FAIRCHILD

Test Systems Group

The Fairchild FST-2 Computer

Publication Number 57000002

November 1978 Edition

This publication is subject to replacement by a later edition. This publication is also subject to technical updating by a Publications Bulletin. The issuance of publication improvements by either a later edition or a bulletin is decided partly on the basis of the degree of change required. To determine whether a later edition or bulletins exist for a particular publication, consult your local branch office, or FTSG Publications Department, 1725 Technology Drive, San Jose, CA 95110 (TWX 910-338-0558).

To request copies of this publication, contact:

Fairchild
Test Systems Group
Customer Services
M/S 36-07/57
1725 Technology Drive
San Jose, California 95110
(408) 998-0123 TWX 910-338-0558

A reader's feedback form is provided at the back of this publication. Help us to help you by providing comments. If the form has been removed, please send comments to FTSG Publications Department.

The Fairchild FST-2 Computer

Publication Number 57000002

November 1978 Edition

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES (Continued)

LIST OF ILLUSTRATIONS (Continued)

LIST OF TABLES

LIST OF ILLUSTRATIONS (Continued)

PREFACE

This manual describes the Central Processing Unit (CPU) and Memory System of the FST-2 computer. The Common Peripheral Interface Unit (CPI) that contains the interface cards and controllers for the I/O devices is described in the CPI manual as are the interface requirements for the TTY/VKT and Line Printer. The interface cards for these items are physically located in the CPU.

REFERENCE DOCUMENTS

FST-2	Common Peripheral Interface Manual	Part #67095734
FST-2	CPU/Peripheral Diagnostics Manual	Part #67095731

SECTION 1

GENERAL INFORMATION

1.0 INTRODUCTION

This chapter provides a general description of the FST-2 computer. Included are general information, and information covering controls and indicators.

The computer consists of 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 1-1. The memory system is located in the A2 module, while the CPU and memory interface (data bus) boards are contained in the A1 module. The A3 module provides space for controllers for the I/O (input/output) peripheral devices.

In the Sentry systems supplied with an FST-2 computer, the common peripheral interface module (A3) is present and provides interface for I/O peripherals such as card readers, magnetic tape units, and disc storage units.

The interface cards for the TTY/VKT, Data Link, and line printer are contained within the CPU (A1) Module. Interface between the computer and tester are provided by 3 CPI cards located in the A1 module.

1.1 SYSTEM DESCRIPTION

The basic FST-2 system configuration is presented in Figure 1-2. The CPU, control panel, and memory are 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 transferred between the CPU and peripherals via the bidirectional N bus (also called the accumulator bus).

The program itself is stored in dynamic memory. Two memories are shown in Figure 1-2, designated A memory and B memory. A and B memory are both present in all Sentry systems and are organized as 8Kx25 bit memories - that is, they provide storage locations for 8192 words of 25 bits each. B memory has the same storage capacity as A memory. A memory contains information for even address, while B memory contains information for odd address. The minimum memory configuration for a CPU is 16Kx25 bits, that is, one board of A memory and one board of B memory.

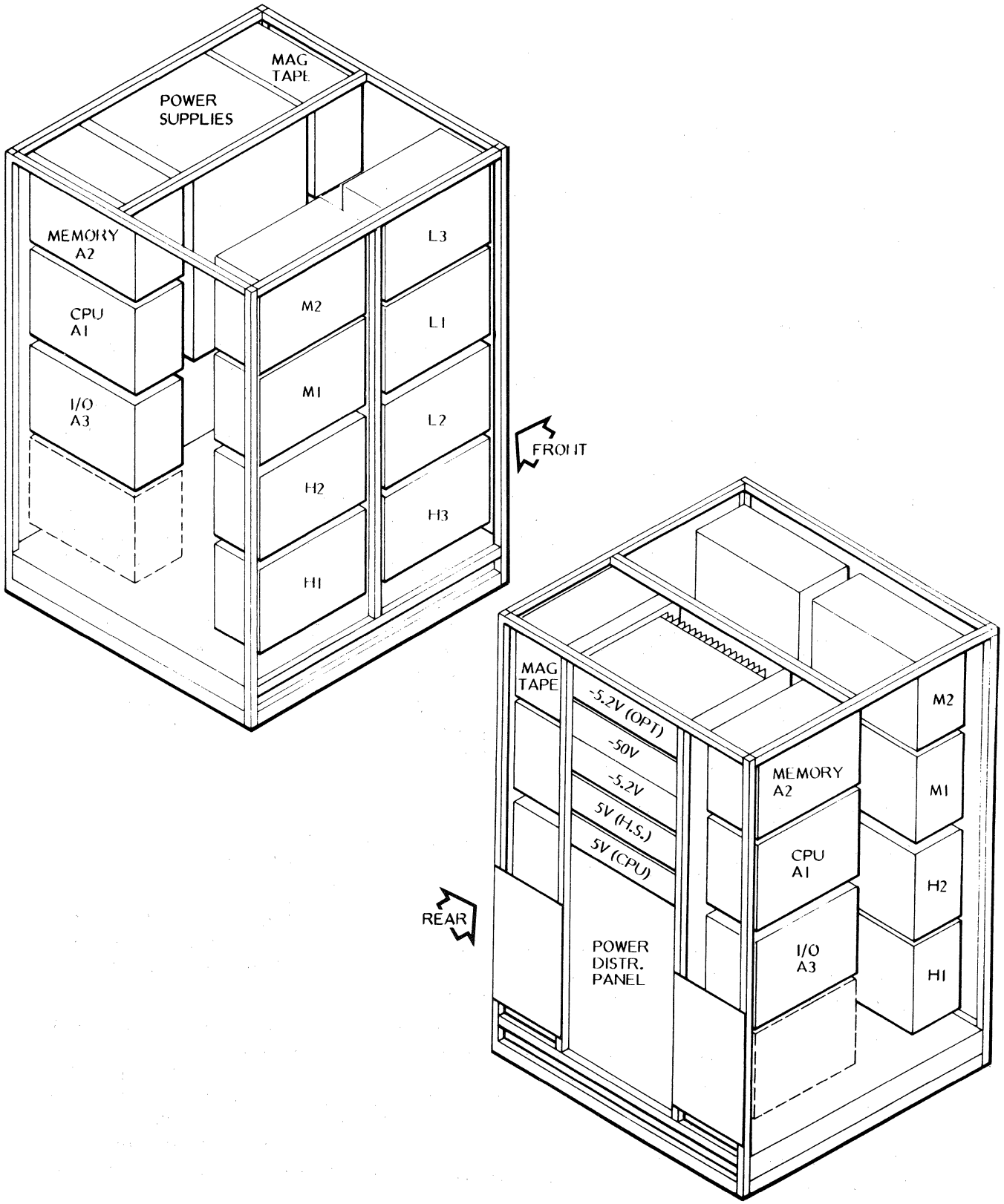


Figure 1-1. Module Locations A1, A2, and A3

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. Discs, mag tape, and card readers can interact with the computer by either of two following 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 writing 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, Data Link Controller, IEEE-488 Bus Controller, 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 memory, while the computer is performing data analysis on previous test measurements, and logging the results onto a disc.

Figure 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).

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, occurring in the relationship shown in Figure 1-3.

The basic timing circuitry is located on the CPU Clock board, slot D1 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.

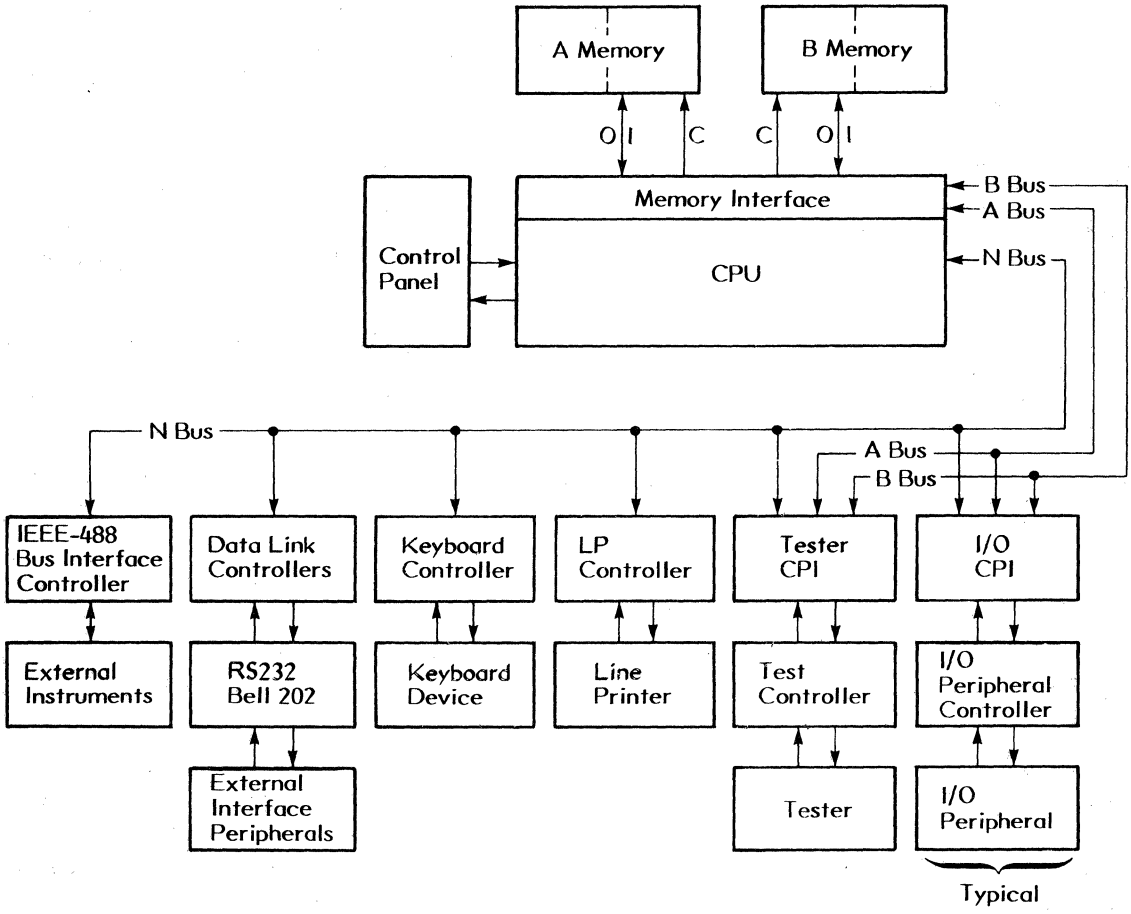


Figure 1-2. Basic FST-2 System Configuration

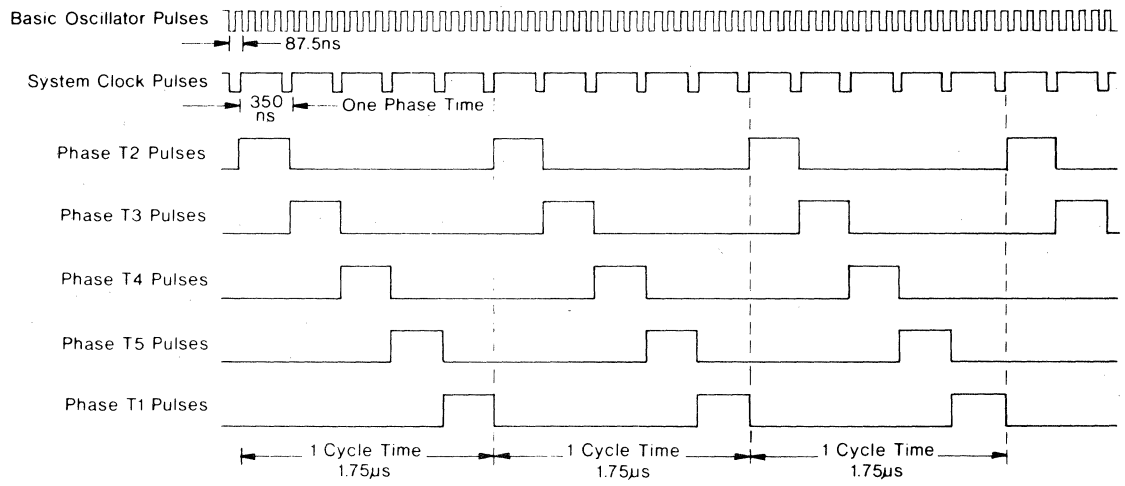


Figure 1-3. Basic FST-2 Timing Diagram

1.2 OPERATION AND MAJOR COMPONENTS

1.2.1 CPU

Figure 1-4 is a block diagram of the FST-2, 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 1-4. Specific CPU hardware information is contained in Section 2 of this manual.

