract is a two step process in which the less significant half-word is subtracted from the E register after which the more significant half-word is subtracted from the A register considering any borrow from the first part.

DSUB is analogous with DADD except that the two's complement of the half operands from memory are formed by using GBFUB (Gate B False to adder input B) during both addition operation and by injecting a one into CI (Carry Input) during the first addition operation. All other signals and timing are identical with DADD.

4.5.33 DLD Instruction

Figures II-4-72 and II-4-73 are the flow and timing diagrams for the DLD instruction. The DLD instruction loads the A and E registers from two consecutive memory locations. DLD is executed exactly the same as DADD except that the A input to the adder is not enabled by either GEUA or GAUA. Consequently, the value of the operand replaces the contents of A and E.

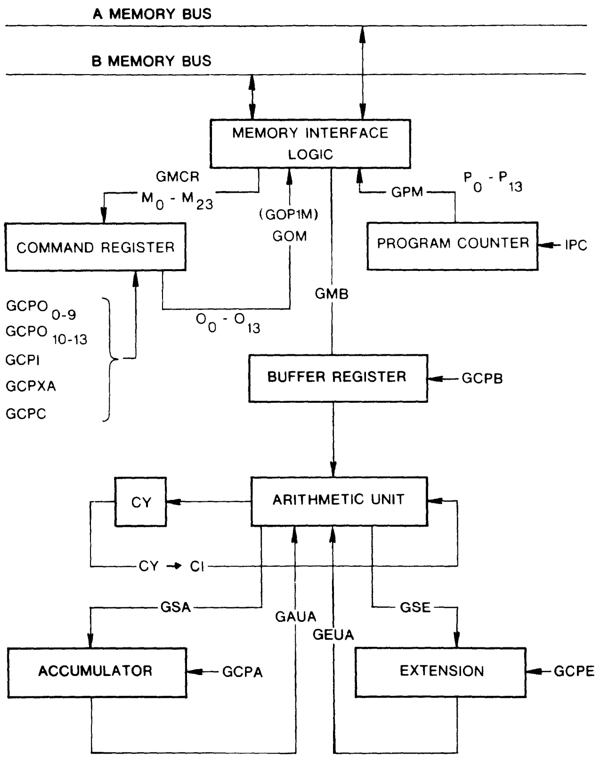


FIGURE II-4-70. DSUB INSTRUCTION FLOW DIAGRAM

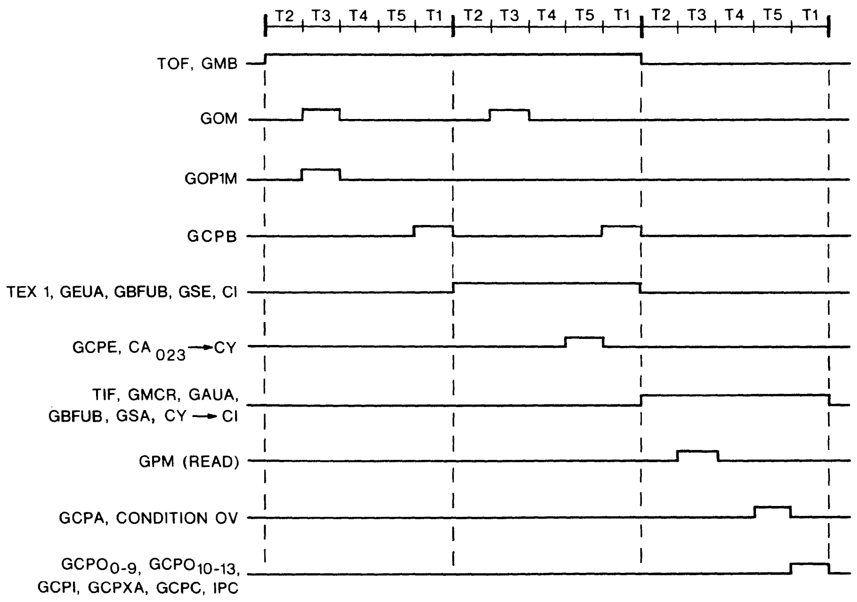

T2 T3 T4 T5 T1 T2 T3 T4 T5 T1 T2 T3 T4 T5 T1
TOF, GMB
GOM
GOP1M
GCPB
TEX 1, GEUA, GBFUB, GSE, CI
$GCPE, CA_{023} \rightarrow CY$
TIF, GMCR, GAUA, GBFUB, GSA, CY → CI
GPM (READ)
GCPA, CONDITION OV
$GCPO_{0-9}$, $GCPO_{10-13}$, GCPI, GCPXA, GCPC, IPC

FIGURE II-4-71. DSUB INSTRUCTION TIMING DIAGRAM

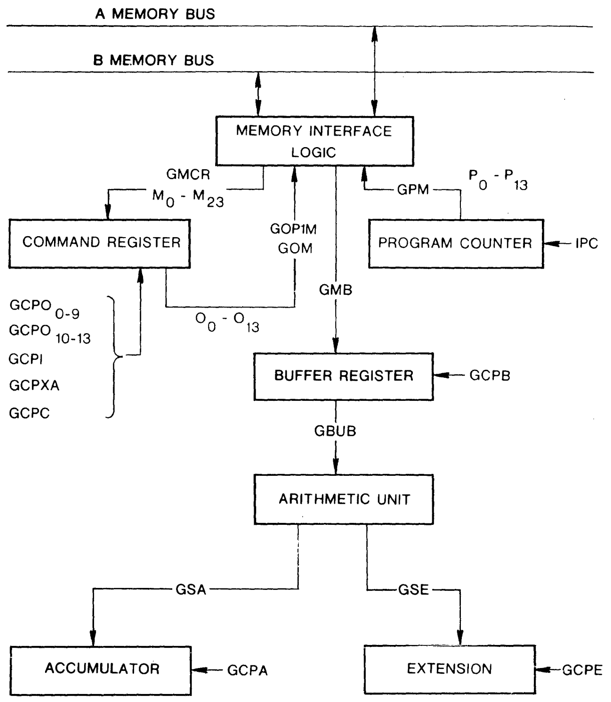


FIGURE II-4-72. DLD INSTRUCTION FLOW DIAGRAM

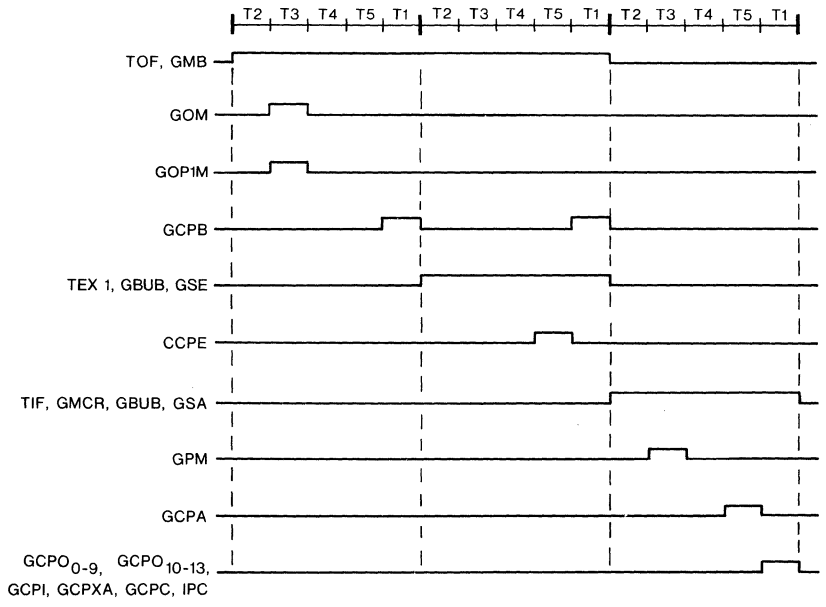


FIGURE II-4-73. DLD INSTRUCTION TIMING DIAGRAM

4.5.34 DST Instruction

Figures II-4-74 and II-4-75 are the flow and timing diagrams for the DST instruction. The DST instruction stores the E register contents into the memory location specified by the operand address plus 1 (the operand address must be even), and stores the contents of the A register into the memory location specified by the operand address. DST is a three memory cycle instruction.

During the first memory cycle, the contents of the E register are stored in memory. GOPIM (Gate Operand address Pulse 1 to Memory) forces the memory address to be odd. Also at T3, WRM (Write Memory) is energized. GEUA and GSB are both energized to gate the contents of E through the AU to the input of B. At T4, GCPB (Gates a Clock Pulse to B), copies the contents of E into B. At T5, the contents of B are gated to memory and stored at the address gated to memory at T3.

The contents of the A register are stored into memory in a similar manner. During the second memory cycle, GAUA and GSB gate the contents of A through AU to the input of B. Since GOPIM is absent, the memory address sent to memory at T3 along with WRM is even (assuming a correct program). At T4, GCPB copies the contents of A into B. At T5 the contents of B are gated to memory and stored.

4.5.35 MUL Instruction

Figures II-4-76 and II-4-77 are the flow and timing diagrams for the MUL instruction. The MUL instruction forms the product of the contents of the memory location specified by the operand address and the contents of the accumulator. The result of multiplication is a double word product in A and E. A holds the more significant half. The MUL instruction is executed by performing a repetitive operation consisting of an addition followed by a right shift of the contents of both A and E.

MUL starts with an operand fetch cycle. During this cycle, the operand is fetched from memory, gated to B with GMB, and clocked into B with GCPB. A is gated to E and clocked into E at T4 with GCPE. Zeros are gated to A, and clocked into A at T4 with GCPA. At T1 of this cycle, CO (iteration Counter) is set to 23.

The second phase of MUL (TEX1) is executed 23 times. During each iteration, the contents of A are gated to the A-input of the AU with GAUA and the contents of B are gated to the B-input of the AU with GBUB. At T5 of each iteration, GSA gates the sum output of the AU to A. If, at T5, the least significant bit of the E

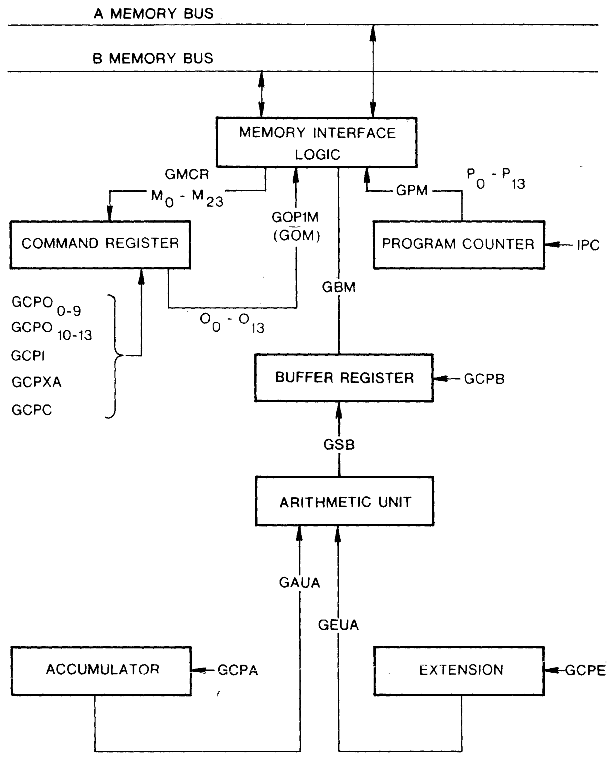


FIGURE II-4-74. DST INSTRUCTION FLOW DIAGRAM

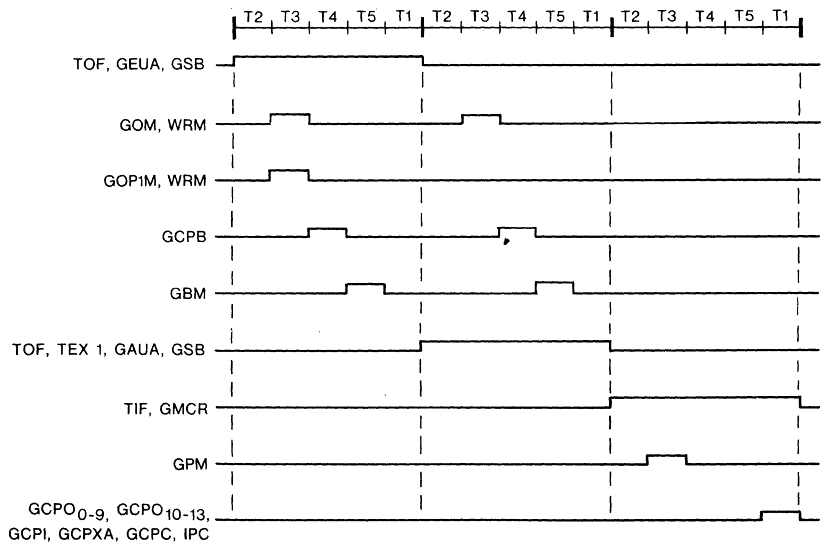

T2 T3 T4 T5 T1 T2 T3 T4 T5 T1 T2 T3 T4 T5 T1
TOF, GEUA, GSB
GOM, WRM
GOP1M, WRM
GCPB
GBM
TOF, TEX 1, GAUA, GSB
TIF, GMCR
GPM
GCPO0-9, GCPO10-13, GCPI, GCPXA, GCPC, IPC

FIGURE II-4-75. DST INSTRUCTION TIMING DIAGRAM

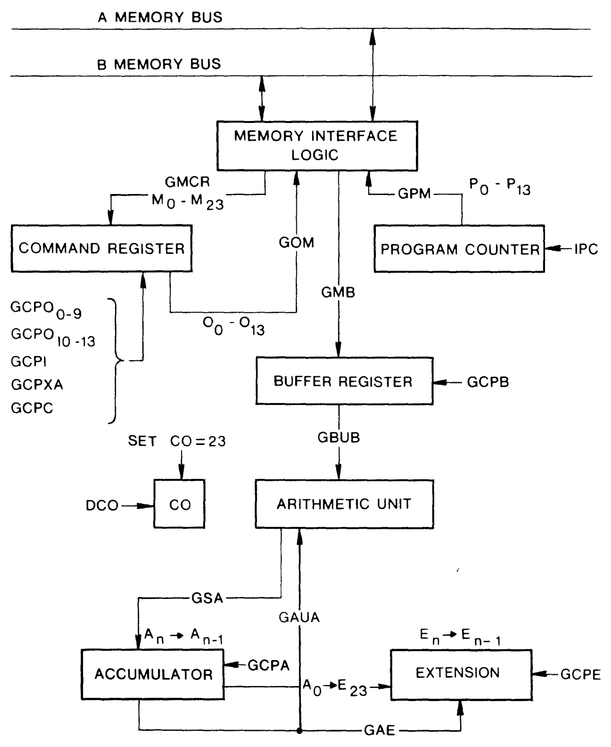


FIGURE II-5-76. MUL INSTRUCTION FLOW DIAGRAM

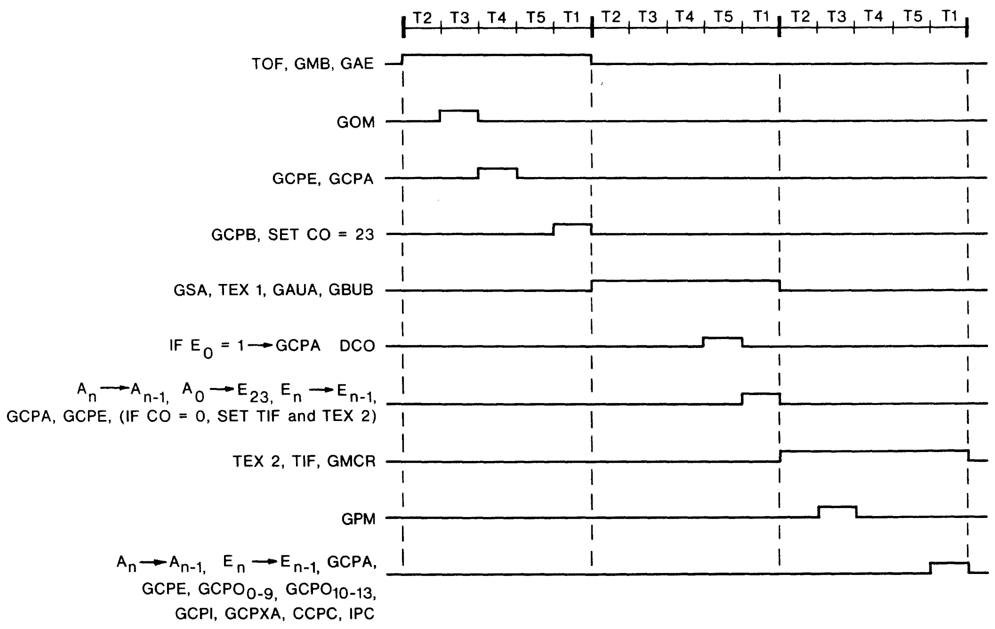

T2 T3 T4 T5 T1 T2 T3 T4 T5 T1 T2 T3 T4 T5 T1
TOF, GMB, GAE
GOM
GCPE, GCPA
GCPB, SET CO = 23
GSA, TEX 1, GAUA, GBUB
IF E0 = 1 → GCPA DCO
An → An-1, A0 → E23, En → En-1, GCPA, GCPE, (IF CO = 0, SET TIF and TEX 2)
TEX 2, TIF, GMCR
GPM
An → An-1, En → En-1, GCPA, GCPE, GCPO0-9, GCPO10-13, GCPI, GCPXA, CCPC, IPC

FIGURE II-4-77. MUL INSTRUCTION TIMING DIAGRAM

register (EO) is a "1," GCPA gates the new sum into A. Otherwise, the contents of A are not changed. At T1 of each iteration, the contents of A and E are shifted right one bit position with a zero entering A23. The signals which are used to control the shift right are:

SAR (Shift Accumulator Right)
SER (Shift Extension Right)

All inputs to A23 are inhibited resulting in a "0" entering with each shift. GCPA gates one clock to the A register and GCPE gates one clock to E to complete the shift.