Memory Interface

The memory interface unit comprises 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 dynamic memory and the program counter, the command register, and the buffer register. Also, for data exchanges between the dynamic memory and the peripheral units, the memory interface unit performs the same service, transferring data along the A and B memory busses between memory and the common-peripheral-interface modules of the peripheral units.

Program Counter (P)

The 18-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 000000_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 memory address of the current instruction to be fetched from memory.

Index Registers (X_0 through X_7)

There are eight 18-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.

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

Computer-program instruction words are transferred from 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 Section 3. The segments of the command register designated on Figure 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 Section 3.

The R bit at the instruction words determines whether the relocation register is to be selected.

The significance of the five "C" bits of instruction words are always the same: they represent the operational code in a binary coded octal, two digit number. These five 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 operand field can be expanded to 18 bits for memory reference instructions via the relocation 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 address of 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" bits designate the address of a memory location whence 18 instruction-word bits replace the "O," "I," and "X" bits in the command register.

Although not designated in Figure 1-4, the six least significant bits taken from the operand-address portion of the command register designate the number of bit-positions shifted during a shift command. In this case, these bits are transferred to the shift counter (CO)).

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.

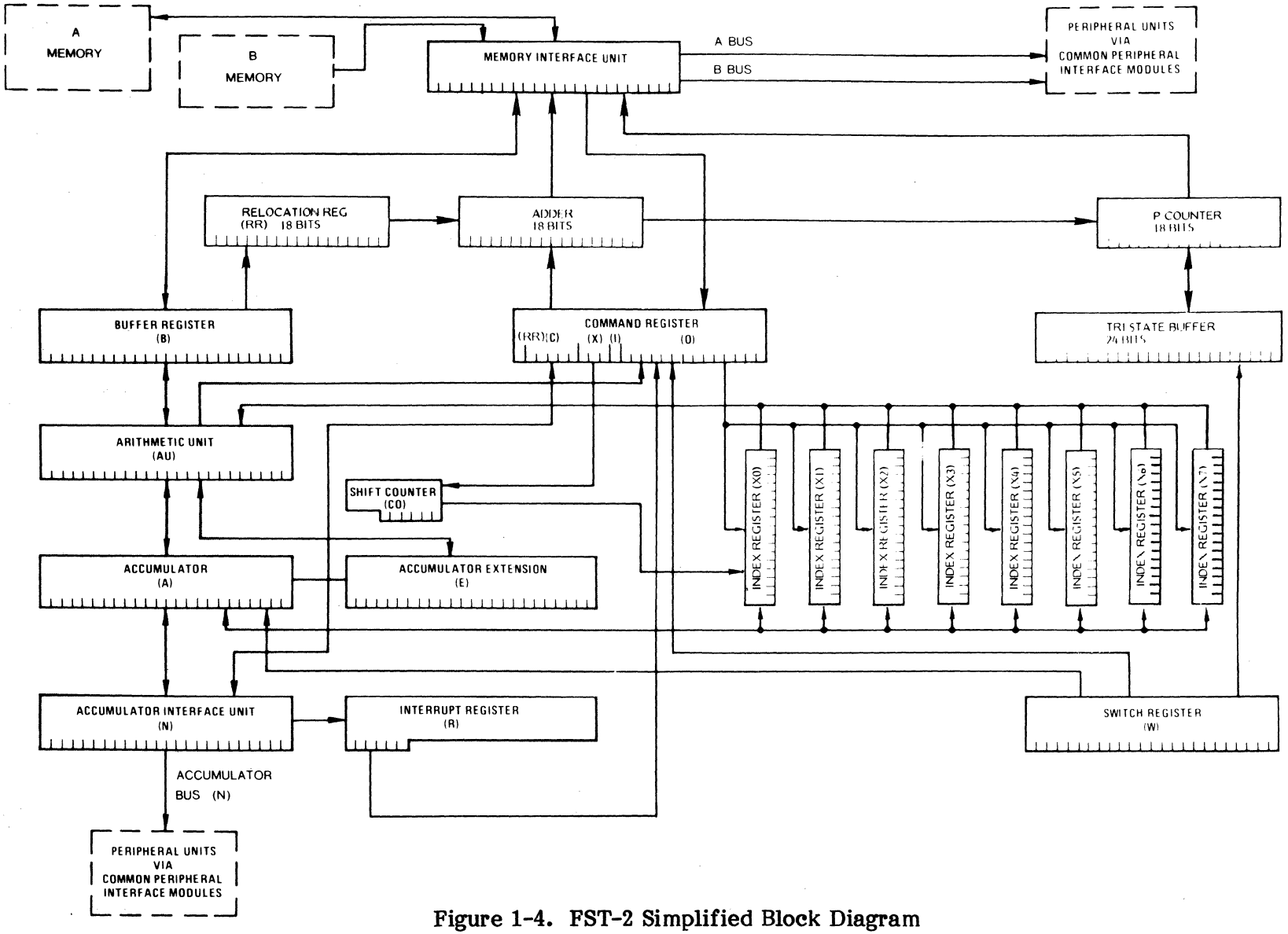


Figure 1-4. FST-2 Simplified Block Diagram

Relocation Register (RR)

This 18 bit register allows modification of the operand-address of memory reference instructions. Under control of the command register bit 23, it is either added to the address field or not. Thus it allows addressing any word in a memory of 2^{18} = 260K words.

The relocation register can be loaded and read out via the accumulator.

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 X0 after termination of the DSN shifting operation.

Buffer Register (B)

Figure 1-4 shows data flow from memory to buffer register and from the buffer register via the relocation register (RR) through the relocation adder (RA) to the memory interface unit. Data also flows 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 required 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.

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 or 18 bits of a data word may be transferred in either direction between the accumulator and any index register (X_0 through X_7). Refer to Section 4 for more details.

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 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 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 1-4.

Accumulator Interface Unit (N)

The accumulator interface unit comprises 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 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 B Boards C5, C7, C9), 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 bits, as indicated in Figure 1-5. The input bus lines shown as O, and B, come from the command register (O = operand address), and the buffer register, respectively. The program counter is located on the same board. Each board contains 8 bits of program counter. However, only bits 0-17 are being used. The command and buffer registers are located on the 12 Two-Bit Slice Boards and Memory Control B Board.

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 1-6 shows details of the memory interface gating functions for one bit. Specific hardware information on the Memory interface is contained in Section 2.

1.2.2 Memory

A memory and B memory are identical in size and configuration. One memory board contains 8K x 25 bit of memory (24 bit word and a parity bit). The basic configuration is 16K word memory expandable to 196K word by increments of 16K words.

Data In/Out	24 lines
Data In	1 line
Data Out	1 line
Address In	18 lines
SOC (Start of Cycle)	1 line
RAS (Row Address Select)	1 line
SWC (Start Write Cycle)	1 line
COLT (Column Timing)	1 line
REF (Refresh)	1 line

Data input and output lines use a bidirectional bus. However, the first 18 of the 25 data input lines are also used as 18 address lines, on a time-shared basis. This is possible because address information is entered at CPU time T3.

Specific hardware details in the memory are contained in Chapter 5.