At T5 of each iteration, DCO decrements the CO counter. If at T1 the counter contents are zero, the second phase is terminated by clearing TEX1 and setting TEX2. The TEX2 phase causes one additional right shift of A and E to account for the 24th bit which must always be "0" (positive numbers only). The next instruction fetch is concurrent with TEX2.

Figure II-4-78 is an abbreviated example of multiplication. The registers are only five (5) bits long to restrict the length of the example. During the operand fetch, the example shows that the B register takes on the value of the operand from memory, the A register is set to zero, the E register takes on the value originally in A and CO takes on the value 4. S is the sign bit of A which in practice must be "0." For each add and shift, the CO counter is decremented. When CO = 0, the contents of A and E are shifted right one bit position to dispose of S.

4.5.36 DIV Instruction

Figures II-4-79 and II-4-80 are the flow and timing diagrams for the DIV instruction. The DIV instruction treats the contents of the A and E registers as one 47 bit word with A the more significant portion. Bit 23 of the accumulator is the sign which must be "0" (positive number). The DIV execution forms the quotient by dividing the contents of A and E by the contents of the memory location specified by the operand address. The DIV instruction requires 26 memory cycles for execution.

The first memory cycle is used to fetch the divisor and place it in the B register. This is a normal operand fetch cycle. CO (iteration counter) is set to 23 at T1 of this memory cycle.

	MULTIPLICAND		MULTIPLIER	
	S 1 0 1 1		0 1 1 1 1	
	B	A	E	CO
	0 0 0 0 0	S 1 0 1 1	0 0 0 0 0	X
operand fetch	LOAD B 0 1 1 1 1	CLEAR A 0 0 0 0 0	COPY A S 1 0 1 1	4
E0 = 1,	Add B to A Right Shift A & E	0 1 1 1 1 0 0 1 1 1	1 S 1 0 1	3
E0 = 1,	Add B to A Right Shift A & E	1 0 1 1 0 0 1 0 1 1	0 1 S 1 0	2
E0 = 0,	No change Right Shift A & E	0 1 0 1 1 0 0 1 0 1	1 0 1 S 1	1
E0 = 1,	Add B to A Right Shift A & E	1 0 1 0 0 0 1 0 1 0	0 1 0 1 S	0
CO = 0,	Right Shift A & E	0 0 1 0 1	0 0 1 0 1	

DOUBLE LENGTH PRODUCT IN BOTH A & E.

FIGURE II-4-78. SIMPLIFIED MULTIPLICATION EXAMPLE

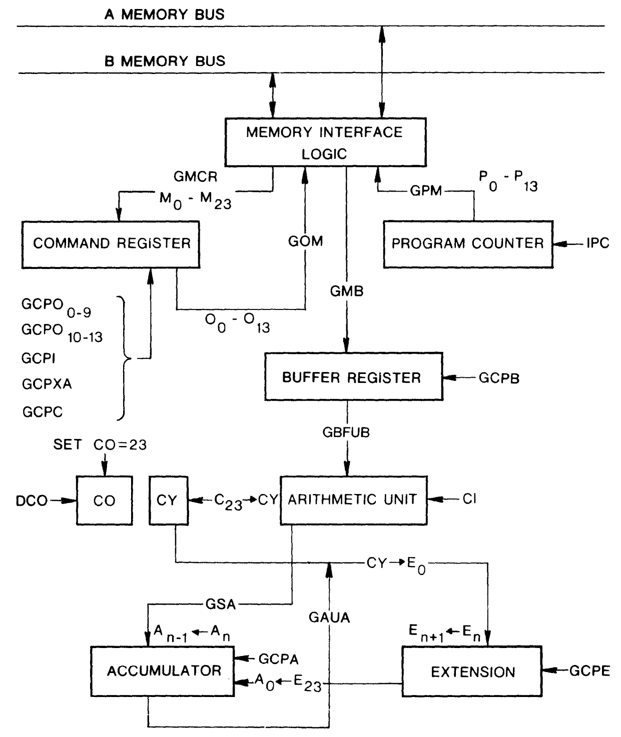


FIGURE II-4-79. DIV INSTRUCTION FLOW DIAGRAM

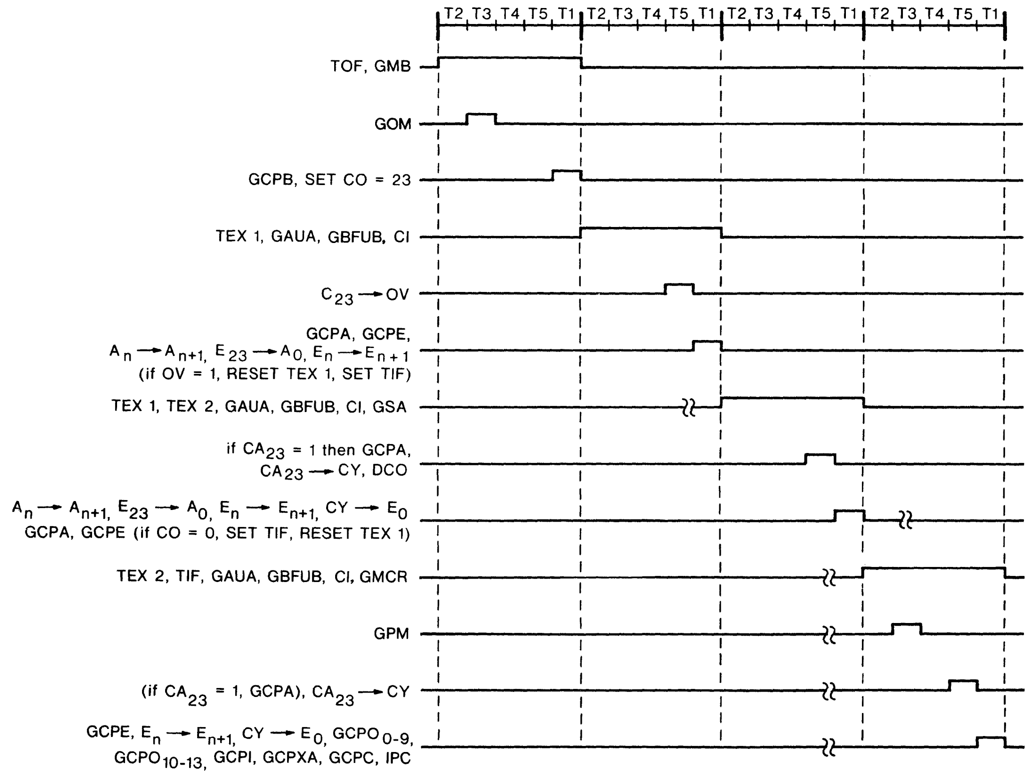

T2 T3 T4 T5 T1 T2 T3 T4 T5 T1 T2 T3 T4 T5 T1 T2 T3 T4 T5 T1
TOF, GMB
GOM
GCPB, SET CO = 23
TEX 1, GAUA, GBFUB, CI
$C_{23} \rightarrow OV$
GCPA, GCPE,
$A_n \rightarrow A_{n+1}$, $E_{23} \rightarrow A_0$, $E_n \rightarrow E_{n+1}$
(if OV = 1, RESET TEX 1, SET TIF)
TEX 1, TEX 2, GAUA, GBFUB, CI, GSA
if CA_{23} = 1 then GCPA,
$CA_{23} \rightarrow CY$, DCO
$A_n \rightarrow A_{n+1}$, $E_{23} \rightarrow A_0$, $E_n \rightarrow E_{n+1}$, $CY \rightarrow E_0$
GCPA, GCPE (if CO = 0, SET TIF, RESET TEX 1)
TEX 2, TIF, GAUA, GBFUB, CI, GMCR
GPM
(if CA_{23} = 1, GCPA), $CA_{23} \rightarrow CY$
GCPE, $E_n \rightarrow E_{n+1}$, $CY \rightarrow E_0$, $GCPO_{0-9}$,
$GCPO_{10-13}$, GCPI, GCPXA, GCPC, IPC

FIGURE II-4-80. DIV INSTRUCTION TIMING DIAGRAM

The second phase of DIV (TEX1) checks that the magnitude of the quotient does not exceed 23 bits, since the twenty fourth bit is reserved for sign of the quotient. This phase is identical with subsequent iterations of the divide cycle except that the quotient bit produced is used to load OV (overflow indicator). If OV is set (quotient bit = "1"), a normal instruction fetch follows. If OV is not set, the magnitude of the quotient is valid and the next phase of division is entered. During this phase both TEX1 and TEX2 are set and CO counts the iterations. For each iteration, the contents of B are subtracted from the contents of A in the AU. GSA is energized at each T5. If, at T5, the carry out of the AU is "1," a clock is gated to A with GCPA and the results of the subtraction are clocked into A. Otherwise, the contents of A are not changed.

The carry out of the AU is also saved in CY (carry flip-flop) at T5, and DCO decrements the iteration counter CO. At T1, both A and E are shifted left. The signals SAL (Shift A left), SEL (Shift E Left) connect A and E as a double length shift register, and the combination of DIV with TEX2 enables the output of CY to the least significant bit of E. GCPA and GCPE complete the shift. This cycle repeats until CO is zero.

The final phase of division (TEX2) is entered when CO equals zero. The same rules apply for the final iteration except that A contains the final remainder and only E needs a left shift to enter the last quotient bit from CY. A normal instruction fetch overlaps the final iteration.

Figure II-4-81 is an abbreviated illustration of division. The registers illustrated are only five (5) bits long to limit the length of the example. The initial conditions are illustrated at the top of the figure. The contents of B are shown as the two's complement of the actual value of the divisor after the operand fetch cycle as this is the value actually added to the accumulator for each iteration. Remember, however, that B holds the divisor as shown above. The operand fetch cycle loads the B register and sets CO to 4. The first subtraction produces no carry, so the division continues. The initial state of CY has no effect. Its progress is shown for reference only. A and E are shifted left, CY entering E0 at the right. The DIV loop is executed four (4) times. After each subtraction, A and E are shifted left. If any of the subtractions produce no carry, the difference is not entered into A.

The fifth subtraction is performed during the instruction fetch cycle. If a carry is produced from this last subtract operation, the difference is entered into A and constitutes the remainder. Only E is shifted left at T1 with CY going to E0. This last shift completes the quotient in E.

The divide instruction is not a 2's complement arithmetic function and assumes a pure binary, positive number fields.

The divisor, pointed to by the effective operand address, is equivalent to a positive, 2's complement, number with a valid range of +1 to + $(2^{23} - 1)$.

NOTE

A zero value will terminate the divide operation with a "divide overflow".

The dividend for the operation is located in the A and E registers. This is a double precision positive binary number with the following weights.

Most Significant Half in A
Least significatn half in E
The binary weights are:

$E_0 = 2^0$

$E_{22} = 2^{22}$

$E_{23} = 2^{23}$

$A_0 = 2^{24}$

$A_{22} = 2^{46}$

A_{23} = UNDEFINED

The range of the dividend is

$0 \leq A,E \leq 2^{47} - 1$

The range of valid operations is bounded by:

$A,E/(Me) \leq 2 \quad -1$

$A/(Me) < 1$

These bounds are consistant for values in their specified ranges.

	DIVISOR		DIVIDEND		
	0 0 0 1 0		0 0 0 0 1	1 1 0 1 1	
	(COMPL)		A	E	CO
operand fetch	1 1 1 1 0	CY	0 0 0 0 1	1 1 0 1 1	x
compare difference		0	1 1 1 1 1	1 1 0 1 1	
Left Shift A & E		0	0 0 0 1 1	1 0 1 1 x	4
1st Subtraction		1	0 0 0 0 1		
Left Shift A & E		0	0 0 0 1 1	0 1 1 x 1	3
2nd Subtraction		1	0 0 0 0 1		
Left Shift A & E		0	0 0 0 1 0	1 1 x 1 1	2
3rd Subtraction		1	0 0 0 0 0		
Left Shift A & E		0	0 0 0 0 1	1 x 1 1 1	1
4th Subtraction		0	1 1 1 1 1		
Left Shift A & E		0	0 0 0 1 1	x 1 1 1 0	0
5th Subtraction		1	0 0 0 0 1		
Left Shift E Only			0 0 0 0 1	1 1 1 0 1	
			REMAINDER	QUOTIENT	

FIGURE II-4-81. SIMPLIFIED DIVISION EXAMPLE

FAIRCHILD
SYSTEMS TECHNOLOGY

CHAPTER III
MEMORY SYSTEM

Section 1
Memory System

1.0 INTRODUCTION

The memory system provided with the FST-1 computer is a random access storage unit. The memory performs a full cycle in 785 nanoseconds, a split cycle within 1275 nanoseconds. Access time is 375 nanoseconds maximum, measured from the leading edge of the initiate signal, to the time data from a specified address is stable on the data output lines.

1.1 GENERAL DESCRIPTION

The memory module is contained in the right hand door of the Sentry System Mainframe as shown in figure III-1-1.

The FST-1 memory system (memory system A) is composed of two modular memory units, each providing a storage capacity of 8192 locations for 12 binary bits. Together, the memory modules provide 8192 locations for a 24-bit computer word. Each module is self-contained, providing all necessary diode addressing matrixes and drive circuitry for complete memory operation. As a self-contained unit, each module is also directly interchangeable with any other memory module within the FST-1 as a plug-in unit. Each memory module interfaces with the central processor unit (CPU) and the I/O interface module via 12 CPU-to-memory data/address input lines, 12 memory-to-CPU data output lines, and 6 control and memory status lines. As an option, an additional 8192 24-bit word capability is available by using a total of four identical memory modules. The first, or basic, 8192 word memory is then designated as memory A and the additional 8192 word memory is designated as memory B. Memory A and Memory B may be operated independant of each other by the CPU, and each is interfaced to a separate respective A memory and B memory bus.

The modules are designed to be inserted in a system chassis assembly which in turn is inserted into the mainframe. Figure III-1-2 shows a memory chassis with full complement of modules. The core memory arrays and all related circuitary are contained within the basic storage module.