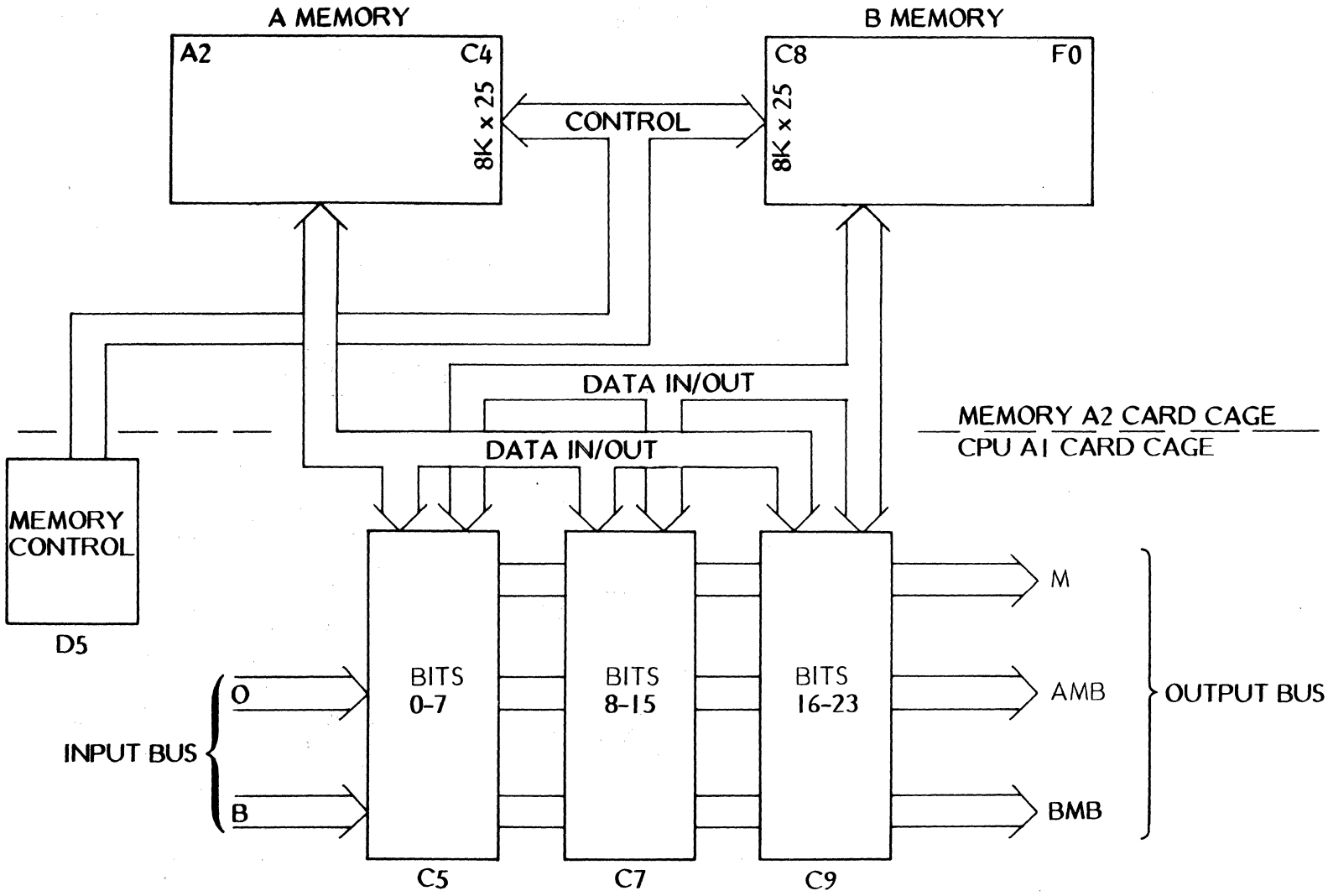


Figure 1-5. Memory Interface

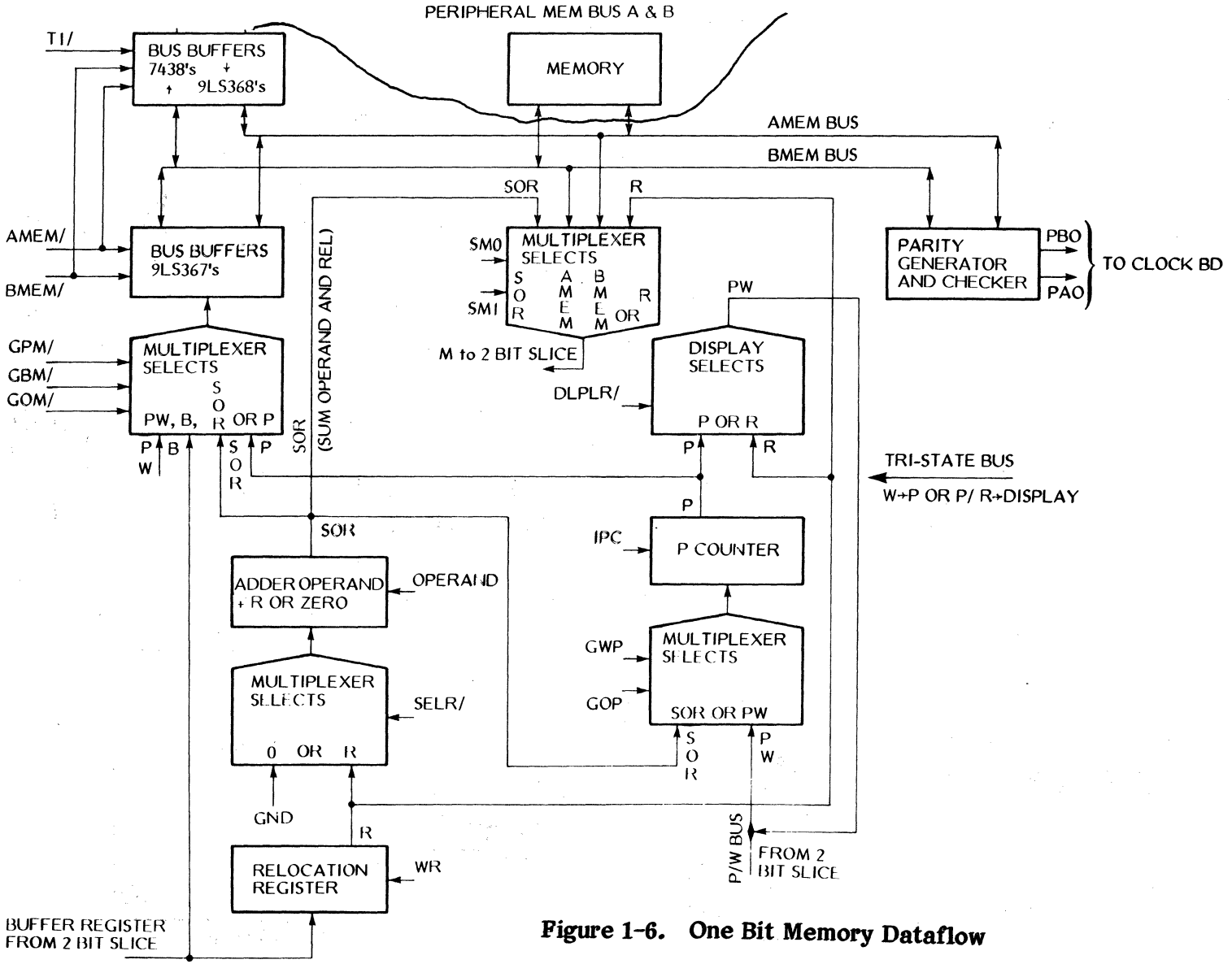


Figure 1-6. One Bit Memory Dataflow

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-E9. The TTY interface occupies three cards --TTY Data Transfer (A1-F1), TTY Control & Code Error (A1-F3), and CRT Interrupt (A1-F5). These same three cards are used as interface for the VKT. Tester CPI boards are contained in A1 slots F7, F9, G1.

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-2 Common Peripheral Interface Manual, Part Number 67095734.

1.3 CONTROLS AND INDICATORS

Table 1-1 provides a description of all controls and indicators for the FST-2 computer. This includes all front panel controls and indicators and switches/indicators mounted on individual circuit cards contained within the CPU. Figure 1-7 is a front panel view of the FST-2 computer.

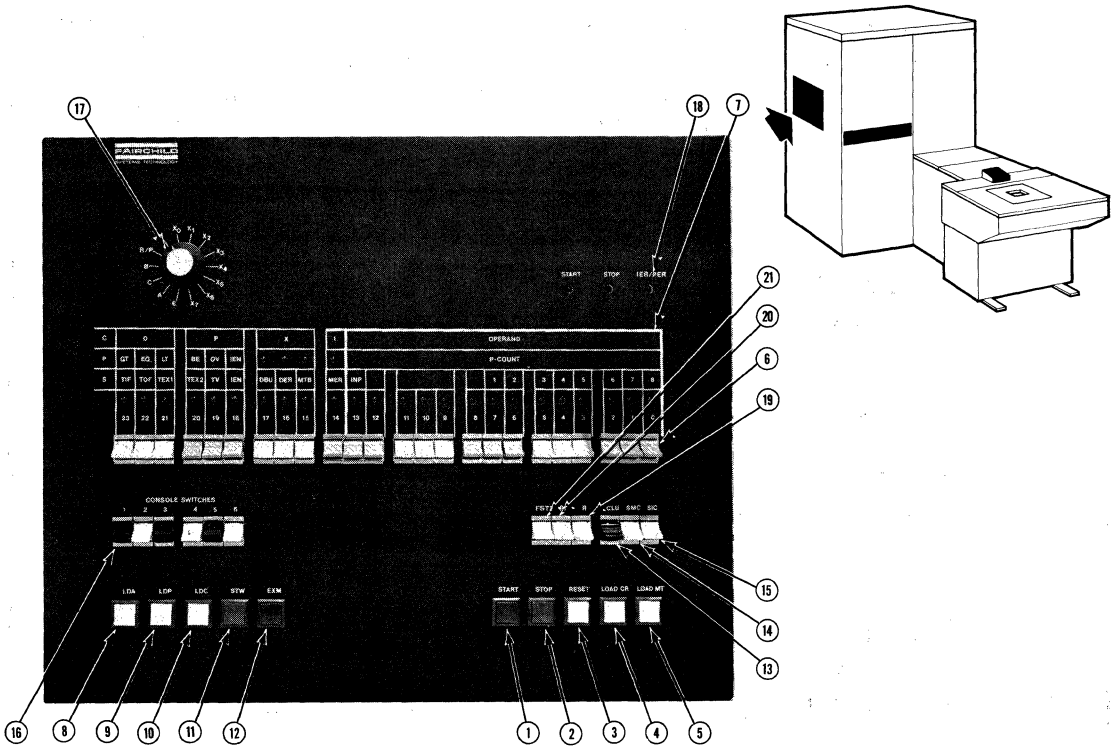


Figure 1-7. FST-2 Front Panel Controls & Indicators

FST-2 COMPUTER MAINTENANCE PANEL

ITEM	CONTROLS/ INDICATORS	FUNCTIONS
1	START Pushbutton/ Indicator	Causes the CPU to start executing program instructions, beginning with the instruction currently held in the command register and turns on the START indicator. While the START indicator is ON, 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 operations as selected.
2	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 ON, all console control switches are enabled.
3	RESET Pushbutton	Resets the program counter to 000000B, and clears any interrupts which may be in process.
4	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 40B consecutive memory locations beginning with address 00100B. This control is primarily used to read the first card of the card object leader program (8 card BOOT) or the one card disc bootstrap.
5	LOAD MT Pushbutton Switch	Loads one record of the magnetic tape DOPSY bootstrap program into 40 consecutive memory locations beginning with address 00100B.
6	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 LDC control, into the A-register by the LDA switch, or by executing an RSR instruction.
7	Register Displays (0 thru 23)	Displays contents of the respective operating register as selected by the Register Display Selector Rotary switch.
8	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.

TABLE 1-1 FST-2 COMPUTER MAINTENANCE PANEL (Cont'd.)

ITEM	CONTROLS INDICATORS	FUNCTIONS
9	LDP Switch	Loads the contents of the switch register into the P-counter. The SIC or SMC switch must also be ON (up) when attempting this operation.
10	LDC Switch	When pressed, this switch loads the contents of the switch register into the command register.
11	STW Switch	Loads the contents of the switch register into the buffer register and the 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.
12	EXAM Switch	Loads the contents of the 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. Thus, the contents of sequential memory addresses may be examined by repeated depression of EXAM switch. The SIC or SMC switch must also be ON (up) when attempting this operation.
13	CLU (command lock-up) Switch	Locks the current command in the command register. The effective memory address is formed by joining bit 0 from the command register with bits 1 thru 17 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 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 memory by loading a STA instruction in the command register, zero in the A-register, and then pressing START.

TABLE 1-1 FST-2 COMPUTER MAINTENANCE PANEL (Cont'd.)

ITEM	CONTROLS INDICATORS	FUNCTIONS
14	SMC Switch	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.
15	SIC Switch	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.
16	CONSOLE SWITCHES (1-6)	The six CONSOLE 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.
17	Reg. Display Sel. 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.

TABLE 1-1 FST-2 COMPUTER MAINTENANCE PANEL (Cont'd.)

POSITION	FUNCTION
C(C-Register)	Selects and displays the contents 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 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.
R/P(P-Register)/ (RR-Register)	Selects and displays the contents of the 18 bit program counter or relocation register on the register display indicator (bits 0-17). In the STOP state, the program counter holds the memory address of the next instruction word that will be loaded into the command-register, if the current instruction is not a branch instruction. See also the RR switch.
S (Control States)	Selects and displays the contents of 8 control states and programmable flip-flops. The contents of the 8 flip-flops (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 ON indicator 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 indicator 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 18 bit index register.

TABLE 1-1 FST-2 COMPUTER MAINTENANCE PANEL (Cont'd.)

PLACARDING	FUNCTION
TIF	When executing any instruction, the TIF (time-of-instruction-fetch) flip-flop will be set, turning ON 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, turning ON 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, turning ON 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, turning ON the TEX 2 indicator, for one memory-cycle time while executing any of the instructions AOM, SOM, MUL, or DIV.
TV	The TV (timing-for-variable-length-shift) flip-flop will be set turning ON the TV indicator while any shift instruction with a non zero shift count is being executed.
POSITION	**FUNCTION**
IEN Interrupt Enable	The interrupt-enable flip-flop is set and the IEN indicator goes ON 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, turning ON 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 ON, indicates that a disc parity-check error has been detected.
MTB	When ON, indicates that the magnetic tape unit is busy.

TABLE 1-1 FST-2 COMPUTER MAINTENANCE PANEL (Cont'd.)

POSITION	FUNCTION
MER	When ON, indicates that a magnetic tape read or write error has been detected.
INP	The input-pending flip-flop is set, turning ON the INP indicator, by an ION instruction for the VKT. This indicator is a visual indicator 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 by depressing the RESET switch. The state of the flip-flop cannot be tested; hence, it cannot control the program-execution sequence. (This function is disabled in FST-2).
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 ON (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

TABLE 1-1 FST-2 COMPUTER MAINTENANCE PANEL (Cont'd.)

POSITION	FUNCTION
OV	The overflow flip-flop will be set and the OV indicator will go ON, in addition to a BRU instruction, by one of the following conditions if the accumulator overflows as the result of executing an ADD,SUB,DADD,DSUB, or DTC instruction. The overflow flip-flop can be reset, turning OFF the OV indicator by executing the appropriate RST instruction or by depressing RESET.
P-COUNT	Defines P-Register or RR-Register contents for rotary switch position R/P.
OPERAND (memory address field)	Defines the instruction word memory address field.
O P X I OPERAND	Defines command register fields: operation code (O P), index (X), indirect (I), and operand (OPERAND) fields.
(placards) C P S	Identifies placards which apply when register selector switch is set for command register (C), program counter (P), and status (S), respectively.

ITEM	CONTROLS/ INDICATORS	FUNCTION
18	IER/ PER	Not Used. Memory parity is provided. When a parity error occurs, the CPU halts and the parity indicator will light up.
19	RR (Relocation Register Switch)	This switch is used when the rotary display switch is at position R/P. At this time, the toggle switch when up displays the relocation register. In the down position, the P counter is displayed.
20	PD Switch	This is the parity disable switch. In the up position it disables parity error detection and can not be overriden by software. In the down position, it allows software to enable or disable parity.

TABLE 1-1 FST-2 COMPUTER MAINTENANCE PANEL (Cont'd.)

ITEM	CONTROLS/ INDICATORS	FUNCTION
21	FST-2 Switch	This switch selects the operating mode of the computer. Up runs the CPU in the new FST-2 mode. When the switch is down, the CPU runs as a standard FST-1. Note: Throwing the switch has no effect on the mode selection until RESET is pushed. This prevents accidental changes while programs are running.

NOTE

The following switches are located on boards within the CPU.

SWITCH	LOCATION	FUNCTION
CRT Transfer	CRT Data Transfer board, Slot F1. Two switches.	Both switches in the down position will provide communication with the VKT. In the up position they provide the option to use a TTY.

SECTION 2

INDIVIDUAL BOARD DESCRIPTION

2.0 INTRODUCTION

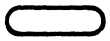
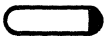
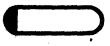
This section describes the circuitry found on the printed circuit cards contained in the CPU and the memory interface. These cards are all located in the A1 card cage.

Table 2-1 shows the layout of the A1 card cage. The CPU occupies positions A1 through G1, except for C5, C7, and C9, which house the three data bus boards comprising the memory interface. Positions E5 and E7 contain the Data Set controllers for available interface to provide a communication link to data processing facilities. Positions E9 through F5 contain the line printer and VKT/TTY controllers. Interface between the CPU and tester is provided by 3 CPI cards F7, F9, and G1.

2.1 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 zones, comprising a 3x3 or 4x8 matrix, indexed by the letters A, B, C, etc., and numbers 1, 2, 3, etc., at the margins, much as in an ordinary road map. Notice the presence of the rotundoid figure, in the following three forms:

Form R  Form S  Form T 

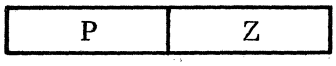
Form R represents an input from another board or location not on this board. Inside the rotundoid is the mnemonic, or abbreviated signal name. To the left or above the rotundoid is the number of the specific pin in the card connector to which that signal is wired. As an example, in zone A3 on page 1 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 (MXNF) 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), 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 on the form



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.

Refer to Appendix E for a summary listing of the major FST-2 hardware signal mnemonics.

2.2 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 1-4 are implemented on these boards:

Accumulator Interface
A (Accumulator) Register
E (Extension) Register
Arithmetic Unit
Buffer Register
Command Register (Less Index and Indirect Bits)
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 18 bits each. The most significant bits are at the C3 end of the 12-board array, while the least significant are at the A1 end.

TABLE 2-1 A1 CARD CAGE

CARD SLOT	CARD NAME	PART NO.
A1	TWO BIT SLICE (BITS 0, 1)	97166001
3	TWO BIT SLICE (BITS 2, 3)	97166001
5	TWO BIT SLICE (BITS 4, 5)	97166001
7	TWO BIT SLICE (BITS 6, 7)	97166001
9	TWO BIT SLICE (BITS 8, 9)	97166001
B1	TWO BIT SLICE (BITS 10, 11)	97166001
3	TWO BIT SLICE (BITS 12, 13)	97166001
5	TWO BIT SLICE (BITS 14, 15)	97166001
7	TWO BIT SLICE (BITS 16, 17)	97166001
9	TWO BIT SLICE (BITS 18, 19)	97166001
C1	TWO BIT SLICE (BITS 20, 21)	97166001
3	TWO BIT SLICE (BITS 22, 23)	97166001
5	DATA BUS B (BITS 0-7)	97420101
7	DATA BUS B (BITS 8-15)	97420101
9	DATA BUS B (BITS 16-23)	97420101
D1	CPU BLOCK A	97420102
3	MASTER CONTROL FF A	97420103
5	MEMORY CONTROL B	97420107
7	INSTRUCTION DECODER A	97420104
9	GATE CONTROL 1A	97420105
E1	GATE CONTROL 2A	97420106
3	MISC REGISTERS	97166009
5	DATA SET INTF BD1	97420109
7	DATA SET INTF BD2	97420110
9	LINE PRINTER CONTROLLER	97166124
F1	TTY DATA TRANSFER	97340202
3	TTY CONTROL & CODE ERROR	97166102
5	CRT INTERRUPT	97340205
7	TESTER CPI 1	97420108
9	TESTER CPI 2	97420108
G1	TESTER CPI 3	97166108

NOTE

PCB's shaded in table above constitute the CPU portion of module A1.

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 devices F1, A3, C2-10, 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 C3 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 seperate 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 18 bits in length, this circuitry is unused on three of the twelve boards.

At the top of page 2 (zone C3), the signal AZERO is generated. This signal is OR-tied with outputs from 11 other boards. AZERO is true if and only if all bits of the A register are zero. AZERO 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 8-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 singals 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.

The TRI-STATE gate E7 buffers the switch register WM, WN to the PWM, PWN bus when GWPX/ (Gate W to P Counter) is low.

2.3 DATA BUS B (97420101-04, 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 1.2.1.

Eight bits of the 24-bit interface are implemented on each board -- bits 0-7 at card slot C5, 8-15 at slot C7, and 16-23 at slot C9.

On sheet one of the Data Bus B board schematic at the extreme left are the multiplexers which control data flow to memory (Gates E1-7 odd). The mux inputs are designated B, P, SOR and PW for the buffer register, program counter, sum of the operand plus relocation register, and the P counter or switch register bus 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, the operand field plus relocation by GOM, and the PW by GPM and GOM1/ (zone A8).

The buffer, P counter, and operand registers are all 24 bits long. The buffer register has all 24 bits gated to memory by signal GBM. The upper 6 bits of the operand input to the board are tied to ground so although 24 bits of SOR are gated to memory by GOM/, only 18 are active. Under a normal instruction fetch cycle 24 bits of the P counter are gated to memory by GPM. Since the upper 6 bits of the P counter are assumed to be zero, the active address to memory is only 18 bits.

During a BSM (Branch Store Return address in Memory), the P counter is again gated to memory by GPM, only GOM1/ is also active which selects the PW inputs on the upper 6 multiplexers on the third board. These inputs are wired to signals OVF1, GT, EQ, LT, BE, and OVF2 thus the status bits are gated to memory to be stored. (OVF1 is overflow in FST1 mode, OVF2 is overflow in FST-2 mode.)

The outputs of the memory data multiplexers are gated to the memory bus through tri-state gates (E2-8 even) by signals GPM/, GOM/, GBM/, AMEM/ and BMEM/. The memory bus is gated to the peripheral memory bus through 7438's (F2-8 even) during T1 time. The peripheral memory bus is gated back to the memory bus through tri-state gates (F3-7 odd) during T2 through T5 unless the CPU is gating data to or from memory during the same cycle.

The multiplexers in the center of page 1 select whether the A memory bus, B memory bus, sum of operand plus relocation, or relocation register is sent to the buffer register via signals designated M. The selection is controlled by signals SM0 and SM1 which originate on the memory control B board.

Devices C5 and D5 on the right side are the A and B memory parity generators. The ninth input (I8) on the first board comes from the 25th bit of memory. The outputs on the last board go to the 25th bit in memory and to the parity error circuit on the clock board. The other I8 inputs and outputs are used for cascading the devices. Each device generates an odd parity from the nine inputs.

The cascading of these devices results in an even parity being sent to memory. During a read cycle, if an odd parity is detected then the output from board three, to the clock board, will be low indicating a parity error.

On sheet two upper left is the multiplexer, devices C4 and C1, for loading the P counter. The signals GWP and GOP select the switch register or sum of operand plus relocation register as inputs to the P counter. The P counter (gates B1 and B4) is 24 bits long. The lower 18 bits contain addresses for memory. Since memory addressing circuitry only monitors 18 bits, the upper 6 bits of P are ignored.

The output of the P counter and relocation register are multiplexed for display purposes by gates A1, 3, and 4. Signal DSPLR/ comes from a switch on the front panel (see Section 1) to control the selection. The output of the mux is a tri-state bus (PWXX). This bus normally contains the data to be displayed. When the P counter is to be loaded from the switch register, signal GWP disables the mux output and the switch register is gated to the bus via the 2 bit slice boards. The PW bus is not wired to the upper six bits on the last board since both R and P only use 18 bits for addresses. These spare inputs are used to bring in the status indicators (OVF1, OVF2, GT, LT, BE, EQ) which must be stored in memory during a BSM instruction. These signals are not tri-state so the mux outputs on board 3 must be disabled by wiring pin D7 to VCC. This is why only half of mux A1 and A3 are used since two bits (16 and 17) must be used for the P and R registers. On boards 1 and 2, Pin D7 is wired to signal GWP.

The lower left gates A2 and A7 are the relocation register. Data is clocked in from the buffer register by signals WR. The multiplexers B2 and B7 select whether the relocation register or zero is gated to the adder. Selection is done by signal SELR/ from the instruction decoder A board.

The adder (gates C2 and C7) adds either the relocation value or zero to the operand. The resultant sum called SOR can be sent to the P counter, memory, buffer register or operand register.

The signal GOPM1 is wired to Pin G5 on the first board to control carry in for the least significant bits. 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 address location X, and the most significant half at location X + 1. The operand supplies the address for the X location and the most significant half word is transferred into (or from) the A register. GOPM1 then goes high adding one to the operand, and (during the next memory cycle) the least significant half word is transferred to (or from) the E register. On boards 2 and 3 this signal is the carry for the upper stages (RCRY08 and RCRY16 respectively).

2.4 CPU CLOCK A (97420102-04, 2 PAGES)

The principle function of this board is to generate the basic CPU timing signals. In addition, a variety of start, stop, and single cycle control functions are implemented. Parity errors are detected and the mode of operation is selected.

At the top of page 2 (zone 7-8D), 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 active, this square wave becomes the input to the divide-by-four counter C5. The outputs are decoded by the AND gates at the center of the page. Of the four possible states, three are decoded at E5-6, E5-8, and E5-11. The fourth state is decoded by all the gates from D5-3 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 D7 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 A7-6 and A7-8 are used. A7-6 eliminates the possibility of having all zeroes in the counter, while A7-8 precludes the presence of two or more ones.

Flip-flop E8 (center left) controls whether the CPU runs in the FST1 or FST2 mode. The mode may be set in two ways. Pushing RESET on the front panel will set the mode to that selected by the mode (FST-2) switch on the maintenance panel (See Section 1). Software may set either state by using the (LAR) load accumulator from relocation register instruction with bits 0 or 1 on in the operand field.

Circuitry in the lower left is for parity error detection. Flip-flop A5 enables or disables parity error detection. Parity can be disabled from the maintenance panel by the parity disable (PD) switch. Parity can be enabled only or disabled by software using the set state or reset state instruction and operand bit 14.

B6-2 and B6-12 monitor the output of the parity circuits, and whether parity is enabled. If an error occurs MAPERR/MBPERR halts the CPU and sets flip-flop A6. LED's indicate which memory, A or B, failed. PAERR goes to the maintenance panel to light the parity error (PER) lamp.

STOPM/ goes to the memory control board to inhibit any further memory accessing.

The circuitry on page 1 serves to implement basic control functions affecting the CPU as a whole. At the top center of the schematic, built around flip-flop A2, 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 (zone D3) 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 B2 must be set. In this case the signal (LRMAN) from A4-8 is used to produce GRCPS, which permits manual loading of registers. KFML is not used. The output ML/ (F5-6), when low, inhibits instruction decoding, instruction fetch, and the gating of address information to memory.

B2 is the halt-run flip-flop. The circuitry to the left of B2 collects at D2-8 all the conditions which bring the CPU to a halt. The following list defines the functions of the input signals involved:

TP1/	Spare Halt Before Execution
ISTOP	Not Used
BAH	Branch after Halt
SMC	Single Memory Cycle
MAPERR	A Memory Parity Error
MBPERR	B Memory Parity Error
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
SPRHLTI	Spare Halt

In all the above cases except BAH and TP1/ the current CPU cycle is finished before stopping; the halt flip-flop is set at phase time T1. Flip-flop C3 allows the CPU to continue when START is pushed following a branch after halt instruction (BAH). Flip-flop C4-6 (and the circuitry to its left) synchronizes the Start operation. The input KSTRT is a pushbutton and D4 is a one-shot which generates a pulse when the START button is pressed. The input .F.D11 is not used.

Flip-flops A4-6 and C4-10 serve functions analogous to those of B2-6 and C4-6, 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
KSTW	STORE (memory location) button
KEXAM	EXAMINE (memory location) button

Flip-flop B4 generates the Command Register Lock-up signals CRLU and CRLU/, under the control of the STORE (KSTW) and EXAMINE (KEXAM) pushbuttons, and the Command Register Lock-up switch on the control panel (KCRLU).

2.5 MASTER CONTROL FF A (97420103-04, 1 PAGE)

This board contains nine 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 Execution 2 (TEX 2)
Time Variable (TV)
Interrupt Enable (IE)
Interrupt (INT)
Overflow (OV)
Carry (CY)

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 instruction 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 T5 time) when an interrupt address 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 multiplexer A6 (lower center) controls where the overflow bits are stored during (BSM), and retrieved from during (BRU-indirect). In the FST-1 mode it uses bit 23 of memory and in the FST-2 mode bit 18 is used.

The latch (B1) and decoder (B2) in zone B1 decode additional augmented instructions (LXA, LAX, LRA, and LAR). The latch is required as these lower operand bits (9, 8, and 7) may change during the instruction execution.

2.6 INSTRUCTION DECODER A (97420104-04, 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 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, and 23 in FST-1 mode by the logic shown at the extreme left.

In terms of the octal digit representing the two most significant bits, the value 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 (O15, 16, 17), are used for purposes other than address modification.

INA1 (formed at F2-8) 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/or LAX, 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 center half of the page (zone B2) IC/ is generated (E7-3). 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.

XSUB/ (E4-6 Zone C1) is the INDEX Subtract signal. This signal is generated when a LDX instruction is executed, if operand bit 13 is on, no indirection or relocation is required and the CPU is in FST-2 mode. XSUB/ goes to the two bit slice boards for bits 14-17 and forces ones into those bits of the selected index. This allows loading of a minus one (all ones) into the 18 bit index directly. Note only 14 bits of operand are available to load index registers directly. Only at the time under the above conditions, when operand bit 13 is on, will ones be forced into the upper index bits.

SELR/ (B8-6 Zone 1C) is the Select Relocation Register signal. This signal goes to the Data Bus boards to control when the relocation register should be added to the operand. The CPU must be in the FST-2 mode before this signal can become active. Three conditions can generate this signal: LAR (load accumulator from relocation register), LXA (load index from accumulator) with accumulator bit 23 equaling a one, or any instruction with operand bit 23 on except during T5 of a STX (store index) instruction.