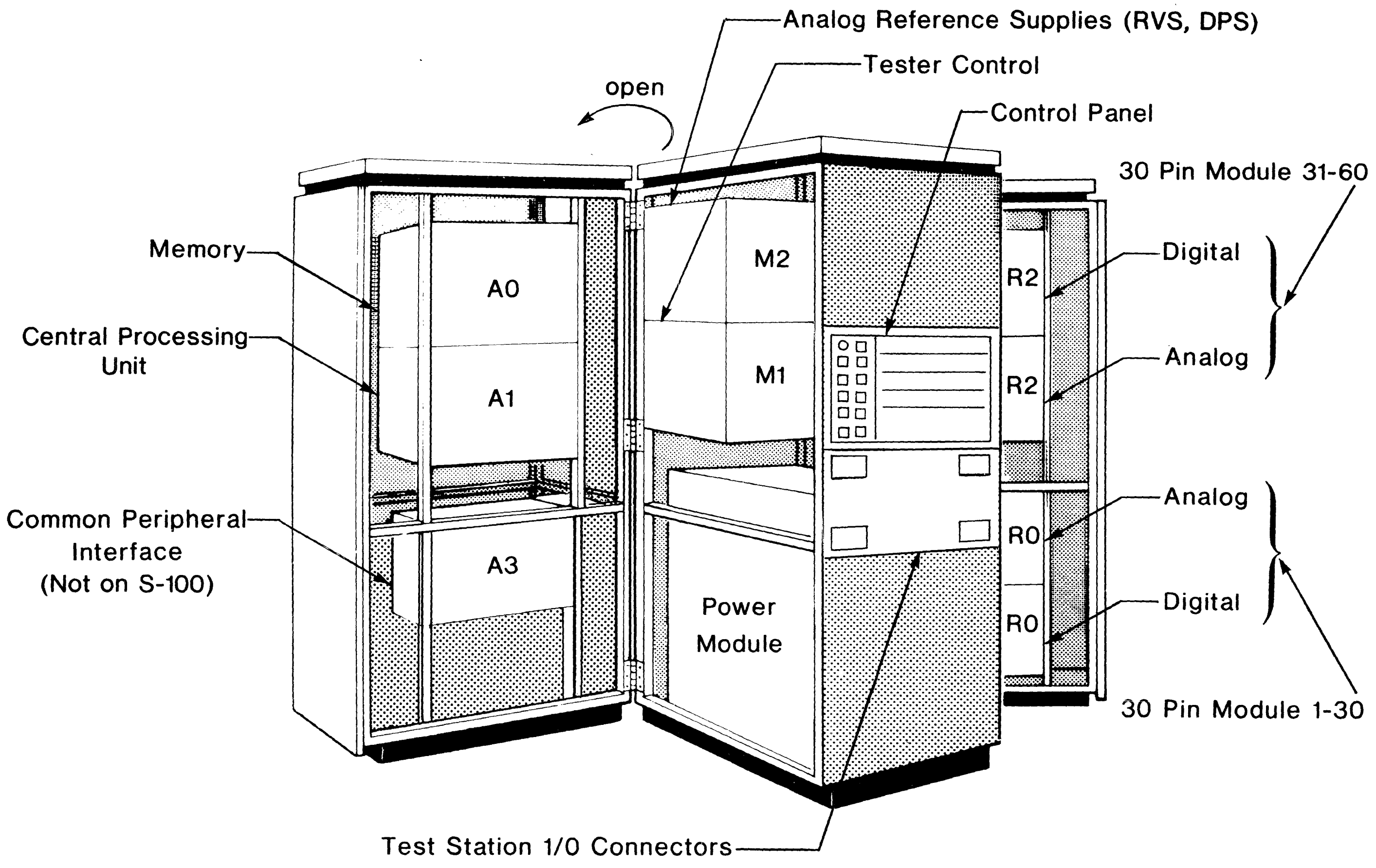


FIGURE III-1-1. SENTRY SYSTEM MAINFRAME

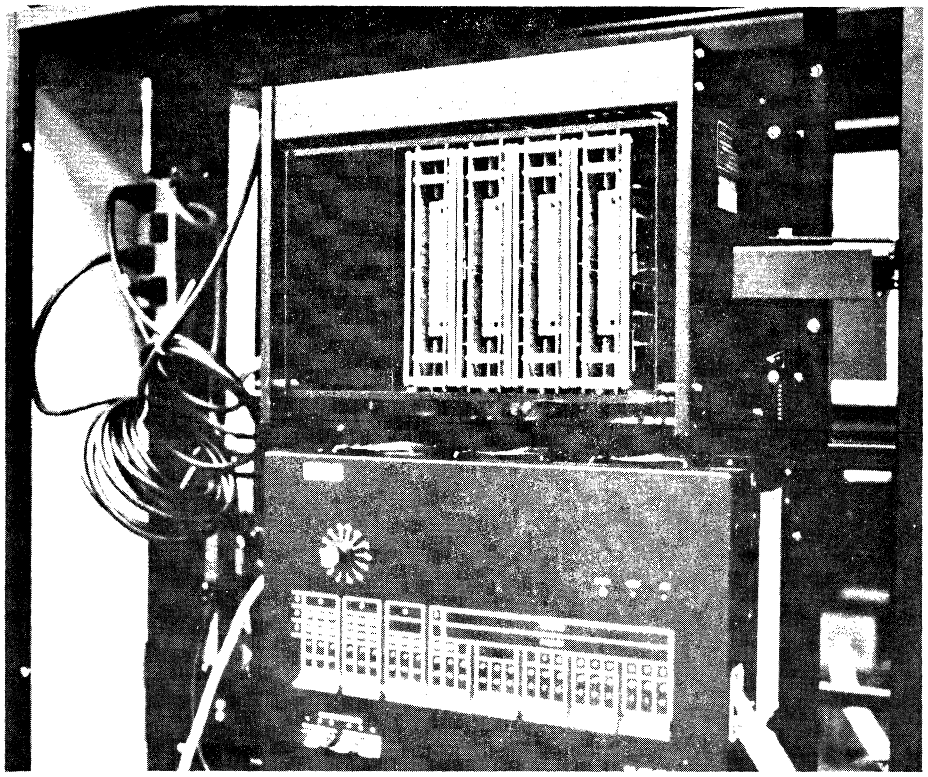


FIGURE III-1-2. MEMORY SYSTEM WITH FULL COMPLEMENT OF MODULES

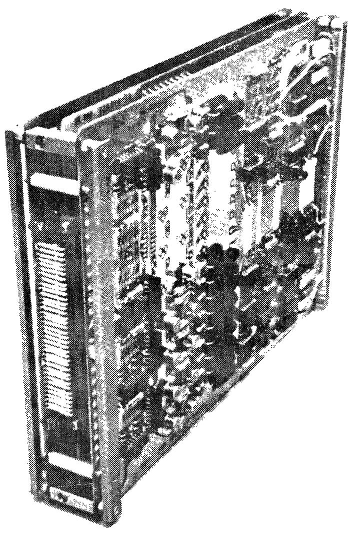


FIGURE III-1-3. BASIC STORAGE MODULE

1.1.1 Basic Storage Module (BSM)

The basic storage module is made up of three printed circuit assemblies (approximately 8" X 11", arranged in sandwich fashion, as shown in Figure III-1-3.

The three assemblies are interconnected by means of standard p.c. board edge-connectors. The number of external connections to the storage module is minimized, and the compact size of component boards results in a rugged assembly.

1.1.2 The Memory Magnetic Assembly (MMA)

The MMA is shown in Figure III-1-4.

This module contains:

The core array. This is the basic storage area for the system, and consists of an 8K word X 12 bit magnetic core array, arranged on one double-sided board;

The sense amplifiers, and sense amp strobe control;

An X-diode matrix containing 8 X 16 X-diode pairs, and a Y-diode matrix containing 8 X 8 Y diode pairs.

1.1.3 The Memory Switch Assembly (MSA)

The MSA, shown in Figure III-1-5 controls the X and Y drive current and switch circuitry and for these purposes contains:

The two X current sources (read and write);

The two Y current sources (read and write);

The X switches: 16 sink switch pairs and 8 drive switch pairs;

The Y switches: 8 sink switch pairs, and 8 drive switch pairs;

X and Y address decoding logic;

2.8 V regulator circuit for the X and Y current sources.

It also contains:

The address register which stores the incoming address information and drives the address decoding logic;

The control and timing logic which receives the control signals from the CPU and generates the internal timing;

Fanout gates for control timing signals;

The power fail circuit.

The Memory Bit Assembly (MBA)

The MBA is shown in figure III-1-6 and contains:

The 12 inhibit drivers;

The inverters for sense amplifier outputs;

The data register and its associated control logic.

Section 2
Principles Of Operation

2.0 INTRODUCTION

This section describes briefly the principles of digital storage devices, and random accessing, and outlines three-wire, 3D core memory organization.

2.1 THE RANDOM—ACCESS, DIGITAL STORAGE DEVICE

A computer memory is a device for storing information. In the case of the digital memory, binary-coded information is stored in the form of "1's" and "0's". This data is set into unique locations in the memory which are identified by addresses.

Information stored in the memory is used by the processor. Information is pulled from the memory in a manner analogous to written information being pulled from a file. After it has been used, it is either returned to the memory or destroyed.

To retrieve specific groups of information from a memory, certain addresses are accessed. There are basically two types of access: random and sequential. Since the core memory is a random-access memory, this section will deal only with random access. With random access, the access time (time required to retrieve information) is independent of storage location. For instance, location M could be accessed immediately after location A, without reference to B, C, D, etc. This means that overall system operations can be performed with great speed.

The Core Memory

The basic element of the core memory is the ferrite core: a doughnut-shaped (toroidal) element, which stores the basic unit of digital information, one bit (binary digit). The core memory uses cores with outside diameters in the range of 14 to 22 mils. The cores are strung on three wires, and arranged in two-dimensional (planar) arrays.

Memory Capacity

The smallest unit stored in the memory is one bit. The bits are considered in functional groups called bytes, and the bytes are themselves considered in groups called words. The capacity of a memory is described in terms of the number of words it will hold, and the number of bits in the word.

2.1.1 Magnetic Characteristics of the Ferrite Core

Information is "stored" in a ferrite core in terms of the direction of magnetization imposed upon the core. The direction of magnetization is altered by means of applied currents, and the core in effect "remembers" the direction of the current sent through it last.

A magnetic core is a non-linear element, thus its state remains unchanged when a half current passes through it. Its state is reversed when a full current, consisting of two half currents, is applied to it in the reverse direction.

The system uses two such coincident half currents to change the state of the core and also to allow address selection. Half currents are applied along the X and Y lines, so that only the core at the intersection of the lines "sees" a full current and is therefore addressed.

The effect of the applied magnetic field on the magnetization state of the core is illustrated by the hysteresis loop in Figure III-2-1.

2.1.2 The 3D System

Core memory system characteristics are defined in terms of the organization and geometry of the system: that is the logical structure of the core array, and the driving and sensing electronics.

3D Configuration

In a 3D system, the cores are arranged in matrices, and there are as many matrices (mats), as there are bits in the word. Thus, a memory using 12-bit word configuration, will have 12 core mats. Each mat contains the same number of cores as there are words in the system so that an 8K system will have 8192 cores (64 X 128) in each mat.

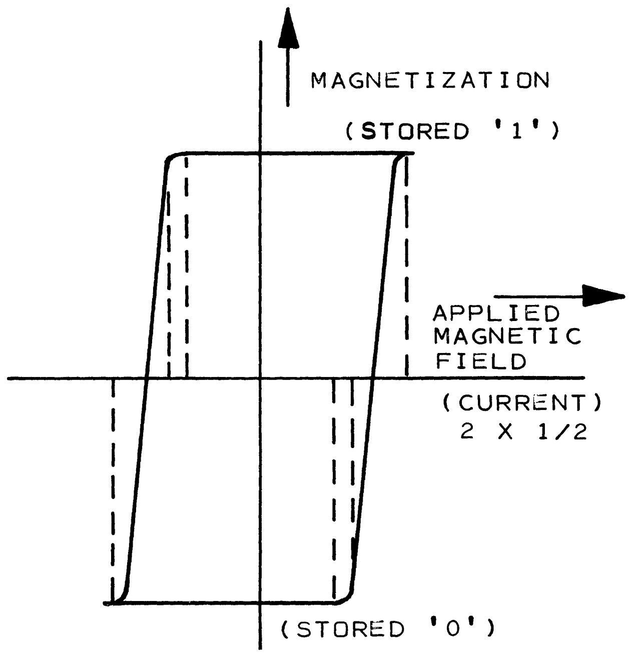


FIGURE III-2-1. HYSTERISES LOOP OF A FERRITE MEMORY CORE

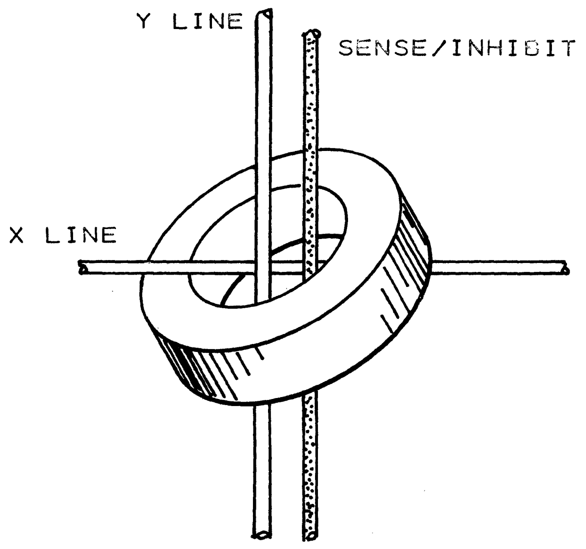


FIGURE III-2-2. FERRITE CORE MEMORY ELEMENT WITH THREE WIRES

The cores are threaded with three wires through each core:

X drive line
Y drive line
sense/inhibit line

The X and Y lines are strung through the cores at right angles to each other. These lines are used to carry half-select currents.

A common line is used for sense and inhibit. It is used to carry induced sense voltage during a read operation, and inhibit current during the write operation. Figure III-2-2 shows a ferrite core threaded with three wires.

2.1.3 Operation

The core memory is capable of two basic modes of operation: read/restore and clear/write. All other operations are variations on these two modes. During read/restore, "1's" and "0's" are retrieved from the memory and then restored into the same location. During clear/write, the data is cleared to the "0" state at the specified location, and then "1's" and "0's" written into the location. The magnetics theory behind these operations is described below.

Read/Restore Operation

To perform a read operation, two half currents (2 X 1/2) are applied to the cores along a selected X and a selected Y line. The core at the intersection of these lines is switched, say to the "0" state. If a current 2 X minus 1/2 had been applied to the same core, it would have switched to the "1" state. However, during a read operation, current is always applied in the direction which switches the core to the "0" state.

When the core switches from a "1" to a "0", the flux change is considerable, and induces a voltage in the sense line. This voltage is transmitted through the sense line and detected by the sense amplifier, resulting in a "1" being read from memory.

If the core switched to the "0" state is already in the "0" state, very little change in flux occurs, and so only a small voltage is produced. This voltage is rejected by the threshold stage in the sense amplifier, so that a "0" is read out from the memory. Figure III-2-3 shows the sense winding output curves for a "1" and for a "0".

As soon as readout is complete, a restore operation takes place during which the information readout from the memory is returned to its original location. This is accomplished by passing current along the X and Y drive lines in a direction opposite from that of the read portion of the cycle. This restore "1's" where required. Inhibit current is used, as in the write operation described below, to restore "0's".

Clear/Write Operation

At the beginning of a clear/write operation, the selected memory location is cleared, or reset to "0" by passing X and Y currents through the selected location in the direction required to set a "0" state. The information readout during this procedure is present on the sense lines but is not used.

To write a "1", X and Y currents are applied in the opposite directions from those required for a read operation. The two half currents add and attempt to change the state of the selected core. If a "1" is to be written, the core is switched.

For those cores where an "0" is to be written, inhibit current is driven through the sense/inhibit line. Since the polarity of the inhibit current is opposite to that of the Y current, the Y current is algebraically cancelled. The X current alone is not sufficient to switch the core, thus it remains in the "0" state and an "0" is written.

Figure III-2-4 shows how currents are applied to read "1's" and "0's" and write "1's" and "0's".