Other signals generated on this board have the following functions:

KLGOM/	(Zone B3)	Kills gating of operand to memory during augmented and add to index instructions. This prevents erroneous parity errors.
GM1UA/	(Zone A2)	Gates a minus 1 to the A input of adder bits 0-13 & 18-23 during SOM (subtract one from memory).
AOMTEX1/	(Zone A2)	Generates CI during AOM (add one to memory) instruction.
GSE	(Zone B2)	Gate adder sum to E register.
GSB	(Zone B2)	Gate adder sum to B register
WR	(Zone B2)	Write the relocation register
GM1UAA/	(Zone A1)	Gates a minus 1 to the A input of adder (bits 14-17) during SOM, ATX no indirection & O13 = 1 and indexing if O13=1.
GUAX1/	(Zone B1)	Select inputs to A side of adder bits 0-13.
GUAX2/	(Zone B1)	Select inputs to A side of adder bits 14-17.
GUAX3/	(Zone A1)	Select inputs to A side of adder bits 18-23.
GBUBX1/	(Zone A1)	Select inputs to B side of adder bits 0-13.
GBUBX2/	(Zone A1)	Select inputs to B side of adder bits 14-23.

2.7 GATE CONTROL 1A (97420105-04, 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 left 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 lower right are the outputs for gating the command register inputs. GCP09/gates bits 0-9, GCP1013/ gates bits 10-13, GOA1417/gates the alternate operand bits 14-17. GCPI/ gates bit 14, GCPXA/ gates bits 15-17, and GCPC/ gates bits 18-22. GCPO23/gates bit 23. GMCR/ (upper right) gates the memory to the command register, GMO14-17/gates memory to alternate operand bits 14-17 (used as address for memory locations octal 40000 and above), GRO/ gates the contents of the interrupt (R) register to the lowest six bits of the command register (used as address for memory locations 77B and below).

Other signals generated on this board have the following functions:

RX15	(Zone 2C)	Clears X15 to permit comparison of contents of odd and even numbered X registers
G23CO	(Zone B6)	Sets iteration counter to decimal 23 (octal 27)--used in multiplication and division
GAE	(Zone A6)	Gates A register to E register
GCRN/	(Zone A5)	Gates command register to N bus
PS	(Zone A5)	Peripheral Select
GAN	(Zone A7)	Gates A register to N bus
IPC	(Zone A3)	Increments program counter
STWB/	(Zone B4)	Store switch register (W) to buffer register
WX/	(Zone D3)	Write index
LDATX/	(Zone D4)	Loads or adds to index register
GXA	(Zone C4)	Gates index register to A register
GCPA/	(Zone C4)	Gates clock pulse to A register
IP	(Zone A4)	Signals peripherals to send interrupt priority signal (on N bus)
IAT	(Zone A4)	Loads interrupt address into R register
GCPB/	(Zone C5)	Gates clock pulse to B register
GCPE/	(Zone B5)	Gates clock pulse to E register
GXUB/	(Zone D3)	Gates index register to B input of adder

2.8 GATE CONTROL 2A (97420106-04, 1 PAGE)

This board serves the same general function as Gate Control #1A. The two (9301) decoders and 9314 latch 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	(Zone A1)	Force carry into arithmetic unit (two's complement)
CYE/	(Zone B3)	Carry to E0 (divide)
DCO	(Zone B3)	Decrement iteration counter
DSA	(Zone B3)	Double shift around
DSN/	(Zone B2)	Double shift normalized
DTC/	(Zone C2)	Double two's complement
EXC/	(Zone C3)	Exchange A register with E register
GAFUA	(Zone B1)	Gate A register complement to A input of arithmetic unit
GBM	(Zone B1)	Gate buffer register to memory
GEFUA	(Zone C1)	Gate E register to A input of arithmetic unit
GOCO	(Zone C1)	Gate operand field (lowest six bits) to iteration counter
GOM/	(Zone A1)	Gate operand field to memory interface
GOM1/	(Zone B1)	Same function as GOM/ plus active during BSM and GPM to gate status bits to memory
GOP	(Zone A3)	Gate operand field to program counter
GPM/	(Zone B1)	Gate program counter to memory
GSA	(Zone B3)	Gate sum output from arithmetic unit to A register
GOP1M/	(Zone A2)	Adds one to operand address (double precision)
RSR	(Zone C3)	Read switch register into accumulator.
RST	(Zone C3)	Reset state
SA	(Zone B2)	Shift around
SN	(Zone C1)	Shift normalized

SST	(Zone C3)	Set state
SAL	(Zone B2)	Shift A register left
SAR	(Zone B2)	Shift A register right
SEL	(Zone B2)	Shift E register left
SER	(Zone B3)	Shift E register right
WRM	(Zone A3)	Write to memory
SAUG/	(Zone C3)	Sub Augmented instruction

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 O0-O7. 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 (CAM) 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 2A 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. See Section 6 for details of peripheral commands. 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 #1A 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 #1A) 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 O14, 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 B (97420107-04, 2 PAGES)

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, Column address, Row address signals, and refresh counter for memory control purposes. Also located on this board is the iteration counter, which plays an important part in multiplication, division, shift operations, index and indirect bits of the operand.

At the bottom of page 1 (Zone A4/A5) 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 B 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 0 in either the command register or the program counter controls which memory is used (bit 0 = 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 A6 to A8).

At gate E6-6 (zone B2) 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 page 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 G0CO 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 F3-10 (zone C-4), COEQ1 at E4-6 (zone B-4), and COLEQ5 at B2-8 (zone D-4)) are formed 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.

In zone C-3 of the schematic, page 2 of 2, are two flip-flops, A6-10 and A6-6 which are the A and B memory write flip-flops. These are set at phase time T3, according to the write request signals from the CPU (WRM) or peripherals on A or B memory bus bit 18. The outputs AMW, BMW, of these flip-flops go to the logic group at zone B5/B6. This logic group generates the memory control signals ASWC, BSWC (Start Write Cycle) at time T1.

The multiplexers C7 and D7 (zone C7 and D7) generate ARAS/,BRAS/, (A and B Row Address Select) signals. These signals occur at different times depending on whether the cycle is read, write, or refresh. Counter C8 (zone B-8) is the refresh counter and causes a refresh cycle to occur every 15 memory cycles.

In the lower left corner flip-flops C6-9 & C6-5 generate ACOLT/, BCOLT/ (A/B Column Address) signals. These signals control when the column address is written into the memory chips.

Across the top are the four flip-flops B7 and B8, comprising the indirect and index bits of the operand register. They are loaded from the switch register or memory depending upon multiplexer F7.

The last circuit on the board (zone B4) generates the gating signals SM0-1 (select M register) which go to the Data Bus B Board. These signals control whether the A memory bus, B memory bus, relocation register, or sum of relocation register plus operand is presented to the buffer register as well as to the command register.

SECTION 3

INSTRUCTION REPERTOIRE

3.0 INTRODUCTION

This section discusses the instruction repertoire of the FST-2 which consists of ten instruction groups. Accompanying each instruction are examples coded as they would be for the FST-2 assembler.

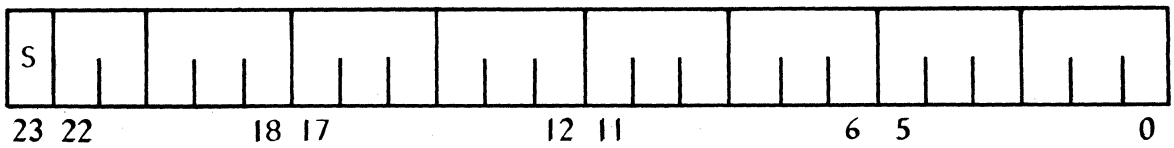
3.1 ABBREVIATIONS AND MNEMONICS

Appendices A, B, and C contain a list of abbreviations, instruction mnemonics, and opcodes which are used in the description of machine instructions. Any abbreviation which is enclosed in parenthesis 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. Refer to 3.2.2 for details. 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

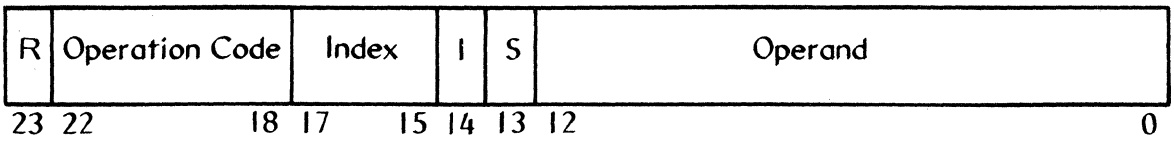
3.2.1 Word Format

The FST-2 CPU data 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 six parts which consist of: first, a 1 bit relocation field to specify absolute (0) or relocatable (1); second, a five (5) bit operation code field; third, a three (3) bit index field; fourth, a one (1) bit field used to specify direct (0) or indirect (1) address; fifth, a bit sign field to specify the operand being positive (0) or negative (1); and sixth, a 13-bit operand field.



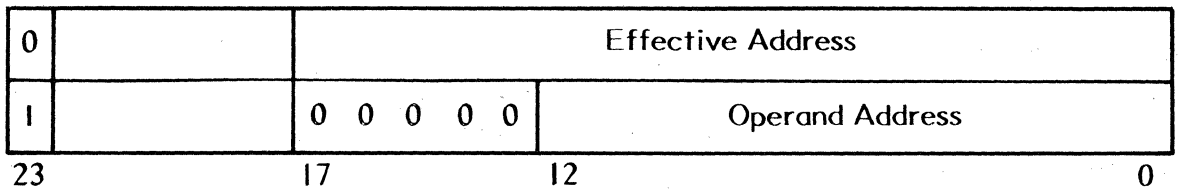
If the instruction can reference memory, then the M in the operand field is the memory location.

If an instruction is indexable, the instruction format will include an X, which indicates that the operand is added to the index to obtain Me (effective memory address).

If the instruction is relocatable, an R is included in the instruction format. When R equals 1, the operand address is added to RR (Relocation Register) to obtain Me. Since a memory address is the only operand which can be relocatable, the R is applicable only in memory reference instructions.

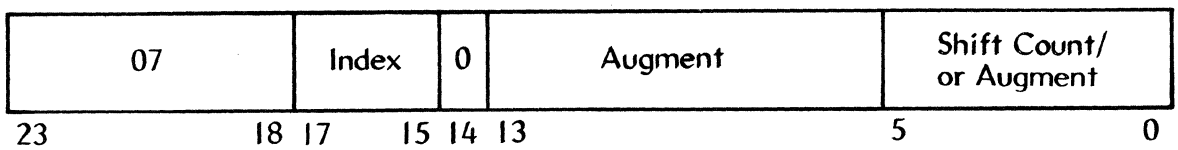
When S equals 1, the operand is subtracted from index to obtain Me. The S is applicable only to indexed instructions and the operand is always absolute, i.e., R is zero.

If an instruction can be indirect address modified, the format will include an I. One level of indirection is allowed, i.e., the contents of address indirectly addressed may not be an indirect or indexed instruction. If the R field of the contents of an address indirectly addressed is equal to 1, then the least significant 13 bits of that address is added to RR Register to obtain Me; otherwise, the least significant 18 bits of the contents of that address is Me. It is illustrated below.



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 00 through 13 to 'augment' the 07 operation code (See sections 3.5, 3.6, 3.7, and 3.8 for further details). An example of an augmented operation code is shown below.



3.3 ASSEMBLER FORMATS

Each of the instruction descriptions which follow is illustrated with an example of the appropriate FST-2 symbolic assembly code.

The FST-2 Assembler, Section 7 of this manual, should be consulted for the details of instruction formats, conventions, etc.

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. If the instruction is indexed it takes an additional one cycle. If it is indirect, it also takes an additional one cycle. Relocation does not add another cycle.

3.5 ARITHMETIC INSTRUCTIONS

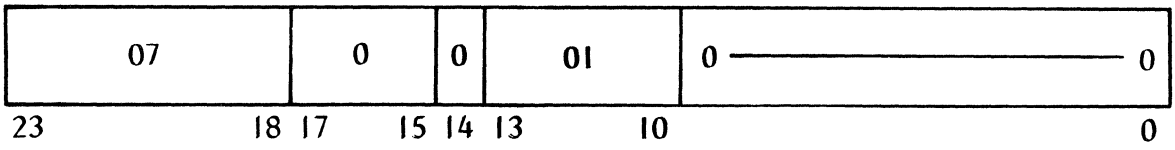
The FST-2 provides the following 10 arithmetic instructions: TCA, DTC, ADD, SUB, DADD, DSUB, MUL, DIV, AOM and SOM.

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.