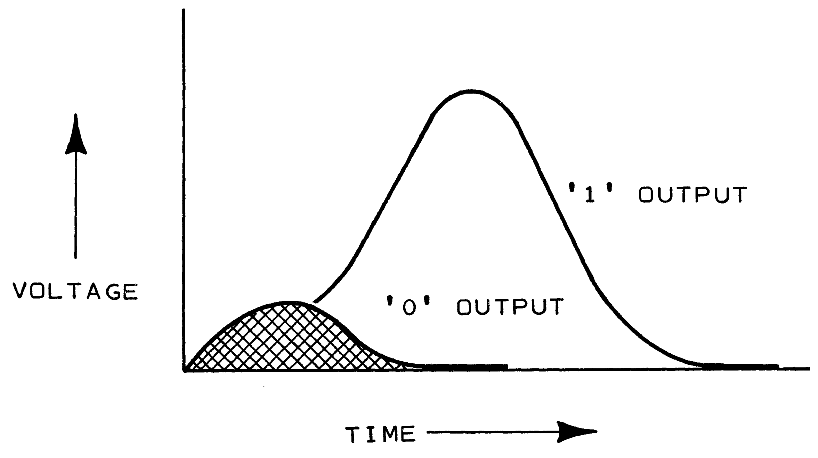


FIGURE III-2-3. SENSE WINDING OUTPUT CURVE

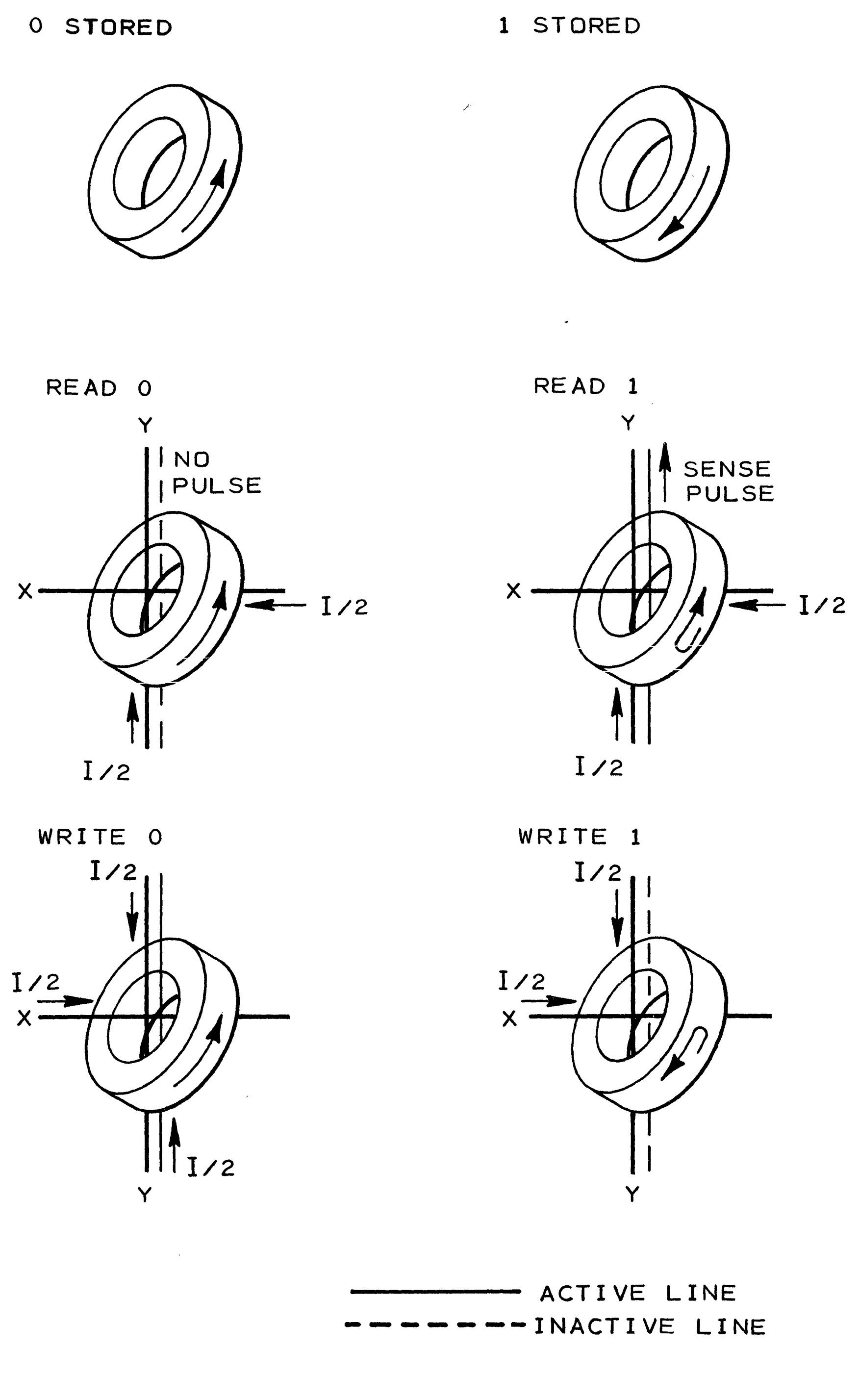


FIGURE III-2-4. READ AND WRITE OPERATIONS

Section 3
Theory Of Operation

3.0 INTRODUCTION

This section describes the organization and operation of the memory system. The input,output, and intra-system data flow are discussed in terms of a system block diagram. Each of the four standard operating modes is then described in detail in actual time sequence, showing both interface and internal timing. Finally, the circuitry represented by each of the blocks in the system block diagram, is described in detail.

3.1 GENERAL DESCRIPTION

Each system consists of three printed circuit board assemblies (the MBA, MSA and MMA) arranged in sandwich fashion to form one pluggable unit (the BSM). The basic system organization is shown in the system block diagram, Figure III-3-1. The dotted lines on this diagram show the partitioning of the system in terms of the three printed circuit boards, while the blocks indicate functional units. A description of each functional block follows.

Timing and Control

The timing and control logic on the MSA receives the control signals (RD INIT, WT INIT, READ ONLY, etc.) from the CPU and from these generates the internal timing signals (ALX, RDR, etc). These signals control the other blocks in the system, as shown in Figure III-3-1. The timing and control logic also generates the output signals, MEMORY AVAILABLE and DATA AVAILABLE/.

Address Register

Address bits 9-6 are gated from the CPU into D-type flip-flops, by the positive-going edge of AIX. BYTE ADDRESS, if required, is similarly gated. The address register functions as an open latch for address bits 7-12. Bits 0-6 are the X address bits, and bits 7-12 are the Y address bits.

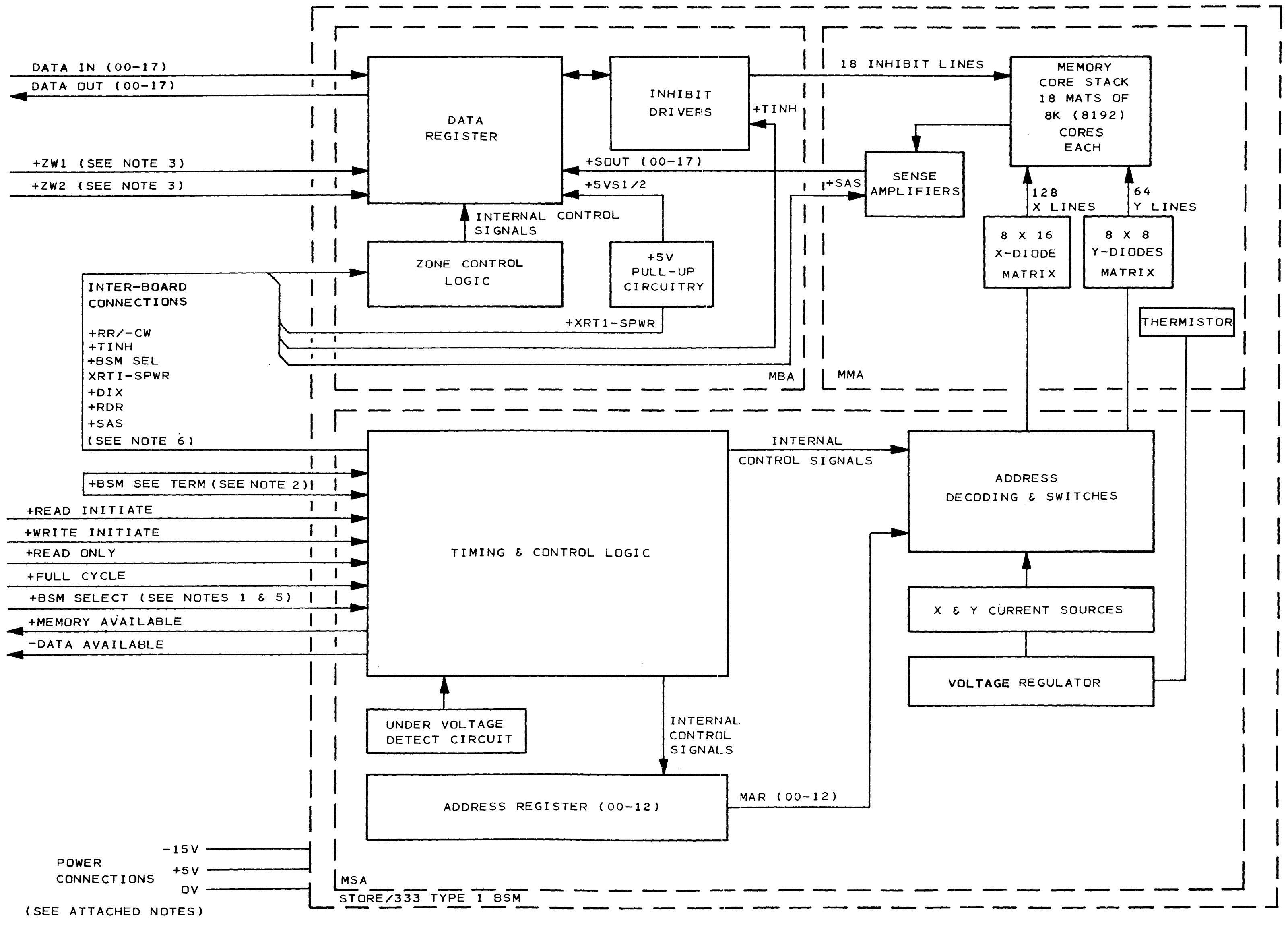


FIGURE III-3-1. MEMORY SYSTEM BLOCK DIAGRAM

Address Decoding

The address decoding circuitry receives the address information from the address register, and decodes this information to turn on one set of X and one set of Y switches, resulting in current drive through one X line and one Y line in each plane. Current 'on' times are controlled by YRT/, XRT1, XRT2/, WT1 and WT2/.

Voltage Regulator

The voltage regulator, using a thermistor located on the MMA, compensates the X and Y voltages (and hence current amplitudes) with respect to temperature.

Under Voltage Detect Circuit

This circuit prevents unacceptable decrease in supply voltage amplitudes which would cause loss of data. A decrease in amplitude of either +5V or -15V is detected and the circuit internally provides the POWER READY SIGNAL. Note that the BSM will accept an externally generated POWER READY signal if external power failure control for the BSM is desired.

Diode Matrices and Core Stack

The X and Y diode matrices translate the selection of a switch pair on the MSA board into the selection of individual X and Y lines in the core stack on the MMA board.

The core stack consists of 12 mats of 8K (8192) cores each. Each X line is threaded through given cores on all 12 mats. The Y line is similarly threaded, but at right angles to the X lines, so that a given address location specifies one core on each mat. The sense/inhibit line, on the other hand, is threaded through all the cores on a given mat, that is, all cores corresponding to a given individual data bit. Thus the X and Y lines carry address information, and the sense/inhibit lines carry data information.

Sense, Inhibit, and Data Circuits

The sense amplifiers detect the output from the cores during a read operation, and convert this output to a digital signal, which is gated into the data register by timing signal SAS.

The Data register itself stores this information until it is sent on the data out lines and/or restored into the core. It is necessary to reset the data register with timing signal RDR prior to gating in sense data with SAS. The contents of the data register are buffered onto the data output lines. The DA/ signal indicates when the data retrieved from memory is true and stable on the data output lines.

The data register also stores incoming data from the data input lines during write operations. This data is gated into the register by simultaneous timing signals DIX and DIR/.

The inhibit drivers generate inhibit current, under the control of timing signal TINH, for all data bits having a '0' in the data register. (Inhibit current, when present, runs parallel to and in the opposite direction from the Y write current, preventing the coincident X and Y write currents from switching the cores involved.) Thus, data is restored into the memory core stack during read/restore operations, and new data stored during write operations.

3.1.1 Operating Modes

The Memory operates in four modes:

1. Clear/Write;
2. Read/Restore;
3. Read/Modify/Write;
4. Read Only.

Clear/Write and Read/Restore are known as full cycle modes. Read/Modify/Write is often called a split cycle mode because of the timing pause before instructions are sent to the memory for the write portion of the cycle. Read Only is the first half of the Read/Restore cycle. It is repeated without the restore half. Each of these modes may use, without modification to the equipment, either synchronous or asynchronous cycle reinitiate.

With synchronous reinitiate, the READ INITIATE signal is returned to the false (0) state before the end of the cycle (700 ns. maximum).

To initiate the next cycle, the READ/INITIATE or WRITE/INITIATE line is brought true (1). The minimum allowable time between successive initiate signals is 750 ns.

With asynchronous cycle reinitiate, these control signals may assume their appropriate values (as explained in the detailed descriptions) prior to the end of a cycle in progress. These lines and the address and other control lines must remain in the valid state for 750 ns. maximum and remain stable for the same timing requirement as synchronous reinitiate with respect to zero time of the memory cycle. The next cycle is then automatically initiate by the memory at the time that output signal MA goes true, to give a maximum cycle time of 750 nanoseconds.

In the descriptions that follow, with one exception, the control and timing signals are active when they attain a logic level '1' (2.5 to 5.0V), which is called 'true', or 'high', and inactive at logic level '0' (0V), which is called 'false' or 'low'. The exception is the case of the output signal DATA AVAILABLE/. This line goes low to indicate that data is indeed available.

Clear/Write

The Clear/Write operation is used to store data in the memory, without reference to the information previously present at the address location used. This location is first cleared, that is, set to all zeros and the new word then written in.

Table III-3-1 shows the sequence of events for the Clear/Write operation. The required address location is cleared during steps 1 through 6, and new data is written in the same location during steps 7 through 10. Figure III-3-1 (System Block Diagram), should be used as an aid in reading this table. Figure III-3-2 shows the internal timing relationships for Clear/Write operations.

TABLE III-3-1. CLEAR/WRITE

STEP NUMBER	ACTION
1	The cycle is initiate by WRITE INITIATE = 1, which defines zero time. In addition, the following conditions must be met: Read Initiate = 0 Read Only = 0 Full Cycle = 1 Address Input lines must be stable, and the Data Input lines remain stable for 150 ns minimum.

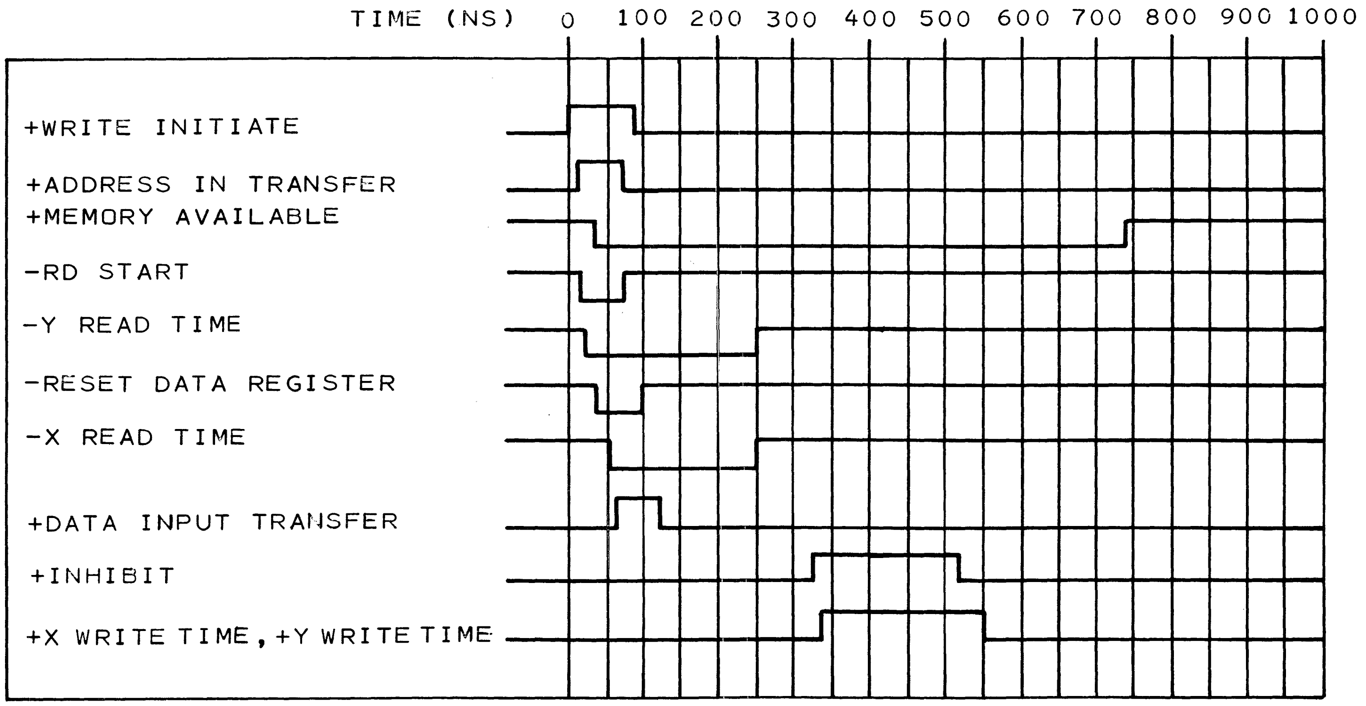


FIGURE III-3-2. INTERNAL TIMING CLEAR/WRITE

TABLE III-3-1. CLEAR/WRITE (Continued)

STEP NUMBER	ACTION
	WT INIT must remain true for 75 ns minimum. RD INIT, FC, RD ONLY, and Address Input must remain in the correct state for 100 ns minimum.
2	Address information is clocked into the address register by the AIX signal which remains high for 85 nanoseconds.
3	READ START/ initiates Y READ TIME/.
4	YRT/ results in the generation of Y read current.
5	XRT1 together with READ PHASE, generates XRT2/ which in turn allows continuation of the Y READ TIME/ control signal.
6	Input data is clocked into the data register by timing signal DIX. Input data must be stable at this time and must remain so for 100 ns minimum thereafter.
7	Output signal MA goes false 50 ns maximum after cycle start, indicating a cycle in progress, and that command inputs are blocked so as not to accept further instructions. MA remains low until about 735 (±15) nanoseconds after zero time.
8	Data registers are reset by timing signal RDR.
9	Timing signals (XRT1 and XRT2/) are sent to the address decoding circuitry, resulting in the generation of X read current.
10	Timing signal TINH is sent to the inhibit drivers, resulting in the generation of inhibit current through the cores corresponding to all data bits for which a zero is to be stored.
11	Write Timing signals (WR1, WR2/) are sent to the address decoding circuitry, resulting in the generation of X and Y write currents.

TABLE III-3-1. CLEAR/WRITE (Continued)

STEP NUMBER	ACTION
12	Output signal MA goes true at 735 ($\pm$15) ns maximum, indicating that a new cycle must be initiated. At the same time, the command inputs are gated so as to accept further instructions.

Read/Restore

The Read/Restore operation is used to read information from the cores, that is, to access a previously stored word from a given address location, and then leave the stored data unchanged when the cycle is finished. Data is first read out from the cores into the data register. It is then sent to the data output lines, completing the read portion of the 4operation. The same data is stored back into its original location, completing the restore porition.

Table III-3-2 shows the sequence of events for the Read/Restore operation. Data from the desired address location is read out during steps 1 through 8, and restored during steps 9 through 11. It is suggested that the System Block Diagram, Figure III-3-1, be used as a reference aid in reading this table. Figure III-3-3 shows the timing relationships for Read/Restore.

TABLE III-3-2. READ/RESTORE

STEP NUMBER	ACTION
1	The cycle is initiated by READ INITIATE = 1, which defines zero time. In addition, the following conditions must be met: +Write Initiate = 0 +Read Only = 0 +Full Cycle = 1 Address Input stable RD INIT must remain true for 75 ns minimum; WT INIT, RD ONLY FC and the Address Input must remain in the correct state for 100 ns minimum.

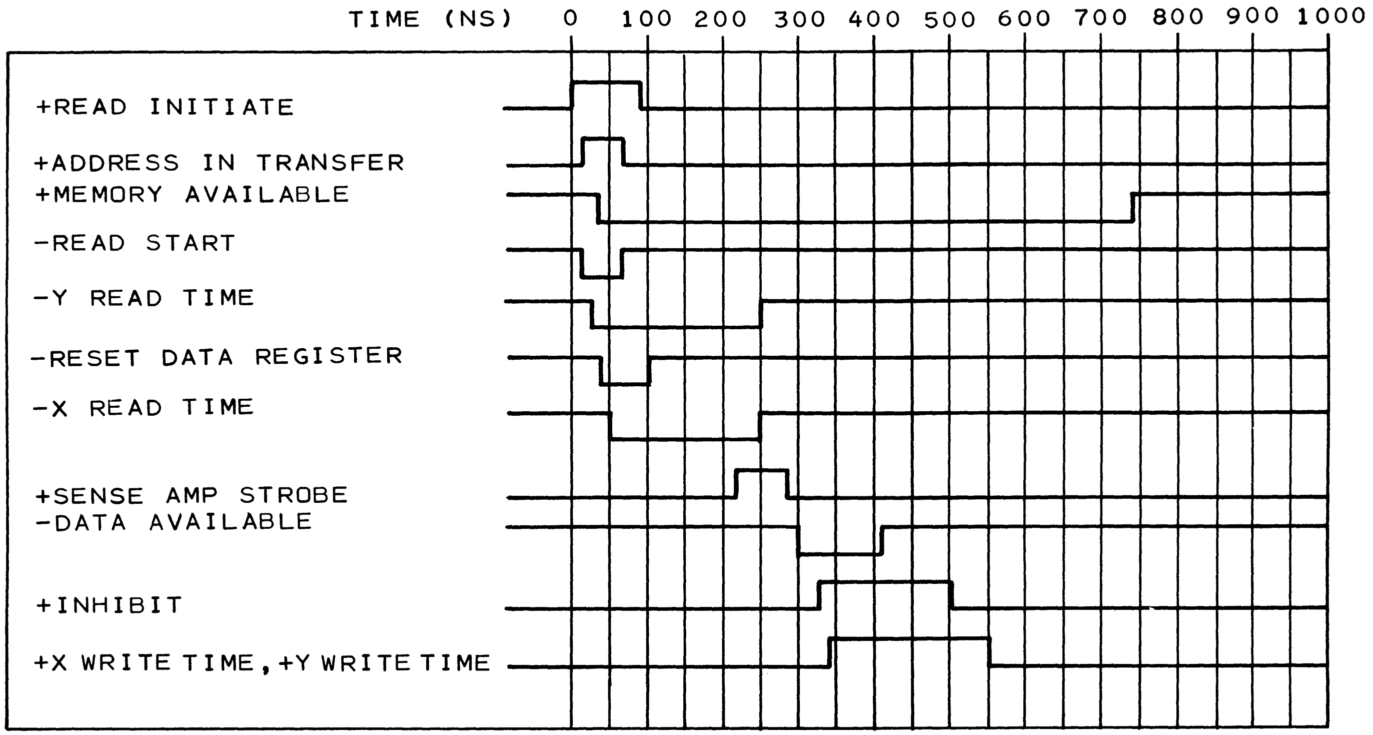


FIGURE III-3-3. INTERNAL TIMING READ/RESTORE

TABLE III-3-2. READ/RESTORE (Continued)

STEP NUMBER	ACTION
2	Address information is clocked into the address register by the AIX signal.
3	Output signal MA goes false 50 ns maximum after the start of the cycle, indicating a cycle in progress, and at the same time any additional command inputs are blocked so as not to accept further instructions. It remains low until 735 (±15) nanoseconds after zero time.
4	YRT/ results in the generation of Y read current.
5	Data registers are reset by timing signal RDR.
6	Timing signals XRT1 and XRT2/) are sent to the address decoding circuitry, resulting in the generation of X read current. Core turnover takes place at the core which receives coincident X and Y currents. The signal reaches the sense amplifier.
7	Timing signal SAS is sent to the sense amplifiers, resulting in sense amplifier output, which sets the data registers in accordance with the data read from memory.
8	Output signal DA/ goes low at data access time and remains in that state for 100 ns (±25 ns) indicating that the requested information is now available and stable on the data output lines. These lines retain the data read out of memory until the next cycle is initiated, or until MS goes low.
9	Timing signal TINH is sent to the inhibit drivers, resulting in the generation of inhibit current through the cores corresponding to all data bits for which the word to be restored contain a zero.
10	Write timing signals (WT1 and WT2/) are sent to the address decoding circuitry, resulting in the generation of X and Y write currents.

TABLE III-3-2. READ/RESTORE (Continued)

STEP NUMBER	ACTION
11	Output signal MA goes true at 735 ns ($\pm$15 ns), indicating that a new cycle may be initiated. At the same time, the command inputs are gated so as to accept further instructions.

Read/Modify/Write

The Read/Modify/Write (Split Cycle) operation is used to read a word from memory, and then to store another word in its place. Data is first read out from the specified address location into the data register. It is then sent to the data output lines completing the read porition of the operation. New data is then accepted from the data in lines into the data register, and subsequently stored in the same address location, thus completing the write portion.

Table III-3-3 shows the sequence of events for the Read/Modify/Write operation. Data from the desired address location is read out during steps 1 through 10. New data is stored during steps 11 through 16. It is suggested that the system block diagram, Figure III-3-1, be used as a reference aid in reading this table. Figure III-3-4 shows the timing relationships for Read/Modify/Write. Note that the WRITE INITIATE signal is shown going high at the minimum time allowable after the completion of the read portion of the cycle. In reality, the write cycle may be started after a considerable time pause.

TABLE III-3-3. READ/MODIFY/WRITE

STEP NUMBER	ACTION
1	The read portion of the cycle is initiated by +Read Initiate = 1, which defines zero time. In addition, the following conditions must be met:

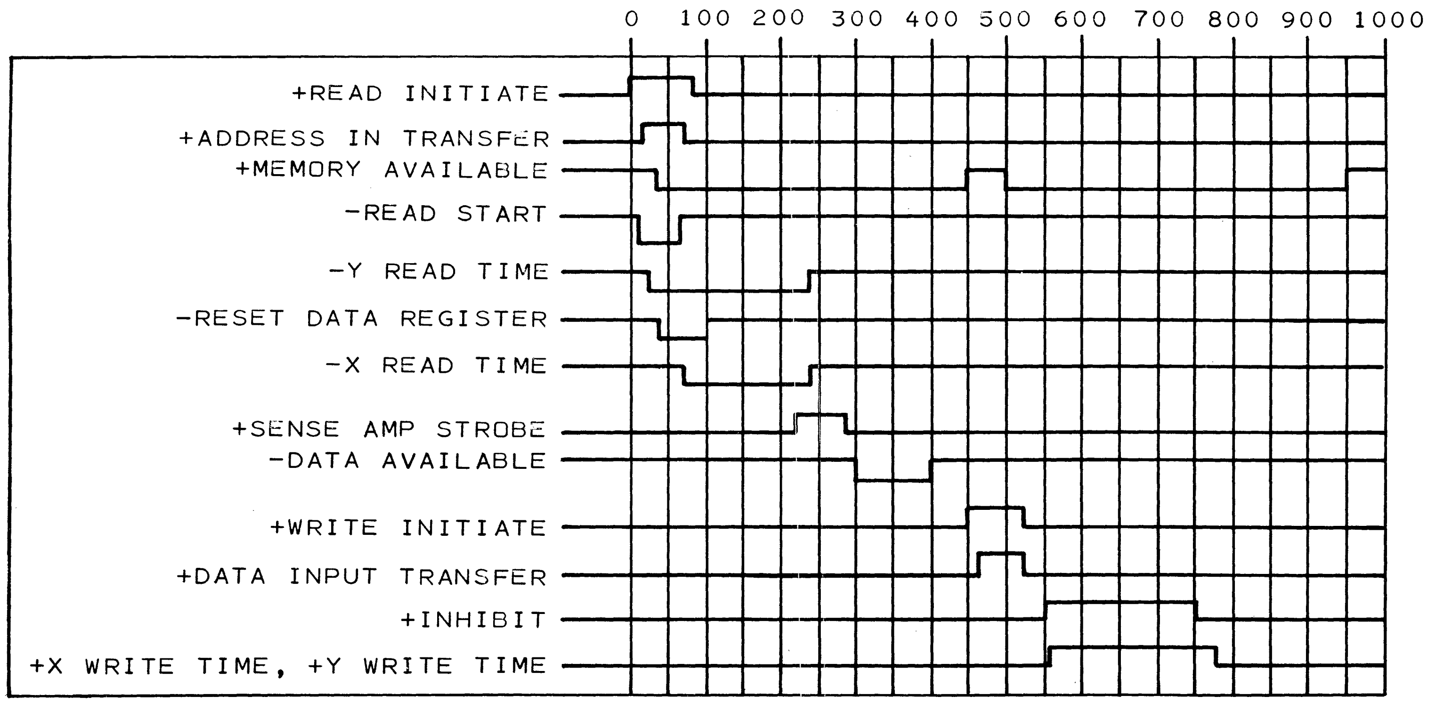


FIGURE III-3-4. INTERNAL TIMING - READ/MODIFY/WRITE

TABLE III-3-3. READ/MODIFY/WRITE (Continued)

STEP NUMBER	ACTION
	+Write Initiate = 0 +Read Only = 0 +Full Cycle = 0 Address Input Stable
	RD INIT must remain true for 75 ns minimum. WT INIT, RD ONLY, FC, and the Address Input must remain in the correct state for 100 ns minimum.
2	Address information is clocked into the address register by the AIX signal.
3	Output signal MA goes false 50 ns maximum after T , and at the same time the command inputs are blocked so as not to accept further instructions.
4	READ START/ initiates Y READ TIME/.
5	Timing signal YRT/ is sent to the address decoding circuitry, resulting in the generation of Y read current.
6	Data registers are reset by timing signal RDR.
7	Timing signals (XRT1 and XRT2/) are sent to the address decoding circuitry, resulting in the generation of X read current. Core turnover occurs at the core receiving coincident X and Y currents. This signal reaches the sense amplifier.
8	Timing signal SAS is sent to the sense amplifiers, resulting in sense amplifier output, which sets the data register in accordance with the data read from memory.
9	Output signal DA/ goes low at data access time and remains in that state for 100 ns (±25) indicating that the requested information is now available and stable on the data output lines. These lines retain the data read out of memory until the next cycle is initiated or until MS goes low.

TABLE III-3-3. READ/MODIFY/WRITE (Continued)

STEP NUMBER	ACTION
10	RD INIT must be returned to the false (0) state before the end of the read portion of the cycle.
11	Output signal MA goes true, indicating that the read portion of the cycle is completed, and that the write portion may be initiated. At the same time, the command inputs are gated so as to accept further instructions.
12	The write portion of the cycle is initiated by WRITE INITIATE = 1 which may not go true earlier than 475 ns. In addition, the following conditions must be met: READ INITIATE = 0 FULL CYCLE = 0 READ ONLY = 0 Data Input must be stable at 450 ns, and remain stable for 150 ns minimum. WT INIT must remain true for 75 ns minimum. RD INIT, FC, RD ONLY, and Address Information must remain in the correct state for 100 ns minimum after the leading edge of WT INIT.
13	Input data is clocked into the data register by timing signal DIX. Input data must be stable at this time and must remain so for 100 ns minimum thereafter.
14	Output signal MA goes false, and at the same time the command inputs are blocked so as not to accept further instructions.
15	Timing signal TINH is sent to the inhibit drivers, resulting in the generation of inhibit current through the cores corresponding to all data bits for which a zero is to be stored.
16	Write timing signals (WT1 and WT2/) are sent to the address decoding circuitry, resulting in the generation of X and Y write currents.

TABLE III-3-3. READ/MODIFY/WRITE (Continued)

STEP NUMBER	ACTION
17	Output signal MA goes true at 950 (±15) nanoseconds under the minimum time cycle conditions, indicating that the write portion of the cycle is completed, and that a new cycle may be initiated. At the same time, the command inputs are gated so as to accept further instructions.