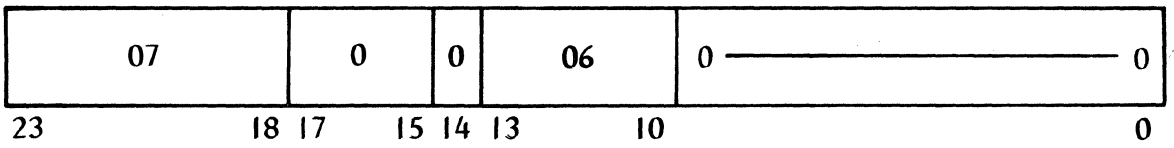
Assembler Format: TCA

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.

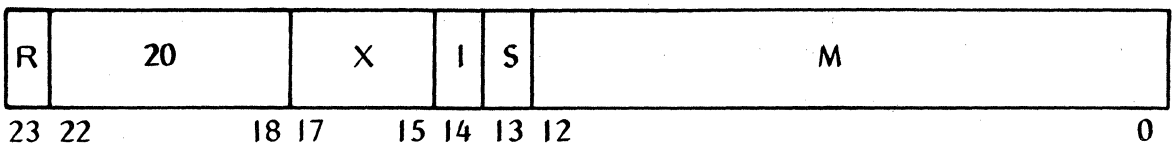
Assembler Format: DTC

3.5.3 ADD Addition

Definition: (A) + (Me)→A

Cycles Required: 2

Instruction Format:



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.

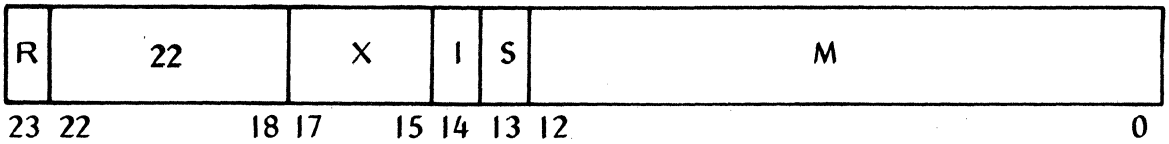
Assembler Format: ADD TABLE+1

3.5.4 SUB Subtraction

Definition: (A) - (Me) → A

Cycles Required: 2

Instruction Format:



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.

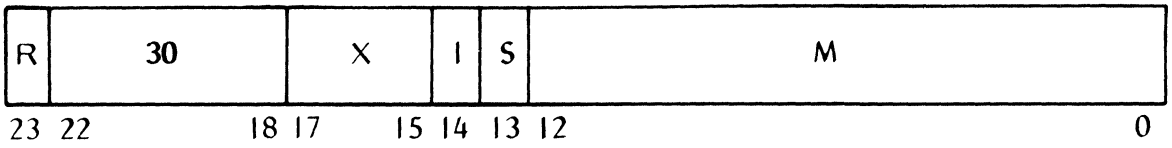
Assembler Format: SUB TABLE+1

3.5.5 DADD Double Addition

Definition: (A and E) + (Me and Me + 1) → A and E.

Cycles Required: 4

Instruction Format:



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.

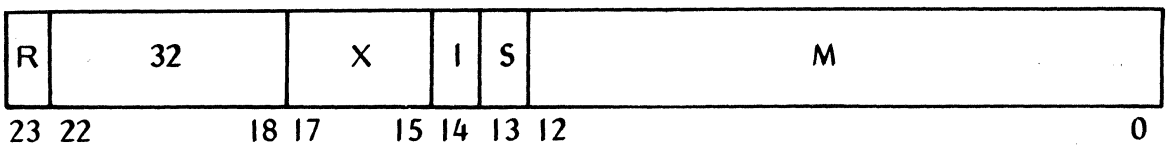
Assembler Format: DADD* TABLE + 1

3.5.6 DSUB Double Subtraction

Definition: (A and E)-(Me and Me + 1) → A and E.

Cycles Required: 4

Instruction Format:



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 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.

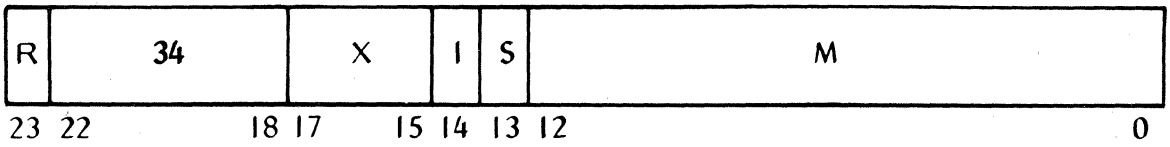
Assembler Format: DSUB* TABLE+1

3.5.7 MUL Multiply

Definition: (Me) x (A) → A and E

Cycles Required: 25

Instruction Format:



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.

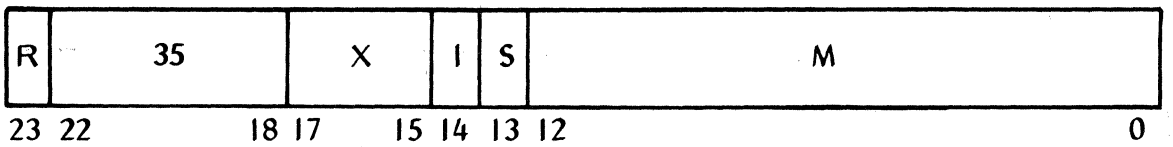
Assembler Format: MUL* ALPHA

3.5.8 DIV Division

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

Cycles Required: 26

Instruction Format:



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) ≥ (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.

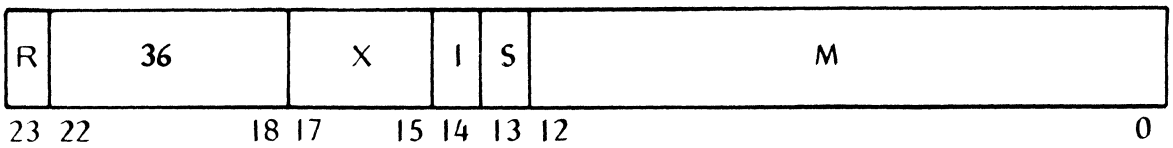
Assembler Format: DIV* ROGER,2

3.5.9 AOM Add One To Memory

Definition: (Me) + 1 → Me

Cycles Required: 4

Instruction Format:



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.

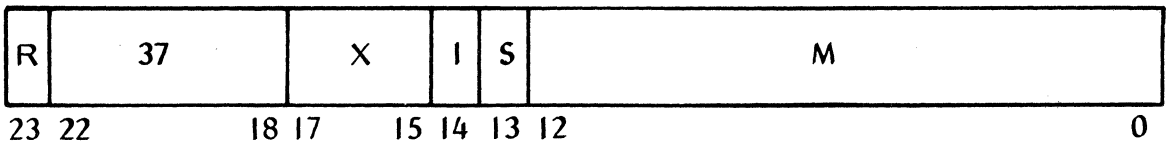
Assembler Format: AOM* DETA

3.5.10 SOM Subtract One From Memory

Definition: (Me) - 1 → Me

Cycles Required: 4

Instruction Format:



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

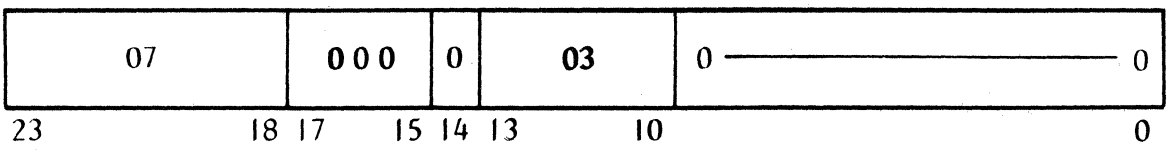
The FST-2 uses the following 11 data transfer instructions: RSR, EXC, STA, STE, LDA, LDE, DLD, DST, LRA, LAR, and CLA.

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.

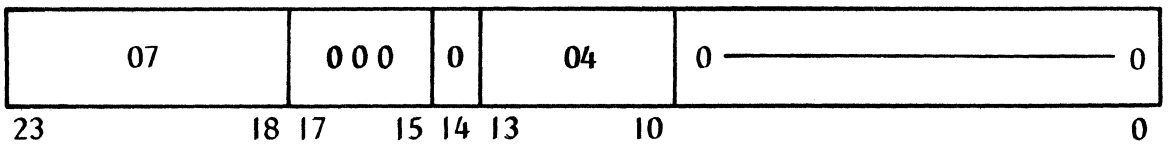
Assembler Format: RSR

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.

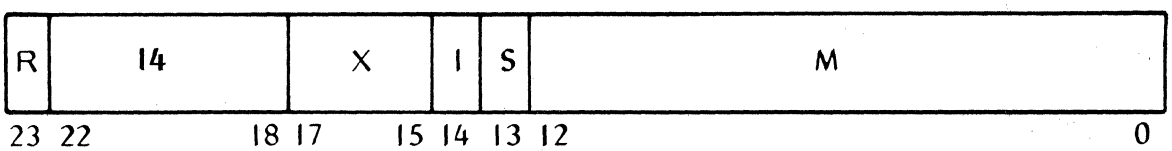
Assembler Format: EXC

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.

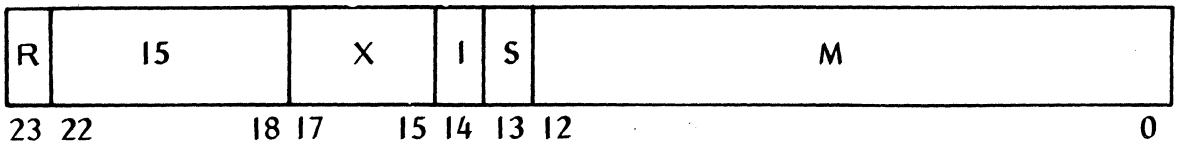
Assembler Format: STA* ALP1,X2

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.

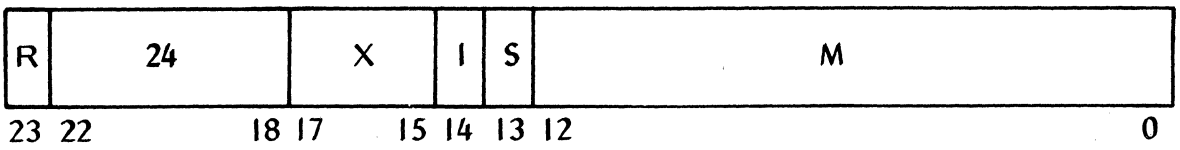
Assembler Format: STE* TEMP5

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.

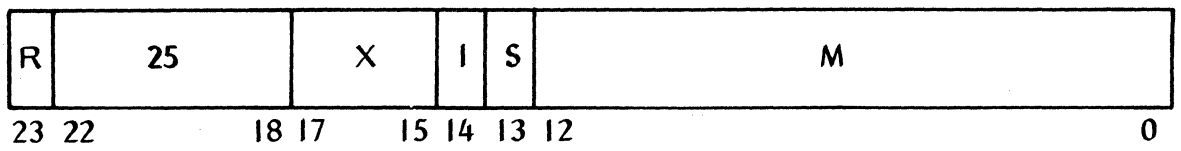
Assembler Format: LDA* TABLE

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.

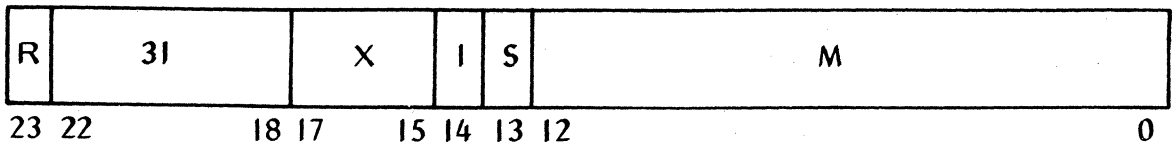
Assembler Format: LDE* TABLE+1

3.6.7 DLD Double Load A And E

Definition: (Me and Me + 1)→A and E.

Cycles Required: 3

Instruction Format:



Description:

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

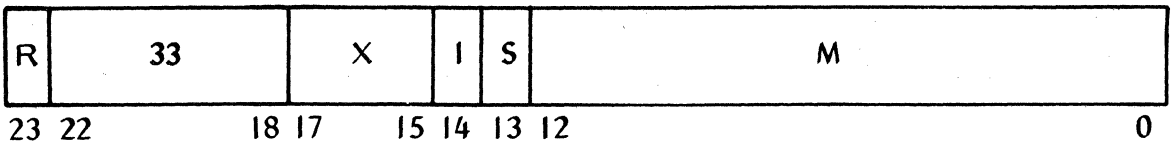
Assembler Format: DLD* TABLE+5,2

3.6.8 DST Double Store A and E

Definition: (A and E)→ Me and Me +1.

Cycles Required: 3

Instruction Format:



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.