Read Only

The Read Only (Half Cycle) operation is used to read a word from memory when it is not necessary to restore the information. The operation is actually a 'Read and Clear' operation, leaving an all zeros word stored at the accessed address location.

Table III-3-4 shows the sequence of events for the Read Only operation. It is suggested that the system block diagram, Figure III-3-1, be used as a reference aid in reading this table. Figure III-3-5 shows internal timing for Read Only.

TABLE III-3-4. READ ONLY

STEP NUMBER	ACTION
1	The cycle is initiated by READ INITIATE = 1 which defines zero time. In addition, the following conditions must be met: READ ONLY = 1 WRITE INITIATE = 0 FULL CYCLE = 0 Address Input must be stable. RD INIT must remain true for 75 ns minimum. WT INIT, FC and Address information must remain in the correct state for 100 ns minimum. READ ONLY must remain true for 100 ns minimum.

TABLE III-3-4. READ ONLY (Continued)

STEP NUMBER	ACTION
2	Address information is clocked into the address register by the signal AIX.
3	Output signal MA goes false, indicating a cycle in progress, and at the same time the command inputs are blocked so as not to accept further instructions.
4	Timing signal YRT/ is sent to the address decoding circuitry, resulting in the generation of Y READ CURRENT.
5	Data register is reset by timing signal RDR.
6	Timing signals XRT1 and XRT2/) are sent to the decoding circuitry, resulting in the generation of X read current. Core turnover occurs at the core receiving coincident X and Y currents. This signal reaches the sense amplifier.
7	Timing signal SAS is sent to the sense amplifiers, resulting in sense amplifier output, which sets the data registers in accordance with the data read from memory.
8	Output signal DA/ goes low at data access time, indicating that the requested information is now available and stable on the data output lines. These lines retain the data read out of memory until the next cycle is initiated, or until MS goes low.
9	RD INIT must be returned to the false (0) state before the end of the cycle.
10	Output signal MA goes true, indicating that a a new cycle may be initiated. At the same time, the command inputs are gated so as to accept further instructions.

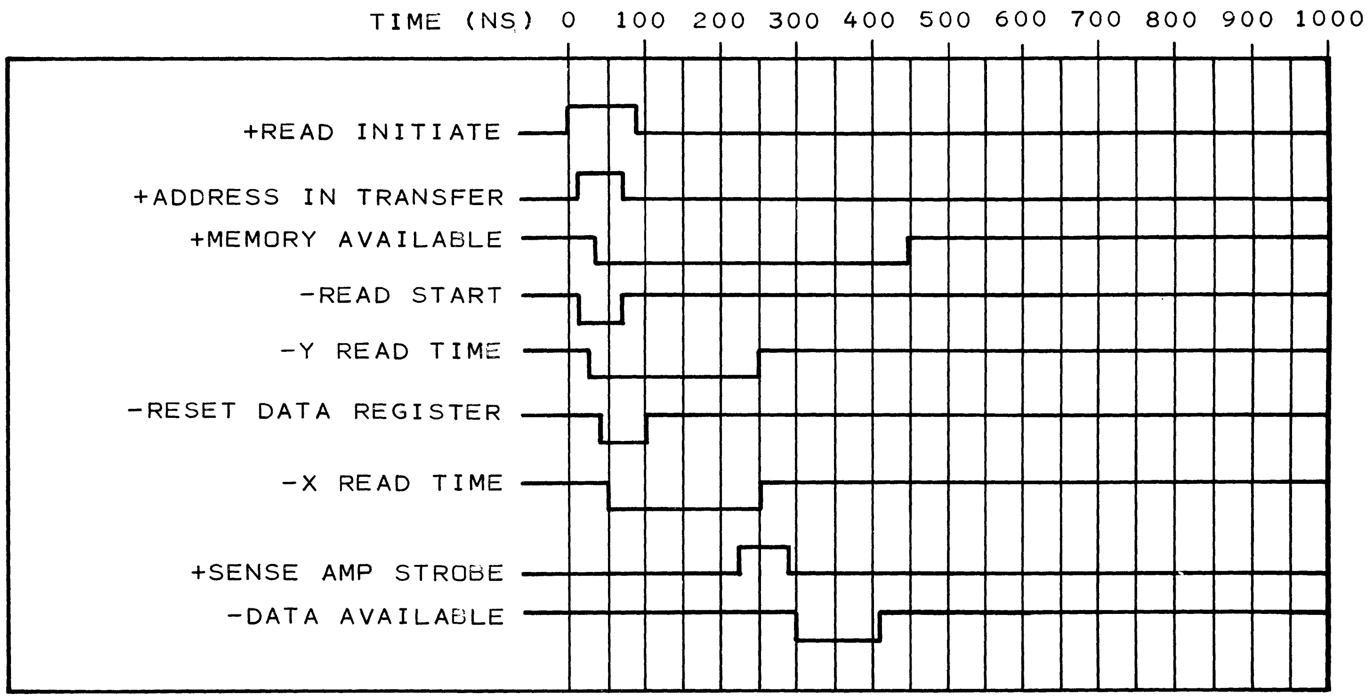


FIGURE III-3-5. INTERNAL TIMING - READ ONLY

Section 4
Circuit Description

4.0 INTRODUCTION

This section contains individual circuit descriptions of the blocks discussed in Section 3.

4.1 MEMORY SWITCH ASSEMBLY (MSA)

The MAS board contains:

The timing and control logic;

The address register;

The general reset circuit;

The address decoding circuitry;

The X and Y switches;

The X and Y current sources;

A positive voltage regulator;

Under voltage detector circuit.

Timing and Control Logic

The main features of the timing and control logic are shown in Figure III-4-1. The timing relationships are detailed in Figure III-4-2.

Cycles are initiated by the command inputs READ INITIATE or WRITE INITIATE, in conjunction with MEMORY SELECT. These inputs cause the Busy flip-flop to be set and the MEMORY AVAILABLE signal to go false. In addition, READ INITIATE and WRITE INITIATE are subsequently OR'ed to form AIX, which is used to clock FULL CYCLE and READ ONLY into their respective flip-flops. The inversions of READ INITIATE and WRITE INITIATE are used to set and clear the Read/Write flip flop.

During the time that the Busy flip-flop is set, the oscillator is running, sending clock pulses to the feedback shift registers.

These registers go through a sequence of states and the outputs from the flip-flops are connected to gating circuits to produce the control and timing signals required. At the end of the active cycle, the Busy flip-flop is reset. A one-shot is used to hold MEMORY AVAILABLE low until the memory cycle in progress is complete.

The functions of the oscillator and the feedback shift registers are detailed below.

The Oscillator

Two clock signals are produced from the oscillator, clock 1 and clock 2. Clock 2 is 20 ns out of phase. Clock 1 is used to produce T40, 80, 120, and 160, while clock 2 produces T60, 100, 140 and 180.

When the oscillator input is low (i.e. when the Busy flip-flop is not set) the output of U1 (Figure III-4-2) remains high, and the oscillator output remains high. When the input is high (i.e. when the Busy flip-flop is set) the circuit oscillates, producing an approximate square wave and its inversion, which serve as the timing clocks for the shift registers. The frequency of this signal is determined by the time delays through U1 and U2, and the values of components R and C. In this system, the oscillator has a period of about 40 ns.

The Feedback Shift Registers

Timing is generated by use of D-type flip-flops which are connected as a feedback shift register. Figure III-4-1 shows detailed working of the shift register, while Figure III-4-2 shows the timing relationships.

The outputs of the oscillator go to the clock inputs (C) of the shift registers. The complement side of the last stage (1DQ, 2DQ) is connected to the input of the first stage (1AD, 2AD).

At the start of each cycle, the Q side of register #2 contains all zeros. When the first clock pulse is recieved, a one is gated into the Q side of stage 2A, and the other stages have zeros. At the second clock pulse, a one is gated into the Q side of stage 2B, so that 2A and 2B now contain ones. Subsequent clockings fill register with ones, from left to right until Q = 1 for all stages. The high output from 2DQ resets 2AQ (at T180) to zero, and the Q side of the register then begins to fill with zeros from left to right.

Register #1 is set and reset in the same way, except that the timing inputs are 20 nanoseconds out of phase with the timing

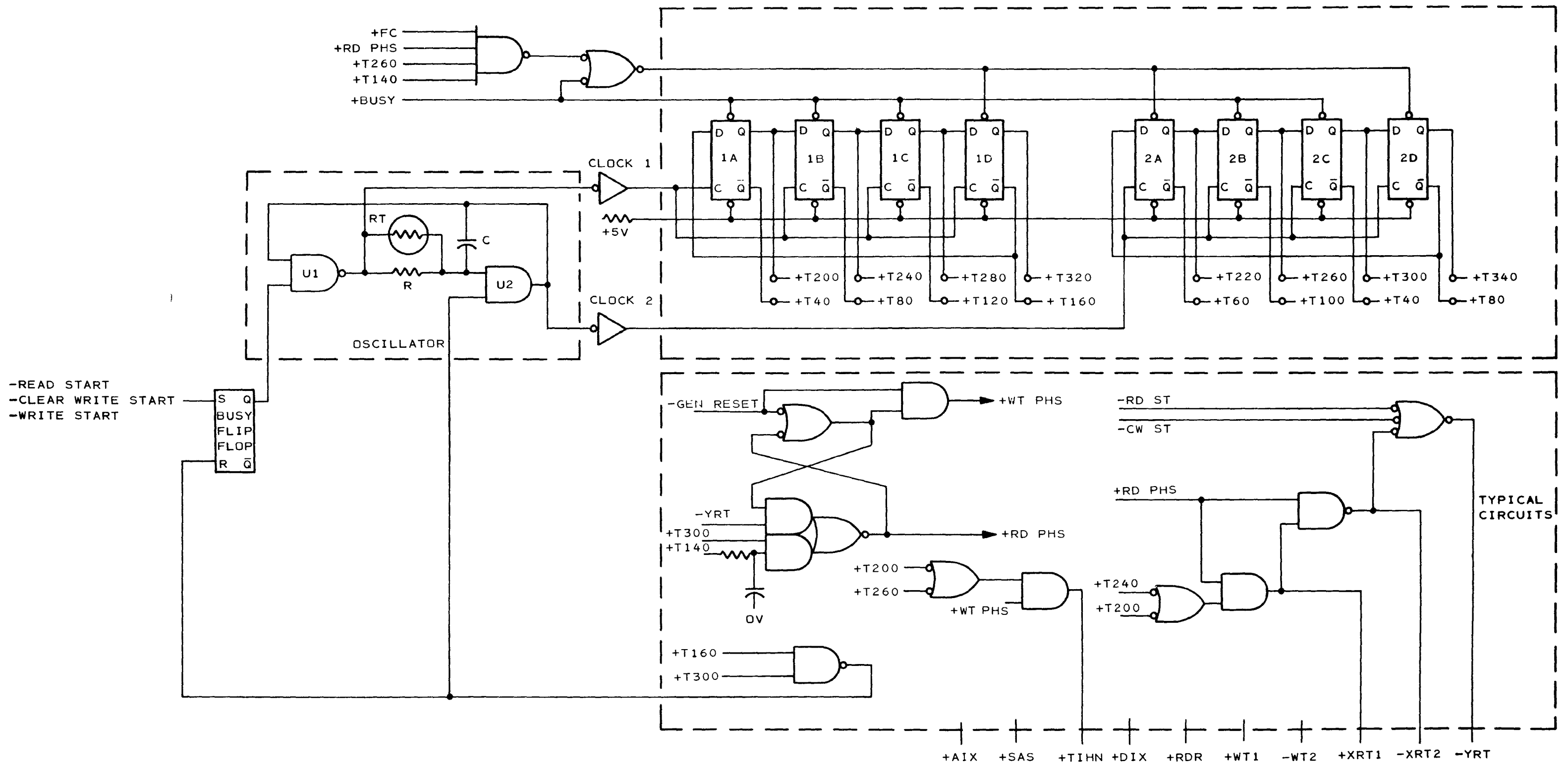


FIGURE III-4-1. TIMING AND CONTROL LOGIC

FAIRCHILD
SYSTEMS TECHNOLOGY

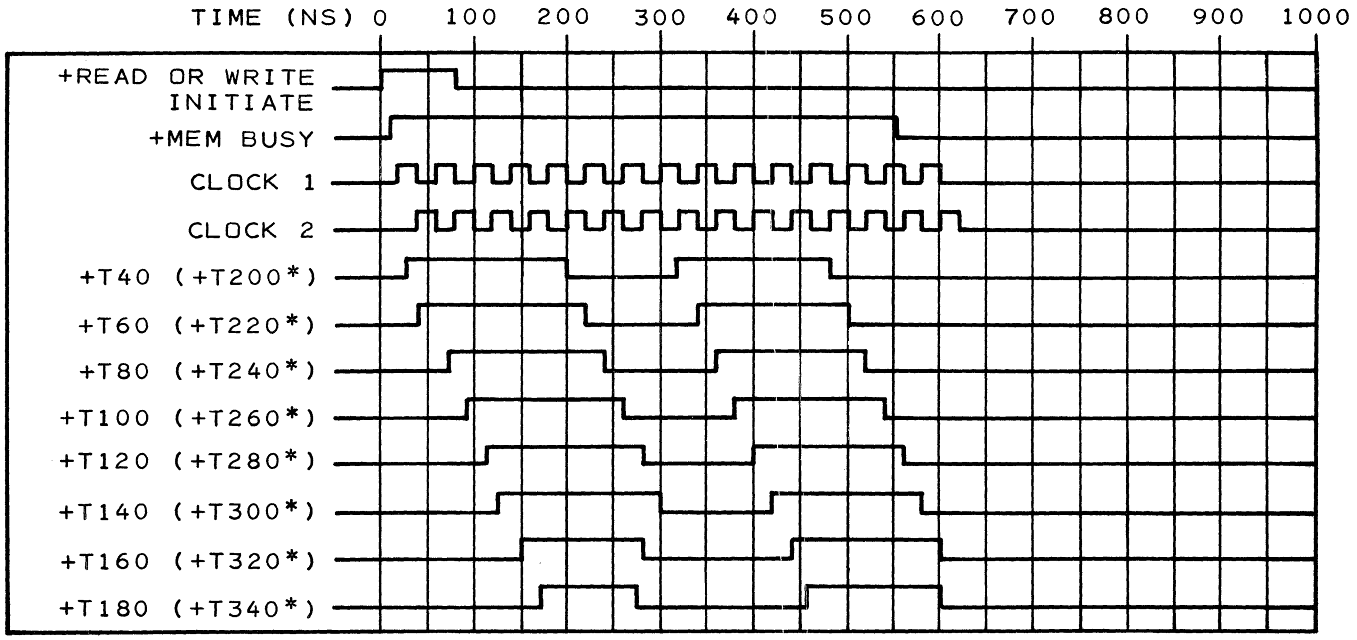


FIGURE III-4-2. INTERNAL TIMING RELATIONSHIP

inputs of register #2. Thus the overall register can produce a series of timing pulses, 160 nanoseconds wide, staggered at 20 nanosecond intervals.

During a full cycle, the Q side of the register, after initially filling with ones from left to right, fills with zeros from left to right, until the Q side of stage 2B = 1, while the Q side of stage 2C is in the zero state. The momentary coincidence of these conditions (T260 and T140) is used by the reset gates to preset the Q side of stages 1D, 2A and 2D to the 1 state. Therefore, 2DQ is 0, so that the reset of Busy is inhibited and the clock continues counting to complete the full cycle. This portion of the operation will be referred to as Procedure A.

The Q side of the register fills up once more with '1's, and then with '0's, from left to right. At the moment the Q side of stage 2C = 1, and the Q side of stage D is still 1 (T160 and T300), the Busy flip-flop is reset. This supplies the additional reset input to all the stages to ensure that the Q side of the register is reset to the zero state, and a further cycle may be initiated. This portion of the operation will be referred to as Procedure B.

During a split cycle, a RD INIT signal is received, and the register goes through Procedure B. When the Q side of the register has been cleared to zero, a WT INIT signal is received, and Procedure B is repeated.

The timing signals sent to the memory electronics are generated from the different register outputs. Where necessary, the signals are AND'ed with RD PHS or WT PHS, so that these pulses will occur in the correct half of the cycle. For example, WT PHS is AND'ed with T200 to form TINH (INHIBIT).

DATA AVAILABLE/is formed from a timing signal, using a one-shot to generate a 100 nanosecond negative-going pulse, beginning approximately 300 nanoseconds after initiation of the read cycle.

Address Register

Thirteen address bits are required for a single 8K word memory module. The bits are numbered 00-12. The address register functions as an open latch for address bits 7-12, one of which is represented in Figure III-4-3. Refering to address bit 7 as typical, when the memory is not ubsy, the Cycle Busy flip-flop is cleared. If the memory cycle is initiated, and PA7 goes high, then MAR7 goes low, and 40 nanoseconds later T200 goes low, and the Cycle Busy flip-flop is set, allowing the inputs to the cross-coupled gates to regenerate and store a '1' for address bit 7. One gate delay after the '1' side of the Cycle Busy flip-flop goes high, the '0' side goes low, disabling any further input from the CPU. At the end of the cycle, CYCLE END/goes low, clearing the Cycle Busy flip-flop and enabling address bit input again.

Address bits 0-6 are gated from the CPU into D-type flip-flops by the positive-going edge of AIX. For bits 0-5, outputs from the low side of the flip-flop (e.g. MAR5) are used for decoding. In the case of address bit 6 both high and low outputs from the flip-flop are used.

General Reset Circuit

Figure III-4-4 is a simplified diagram of the general reset circuit. This circuit resets most of the control flip-flops (e.g. Busy, etc.) when +5VDC power first comes up in the system. It also prevents accidental loss of data in the event of power failure providing an orderly shut-down sequence and ensuring that the cycle in progress is properly completed.

When +5V is applied the inputs to U1 are momentarily held low by the RC network, R1, C1. The gate is thus disabled, so that as +5V comes up, base current passes through R3 and R4, which turns on Q1, to produce GENERAL RESET/(i.e. GR/goes low).

Presuming that POWER READY is high (i.e. there is no power failure), R1 and C1 charge up, causing the output of U1 to go low, turning off Q1, and GENERAL RESET/goes high, enabling the control flip flops.

If POWER READY goes low, and Busy is low, the output of U1 goes high, causing GENERAL RESET/to go low. However, if POWER READY goes low during a memory cycle, when Busy is high, the output of U1 stays low until Busy goes low, and then a GENERAL RESET/signal is issued. Another memory cycle cannot be issued until POWER READY goes high again.

Address Decoding Circuitry and X and Y Switches

Address bits 0-12 are used to control the X and Y switches; they are assigned as shown below:

Address Bits

0-2	3-6	7-9	10-12
X DRIVE	X SINK	Y DRIVE	Y SINK

The X and Y switches in turn direct current through the X and Y lines in the core stack, selecting one core out of the 8192 (8K) in each mat.

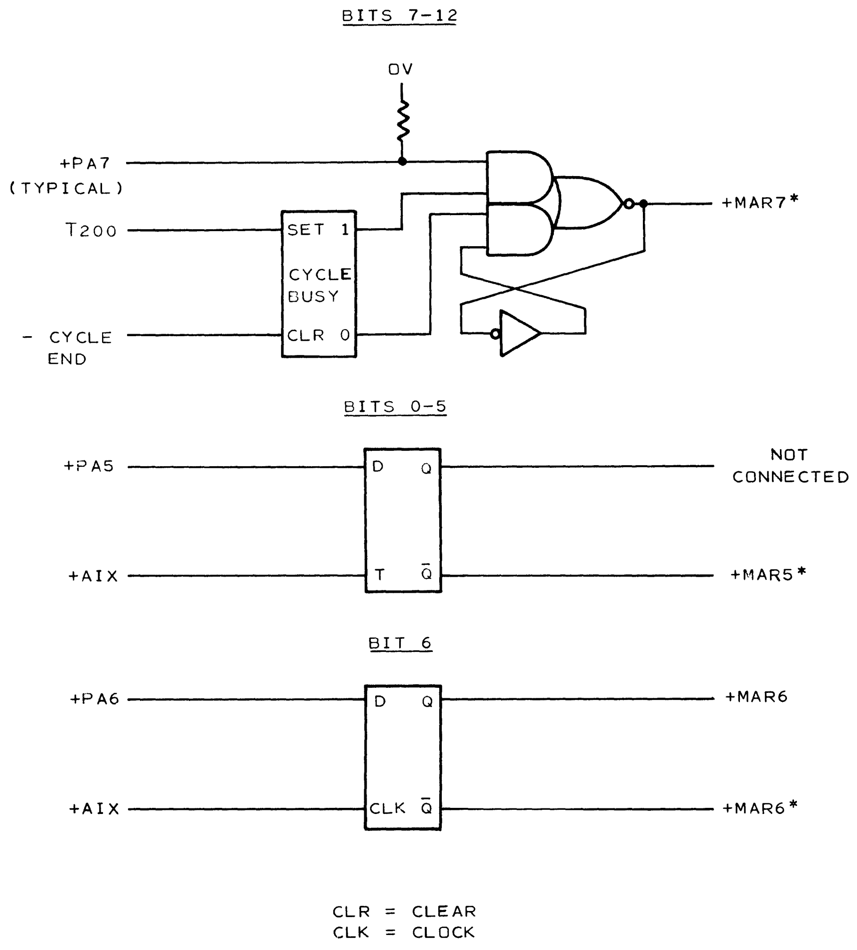


FIGURE III-4-3. ADDRESS REGISTER

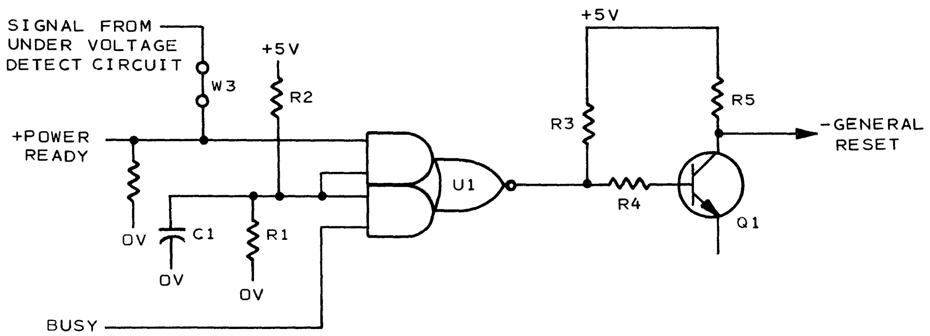


FIGURE III-4-4. GENERAL RESET CIRCUIT

NOTE:

There are 12 mats per core, stack, each mat corresponding to one word.

Figure III-4-5 shows how these 13 bits actually determine which X and Y lines are driven. The boxes marked A, B, C and D represent TTL decoding logic driving a set of switching transistors. The overall effect is that for any combination of ones and zeros for the 3 (or 4) address bit inputs, one and only one of the 8 (or 16) output lines is connected to the current source indicated.

Any combination of the 13 address bits then turns on two X switches (one in block A and one in block B), and two Y switches (one in block C and one in block D).

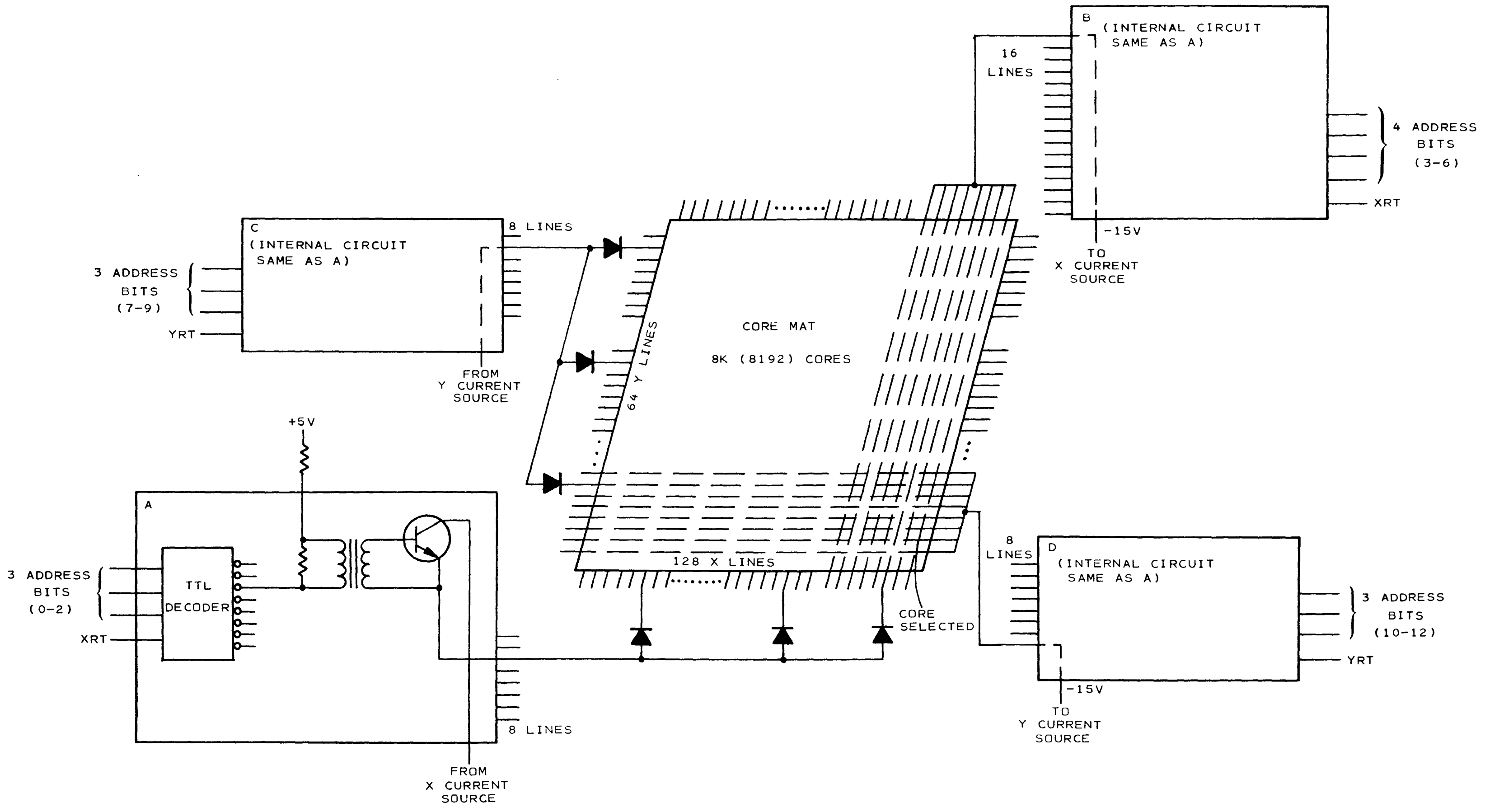
In this manner two complete circuits are formed. One proceeds from the positive leg of the X current source, through a switch in block A, through an X line in the core stack, then through a switch in block B, and finally, into the negative leg of the X current source. Note that only one line is connected to a current source at both ends.

In a similar manner, the second complete circuit proceeds from the positive leg of the Y current source, through a switch in block C, through a Y line in the core stack, then through a switch in block D, and finally, into the negative leg of the Y current source.

A single core (in each mat) lies at the intersection of the X and Y lines driven, and thus receives the coincident current.

For simplicity, Figure III-4-5 shows only the current paths for the read operation. The blocks A, B, C, and D also contain similar circuitry for the write operation, which differs mainly in that the write current is driven in the opposite direction. Thus for instance, block A, if shown more completely, would have an additional set of 8 lines, which would be connected by another set of switches (write switches) to the negative (-15V) leg of the X current source, and these additional 8 lines would connect to the same points as the 8 lines shown, but through diodes installed with opposite polarity.

The complementary action of the read and write circuitry is further explained on the following page.



EACH LINE FROM A CONNECTS (THRU DIODES) TO ONE X LINE FROM EACH OF THE 16 GROUPS
EACH LINE FROM B CONNECTS TO ALL 8 X LINES FROM A SINGLE ONE OF THE 16 GROUPS
} 128 X LINES —— 16 GROUPS OF 8 EACH

EACH LINE FROM C CONNECTS (THRU DIODES) TO ONE Y LINE FROM EACH OF THE 8 GROUPS
EACH LINE FROM D CONNECTS TO ALL 8 Y LINES FROM A SINGLE ONE OF THE 8 GROUPS
} 64 Y LINES —— 8 GROUPS OF 8 EACH

FIGURE III-4-5. ADDRESSING DECODING

X and Y Current Sources

Figure III-4-6 shows how the X and Y current sources are arranged. This figure could represent the drive of either an X line or a Y line. The switches, selected and controlled by the decoding circuitry described in the previous section, are represented simply by the symbol with the arrow pointing in the direction of the current flow.

During any given operation either a read switch pair (RS1, RS2) or a write switch pair (WS1, WS2) is turned on, thus providing read current (RC) or write current (WC), as shown.

Each current source connected to the regulated voltage supply consists of 2 series resistors, and two by-pass capacitors. The balun transformer, connected as shown, ensures that the current entering the stack is equivalent to the current leaving the stack.

Voltage Regulator

The temperature compensated positive voltage regulator supplies approximately +2V needed for the X and Y current sources.

The voltage regulator consists of an operational amplifier and related circuitry. Figure III-4-7 shows a simplified version of the amplifier function.

Amplifier gain (G) is defined as follows:

$$G = R4 \quad \frac{RT1 + R3 + R2}{R2}$$

The output of the regulator is determined by the level of the reference voltage and by the amplifier gain. The values of R2, R3 and R4 have been chosen in relation to RT1 to give a nominal output voltage of 2.8 volts at the mid-range value of R6 (at 25 C), while also setting the required temperature compensation.

The reference voltage is adjusted by resistor R6 to give an output voltage of +3.0 Volts (with temperature compensation).

The current capability of the voltage regulator must be sufficient to supply 600 milliamperes of current to the switches (300 ma for the X switches, and 300 ma for the Y switches). A pass transistor, Q1, is added to the output line to provide sufficient power amplification.

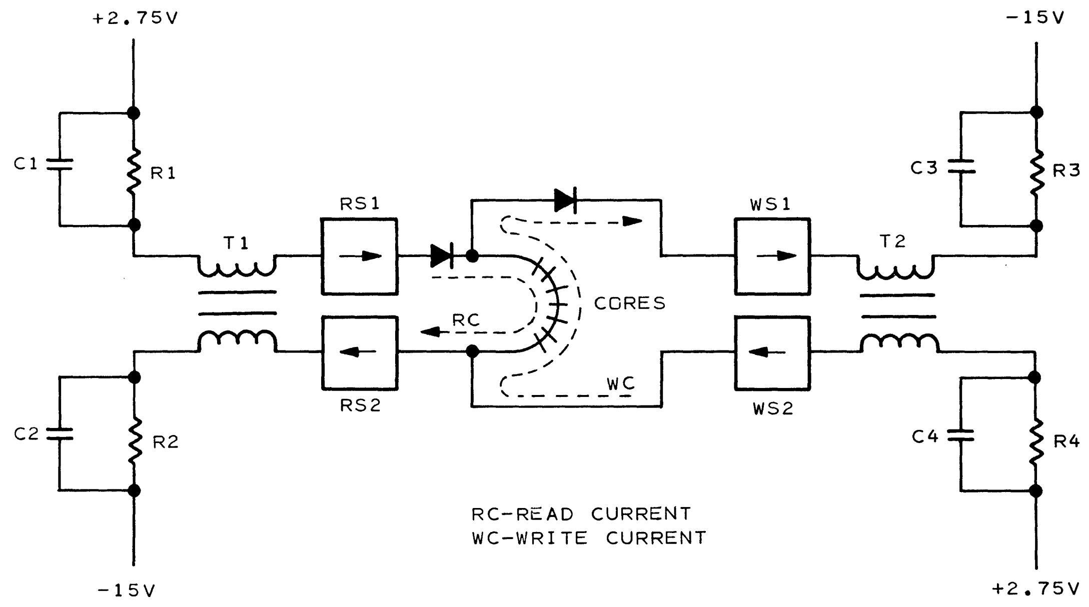


FIGURE III-4-6. SIMPLIFIED VIEW OF CURRENT SOURCES

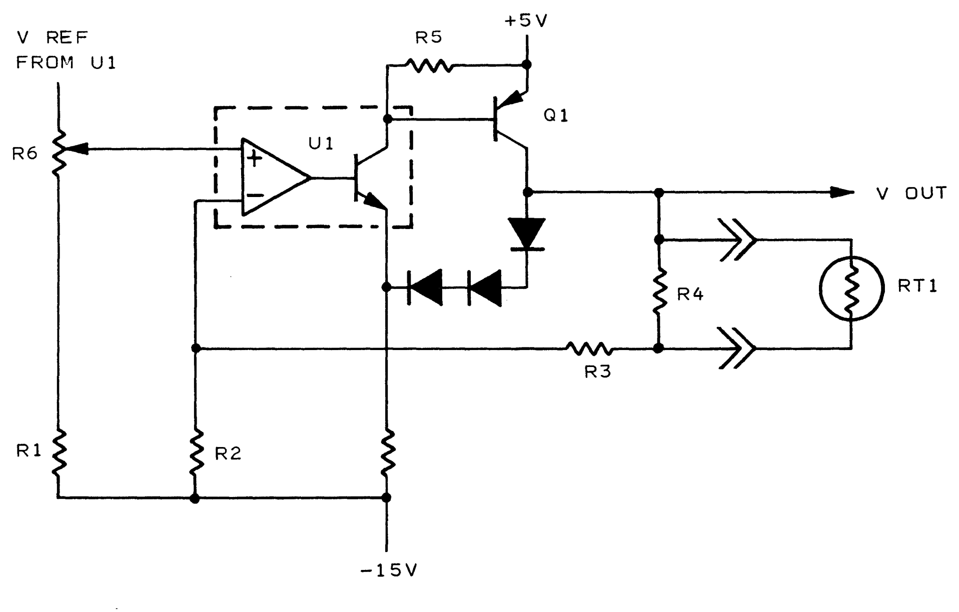


FIGURE III-4-7. VOLTAGE REGULATOR

It should be noted that the voltage regulator circuit is referenced to -15 Volts so that the +3.0 Volt output will follow and compensate for any variations in -15 Volts.