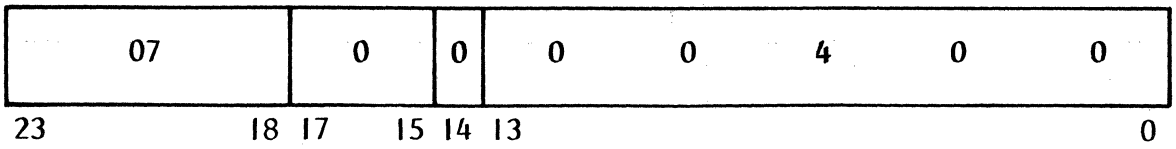
Assembler Format: DST TEMP2+2

3.6.9 LRA Load RR From A

Definition: (A)→RR

Cycles Required: 1

Instruction Format:



Description:

The contents of accumulator (A17-A0) are loaded to the Relocation Register, RR. The contents of A are not changed.

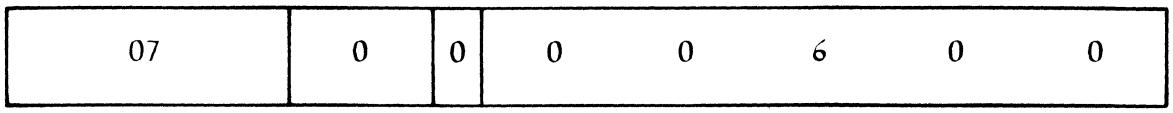
Assembler Format: LRA

3.6.10 LAR Load A from RR

Definition: (RR)→A

Cycles Required: 1

Instruction Format:



Description:

The contents of Relocation Register are loaded into the accumulator. A23-18 are cleared to zero. The contents of RR are not changed.

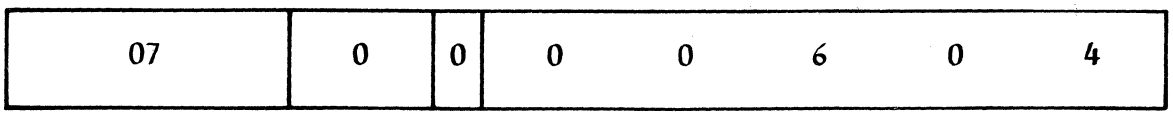
Assembler Format: LAR

3.6.11 CLA Clear A

Definition: 0→A

Cycles Required: 1

Instruction Format:



Description:

The contents of Accumulator are cleared to zero. A23-A0=0.

Assembler Format:

CLA

3.7 INDEX INSTRUCTIONS

The FST-2 provides the following 5 index instructions: LDX, LXA, ATX, STX, and LAX.

NOTE

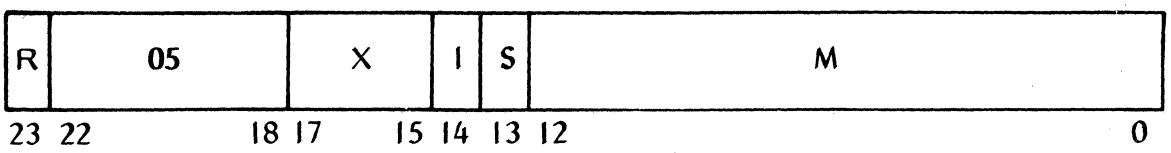
It is standard practice to use the index registers in the FST-2 in adjacent pairs, viz: X7 with X6, X5 with X4, X3 with X2, and X1 with X0. When so used, the odd index is the active, working index while the even index is the limit index for comparison purposes.

3.7.1 LDX Load Index

Definition: Me → Xn (for case 1 as described below)

Cycles Required: 1

Instruction Format:



Description:

1. No Indirection and no Relocation - the 14 bit address field is loaded into the addressed index register. If bit 13=0, index register bits 14-17 are loaded with 0 (positive). If bit 13=1, index register bits 14-17 are loaded with 1 (negative) number.
2. No Indirection with Relocation - the sum of relocation register (18 bits) and address field (14 bits) are loaded into the addressed index register.
3. Indirection - the least significant 18 bits of the contents of the addressed memory location are loaded into the addressed index register.

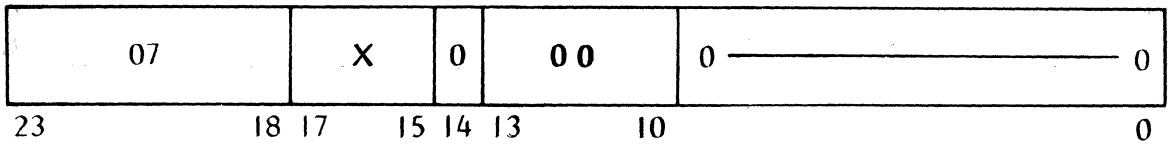
Assembler Format: LDX* X7, TEMP

3.7.2 LXA Load Index From A

Definition: (A)→Xn

Cycles Required: 1

Instruction Format (Augmented):



Description:

If bit 23 of A is zero, the least significant 18 bits of A are loaded to Xn. If bit 23 of A is one, the least significant 14 bits are added to RR Register and Loaded to Xn.

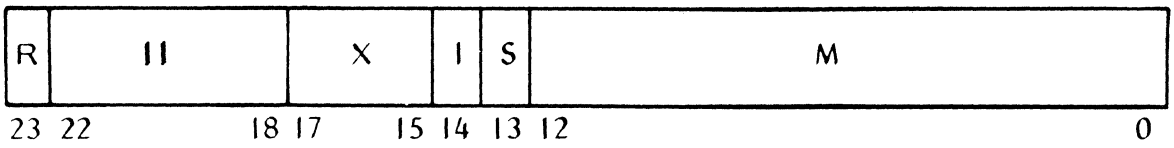
Assembler Format: LXA X3

3.7.3 ATX Add To Index

Definition: Me + (Xn)→Xn (for case 1 as described below)

Cycles Required: 2

Instruction Format:



Description:

1. No Indirection and no Relocation - the 14 bit address field is extended to 18 bits (If bit 13=0, bits 14-17=0; if bit 13=1, bits 14-17=1); then it is added to the addressed index register Xn and the sum placed back in Xn. The GT, EQ, and LT indicators are then set by comparing Xn to Xn-1 (n must be odd).
2. No Indirection with Relocation - the sum of relocation register (18 bits) and address field (14 bits) is added to Xn and the sum placed back in Xn. The GT, EQ, and LT indicators are then set by comparing Xn to Xn-1. (n must be odd).
3. Indirection - the least significant 18 bits of the contents of the addressed memory location are added to Xn and the sum placed back in Xn. The GT, EQ, and LT indicators are then set by comparing Xn to Xn-1. (n must be (odd).

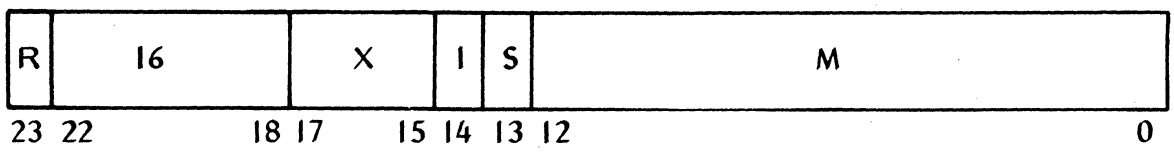
Assembler Format ATX* X5, TABLE 1

3.7.4 STX Store Index

Definition: (Xn) → Me

Cycles Required: 2

Instruction Format:



Description:

The contents of the addressed index register X_n are stored in memory location Me, bits 0-17. Bits 18-23 of Me are loaded with zeroes. Because the X field indicates the index register to be stored, indexing is not possible. Indirect Addressing is allowed.

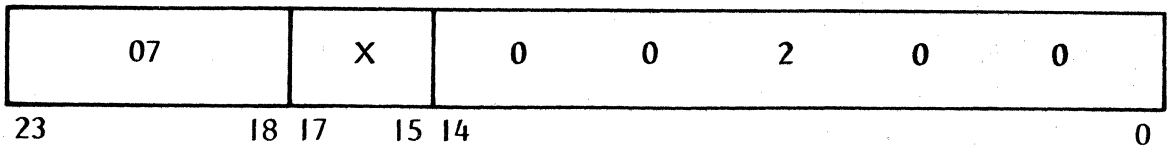
Assembler Format: STX X5,TEMP1

3.7.5 LAX Load A From Index

Definition: (Xn) → A

Cycles Required: 1

Instruction Format:



Description:

The contents of the specified index register are transferred to A, bits 17-0. Bits 23-18 of A are zeroed.

Assembler Format: LAX X2

3.8 SHIFT INSTRUCTIONS

The FST-2 provides 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:

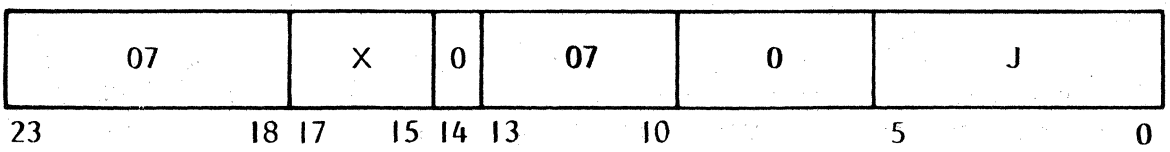
$$T = 2 + \underset{\text{integer}}{[(Je-9)/5]} \quad \text{cycles.}$$

3.8.1 DSN Double Shift Normalize

Definition: Normalize A and E

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

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 than in A22. E23 shifts into A0 and zeroes 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.

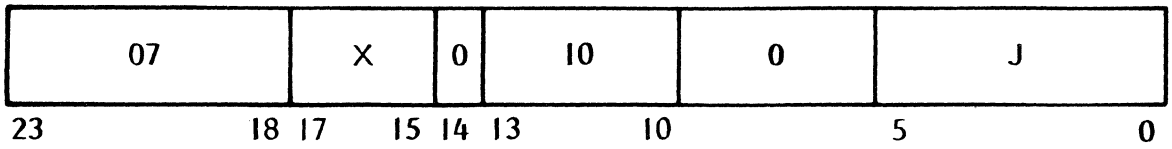
Assembler Format: DSN 4

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, 21, etc., 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.

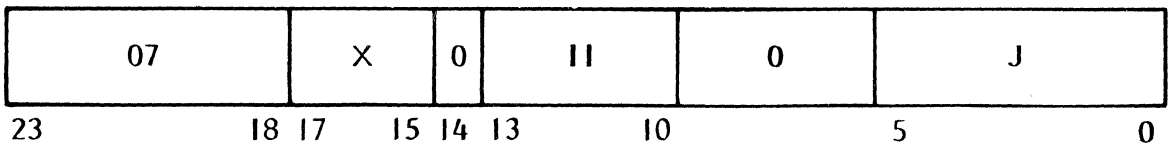
Assembler Format: SR 5

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, zeroes 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.

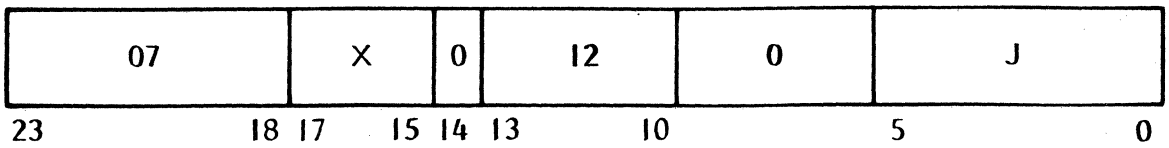
Assembler Format: LS 5

3.8.4 SA Shift Around

Definition: Shift (A) Left Around

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

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.

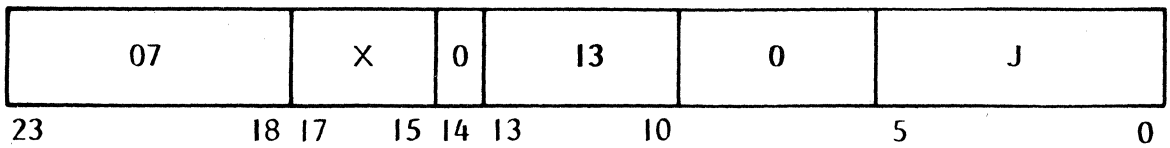
Assembler Format: SA 4

3.8.5 SL Shift Left

Definition: Shift (A) Left End Off

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

Instruction Format:



Description:

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

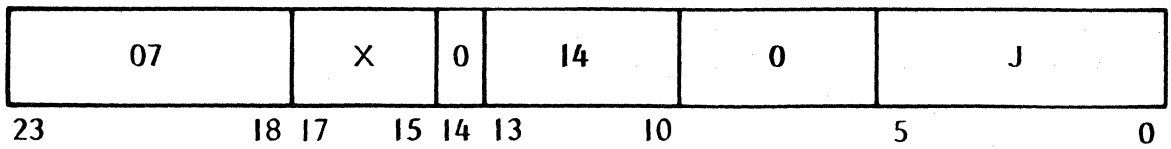
Assembler Format: SL 0, X3

3.8.6 DSR Double Shift Right

Definition: Shift A and E Right Arithmetical

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

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.