Under Voltage Detect Circuit

This circuit, detects an under voltage condition for either +5V or -15V and hence prevents loss of data, which would occur if the memory continued to operate with unacceptably low magnitude supply voltages.

The buffered output of the detect circuit is the internally generated POWER READY signal which, together with BUSY controls the general reset circuit.

The connection of an externally generated POWER READY signal remains optional if external power failure control for the BSM is desired.

4.1.1 Memory Magnetic Assembly

The MMA contains:

The core array;

Diode matrices;

Sense circuitry.

MMA Diode Matrices and Core Array

The X diode matrix consists of 128 read diodes and 128 write diodes. The X read and write diodes are installed with opposite polarities. The read and write diodes are arranged in 8 diode groups of 16 each, and make it possible to drive a single X line when the appropriate X-switch pair is turned on.

The Y diode matrix, similarly, consists of 64 read diodes and 64 write diodes, arranged in 8 groups of eight each, enabling a single Y line to be driven when the appropriate Y-switch pair is turned on.

On each core mat, a single core lies at intersection of the one X line and one Y line being driven. This core, which represents one bit of information, receives coincident currents and is thus selected as the address location at which the operation is being performed. The core array consists of 12 identical core mats with the X and Y lines strung through all 12 mats, each mat corresponding to one of the 12 data bits.

The SAS-1 signal enters the MBA and is buffered by U9. Sense amplifier output from the MMA goes to the S Out NAND gate (U2) then goes to the data register.

NOTE:

These circuit operations occur only during the read portions of system cycles.

In addition, SAS enters the MMA via the MBA from the MSA, where it has been gated with BSM SEL. SAS strobes the sense amplifier output via U1.

Inhibit Circuitry (See Figure III-4-8)

The inhibit circuitry is operative only during the write portions of system cycles (i.e. the restore portion of Read/Restore, and the write portions of Clear/Write and Read/Modify/Write).

When a '1' is to be written, coincident X and Y write currents alone are used to drive the cores, resulting in core turnover at the desired address location, and causing a '1' to be written. If a '0' is to be written, inhibit current is driven through the sense/inhibit line, in the opposite direction from the current in the Y line, and of amplitude approximately equal to that of the Y current. This causes the Y current to be algebraically neutralized. There is thus no core turnover, and the core is left in the '0' state.

When data is to be written into memory, the output of the data register is AND'ed with the logical product of TINH and BSM SEL via U8, to drive Q1 through T1. R1 is selected to pass 2 X Y current. This current is split by balun transformer T2, so that one half of the current goes down one side of the sense inhibit line, and the other half of the current goes down the second side of the sense/inhibit line. C1 is used to speed up the rise time of the inhibit current.

Data Register

The data register consists of 12 flip-flops, one for each system bit. Figure III-4-8 shows a simplified view of a typical data register flip-flop (U3, U4) and the related circuitry.

The operation of the register is described for one bit typical, and in terms of the Read/Restore and the Clear/Write operations.

Each mat is wired independently, with respect to its sense/inhibit line. A single sense/inhibit line threads through all 8K cores in one mat, running parallel to the Y lines within a bit. A bow-tie stringing arrangement is used to reduce half-select noise.

Thus, the X and Y lines carry the address information while the sense/inhibit lines carry the data. The following section explains the operation of the sense circuitry.

Sense Circuitry

A single sense/inhibit line threads through all the cores corresponding to one data bit. Since the sense circuitry is identical for all 12 bits, Figure III-4-8 illustrates circuit operation for one bit typical.

The sense circuitry is operative only when a read operation is to be performed (i.e. during the read portion of Read/Restore, Read/Modify/Write and Read Only).

During a read operation, the coincident X and Y select currents cause turnover of a single core if a '1' has been previously stored in the selected core. This turnover induces a voltage of about 30 millivolts across the ends of the sense/inhibit line. This voltage is fed into the inputs of a differential amplifier, U1, the sense amplifier. The output of U1 is a digital logic signal which is strobed to the MBA by timing signal SAS.

4.1.2 The Memory Bit Assembly

The MBA board contains:

The inhibit drivers;

The sense amp strobe buffer gate;

The sense out NAND gate;

The data register;

Zone control circuitry;

+5V Pull up circuitry.

SAS Buffer and S Out NAND Gate (See Figure III-4-8)

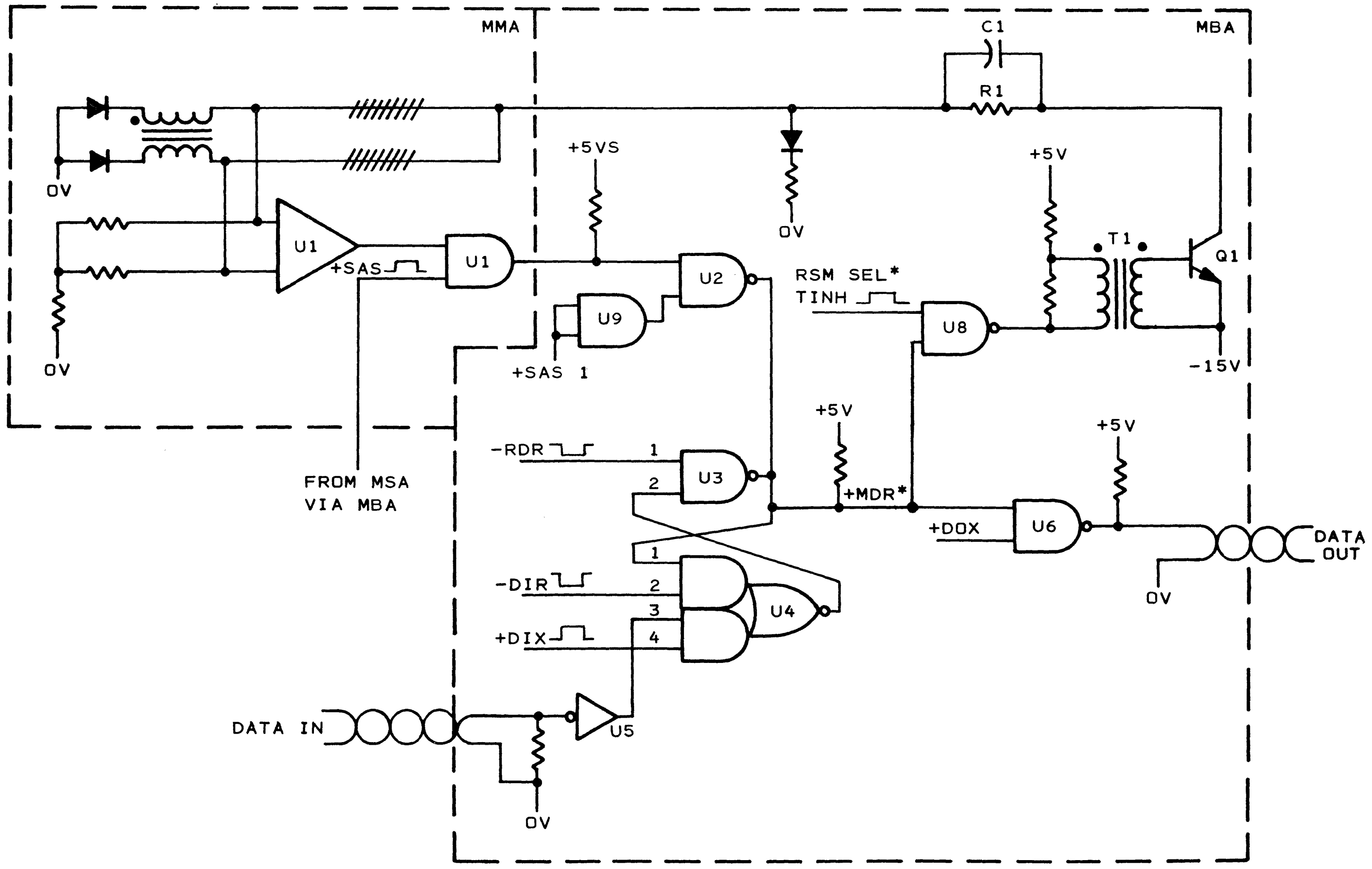


FIGURE III-4-8. SENSE, INHIBIT AND DATA CIRCUITRY

READ/RESTORE

At the beginning of a Read/Restore cycle, DIR/is high and DIX is low. RDR/goes low, to reset the data register, so that the output of U3 is high. The high output of U3 is applied to U4-1, which, with DIR high, makes the output of U4 low. This low output is applied to U3-2, to hold the output of U3 high.

If a '0' is read from core, then MDR (the buffered output from the sense amplifier) is high. Therefore, the data register does not change state, and so DATA INHIBIT is high. DATA INHIBIT high enables the inhibit driver on the MBA, provided that TINH is present. The enabling of the inhibit driver allows a '0' to be restored to core during the restore portion of the cycle.

When a '1' is read from core, +MDR goes low, and applied to U4-1 changing the state of the data register, so that the inhibit driver is disabled, and a '1' is restored. Data is gated out of the data register by TDOX via U6. DATA OUT is high when a '1' is read out of core, and low for a '0'.

CLEAR/WRITE

During Clear/Write there is no SAS signal, and therefore no MDR is received by the data register. DATA IN is used to set the data register to the required state prior to the write portion of the cycle.

If DATA IN is high, a DIR/signal present at U4-2 sets the data register so that the output of U4 is high. It should be noted that when DATA IN is high, the input of U4-3 is low, so that the presence of a DIX pulse has no effect on U4, and the state of the register remains as set by DIR. With the output of U4 high, MDR is low, which disables the inhibit driver. This means that during the write half of the cycle, a '1' is written into core.

If DATA IN is low, DIR/attempts to set the data register so that the output from U4 is high. However, in the meantime, DATA IN low is inverted via U5, so that the input at U4-3 is high. A high at U4-3 in conjunction with DIX high at U4-4 sets the register so that the output from U4 is low. This means that MDR is high, during the second half of the cycle, and the inhibit driver is enabled at TINH time, to write an '0' into core.

NOTE:

DIX must extend beyond the duration of DIR.

Zone Control Circuitry

Figure III-4-9 shows the zone control circuitry for 1 zone typical. The zone control feature is operative during Clear/Write and Read/Modify/Write cycles, and allows selective clear/write operations and permits the generation of SAS-1 (or SAS-2) as required. These conditions allow a read/restore type of operation to be performed for the zone not selected for Clear/Write or Read/Modify/Write.

When a Read/Restore operation is being performed, zone control is not functional. RR/CW/must be high (it is set high by READ INITIATE) and ZW1 may be high or low. (See Figure 6-24). When RR/CW/is high, U4-2 and U3-2 are low, and U4-3 is high and U3-3 low. When U3-3 is low, the DIX pulse is not gated, and DIR/is held high, while DIX is held low. With U4-3 high, RDR is gated to allow the generation of RDR-1/(RDR-2). DOX is gated and SAS is gated to allow the generation of SAS-1 (or SAS-2). These conditions enable a Read/Restore operation to be performed.

4.2 SIGNAL NAMES AND ABBREVIATIONS

This section contains a list of the signals used in this manual, and in the schematics. The polarities of the signals (/ = normal LO active state) are indicated and, where appropriate, the abbreviations commonly used.

AIX	ADDRESS INPUT TRANSFER
BSM SEL	BASIC STORAGE MODULE SELECT
BUSY	BUSY
CW ST/	CLEAR WRITE START/
CYCLE END/	CYCLE END/
CYCLE INITIATE	CYCLE INITIATE
DA/	DATA AVAILABLE/
DI	DATA INPUT
DO	DATA OUTPUT
DIR/	DATA INPUT REGISTER/
DIX	DATA INPUT TRANSFER
DOX	DATA OUTPUT TRANSFER
GEN RST/	GENERAL RESET/
MA	MEMORY AVAILABLE
MAR	MEMORY ADDRESS REGISTER
MDR	MEMORY DATA REGISTER BAR
PA	PROCESSOR ADDRESS

PWR READY	POWER READY
RD INIT	READ INITIATE
RD PHS	READ PHASE
RDR 1/	RESET DATA REGISTER 1/
RDR 2/	RESET DATA REGISTER 2/
RD ST/	READ START/
READ ONLY	READ ONLY
RDR	RESET DATA REGISTER

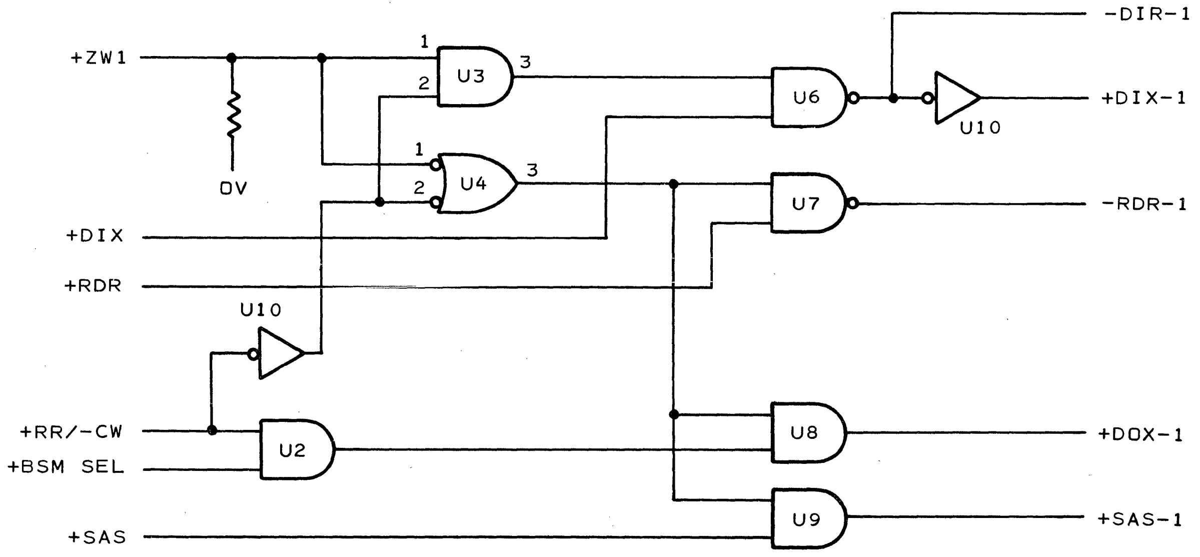


FIGURE III-4-9. ZONE CONTROL CIRCUITRY