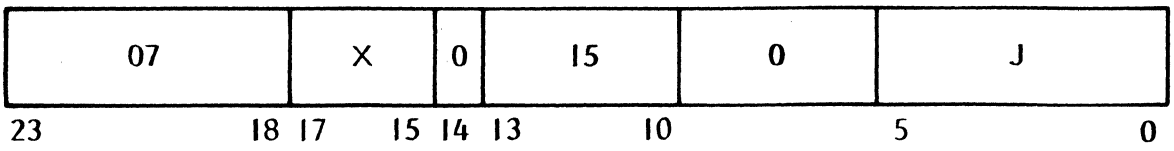
Assembler Format: DSR 25

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. Zeroes 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.

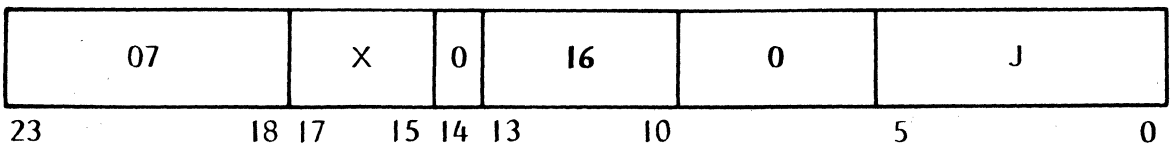
Assembler Format: LDS 0, X5

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.

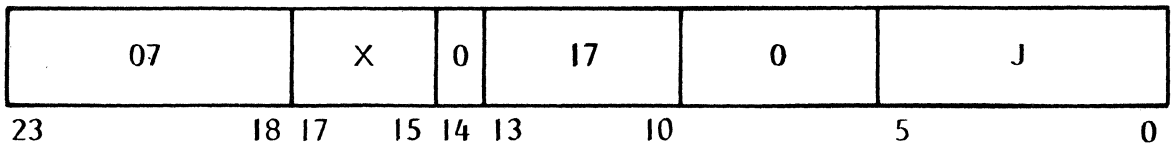
Assembler Format: DSA 24

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. Zeroes 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

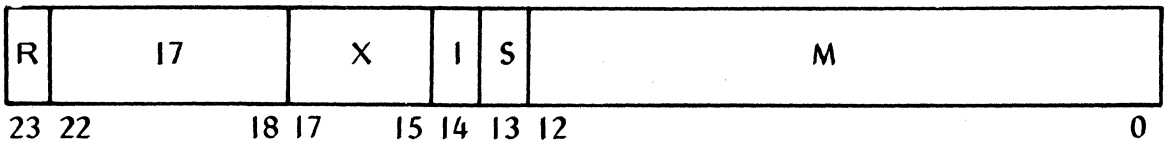
The FST-2 provides the following 4 logical instructions; RUM, EOR, AND and OR.

3.9.1 RUM Replace Under Mask

Definition: [(Me) AND (E)]EOR[(A) AND ($\bar{E}$)]→A
on a bit by bit basis.

Cycles Required: 2

Instruction Format:



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.

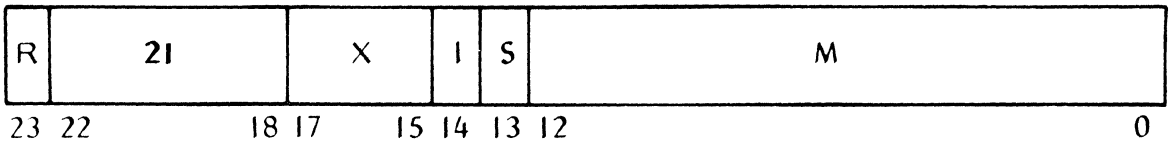
Assembler Format: RUM O 2020

3.9.2 EOR Exclusive OR

Definition: (A) EOR (Me)→A

Cycles Required: 2

Opcode Format:



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.

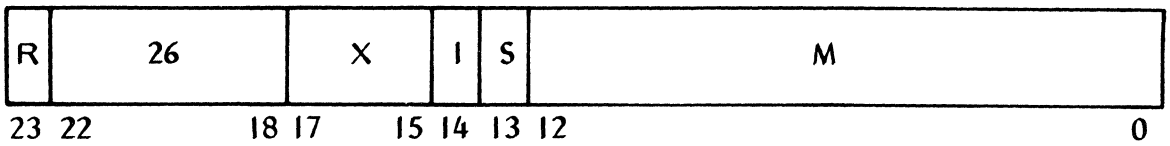
Assembler Format: EOR TEMP1+1

3.9.3 AND Logical And

Definition: (A) AND (Me)→A

Cycles Required: 2

Instruction Format:



Description:

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

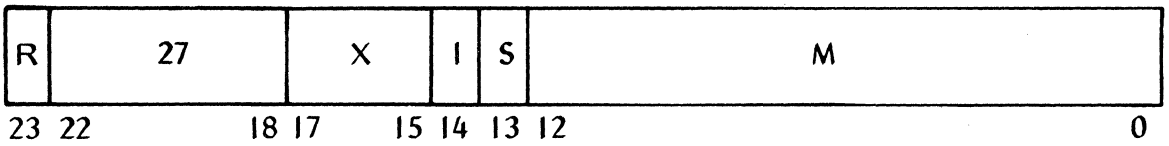
Assembler format: AND TEMP1

3.9.4 OR Logical Or (Inclusive OR)

Definition: (A) OR (M)→A

Cycles Reqired: 2

Instruction Format:



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 B7

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 and bit 14 of the instruction. The eleven 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, the overflow indicator OV, and the parity disable PD. The individual controls for these indicators are the set state and reset state instruction bits O0 through O9 and O14 respectively. If the effective address of the set state (or reset state) instruction has a logical one in the least significant bit, bit O0, SW0 will be set (or reset) by the instruction. If O0 is a logical zero, SW0 will not be changed. SW1 will be set (reset) if a logical one exists in bit O1 of the effective address of the instruction. Any number of the state flip-flops can be set (or reset) with one instruction execution, as follows:

Operand Address Bit	State Flip-Flop Affected
O_0	SW0
O_1	SW1
O_2	SW2
O_3	SW3
O_4	SW4
O_5	SW5
O_6	SW6
O_7	SW7
O_8	IE
O_9	OV
O_{14}	PD

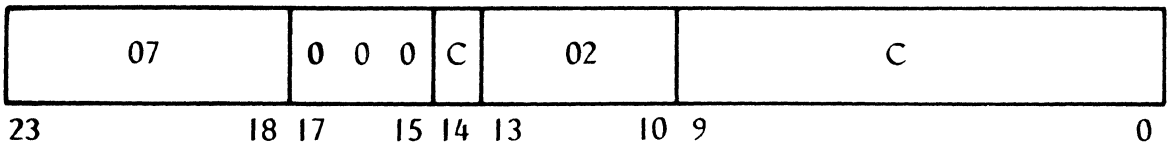
3.10.1 SST Set State

Definition: Set States Defined by C

Cycles Required: 1

Figures 4-66 and 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.

Instruction Format:



Description:

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

Assembler Format: SST SW0, SW1, SW4, OV

NOTE

A special assembler mnemonic exists for setting bit O8 (IE). This is IEN for Interrupt Enable.

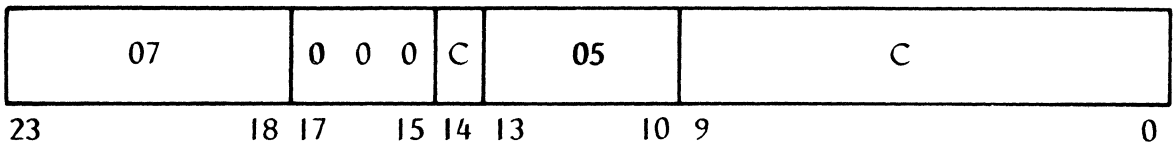
Assembler Format: IEN

3.10.2 RST Reset State

Definition: Reset States Defined by C

Cycles Required: 1

Instruction Format:



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 O8 (IE). This is IDA for Interrupt Disable.

Assembler Format: IDA

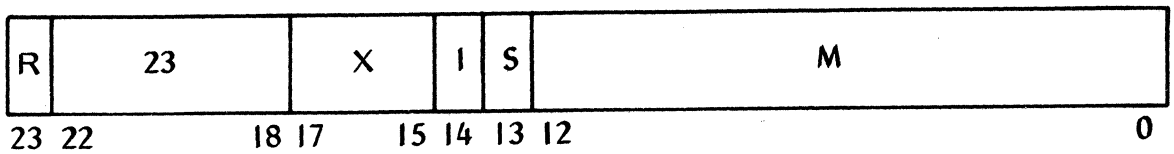
3.11 COMPARE INSTRUCTION

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:



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

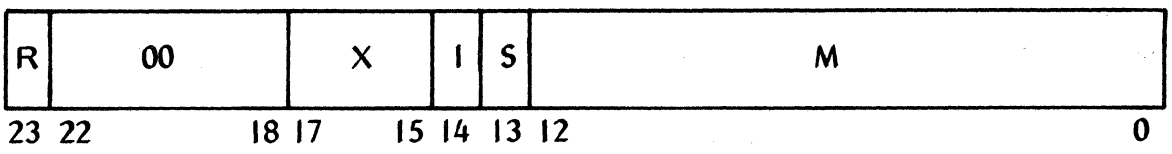
The FST-2 uses 6 instructions which effect transfer of control (or branching). They are: BAH, BRU, BAT, BOI, BOS, and BSM.

3.12.1 BAH Branch After Halt

Definition: Halt and Branch to Me when the CPU START is pressed.

Cycles Required: 1

Instruction Format:



Description:

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

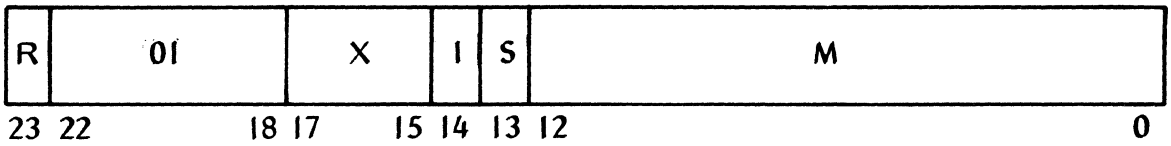
Assembler Format: BAH START2

3.12.2 BRU Branch Unconditionally

Definition: Branch Unconditionally to Me

Cycles Required: 1

Instruction Format:



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 GT, EQ, LT, BE, and OV from bit positions 22, 21, 20, 19, and 18 of the memory location containing the effective address word. For example, if bit position 18 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 a BSM ("Branch, store return at location M") instruction. Note that this restores the five indicators to the states which existed when a BSM instruction was executed.

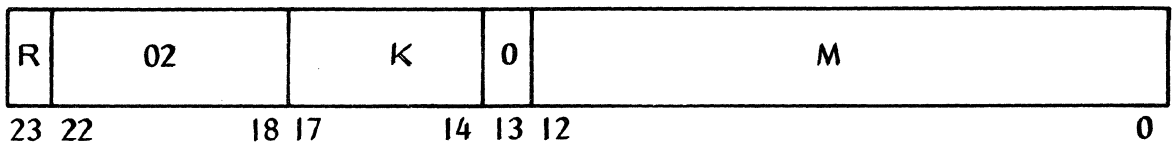
Assembler Format: BRU* STATE+5

3.12.3 BAT Branch A Register Test

Definition: Branch to M on A Register Test

Cycles Required: 1

Instruction Format:



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 = 1000, 1100, 0100, 0110, 0010, 1010, 0001, respectively, (i.e., Branch Positive, Positive or Zero, Zero, Negative or Zero, Negative, Not Equal to Zero, and Odd, respectively).

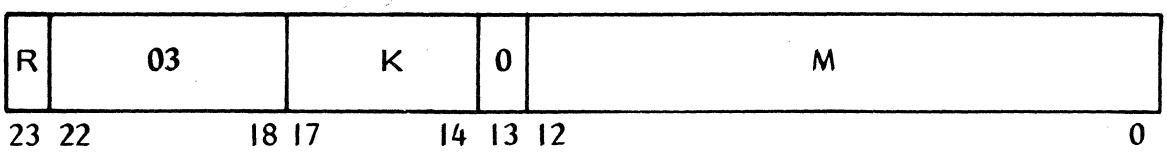
Assembler Format: BPZ TEST2

3.12.4 BOI Branch On Indicator

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

Cycles Reqired: 1

Instruction Format:



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 to 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

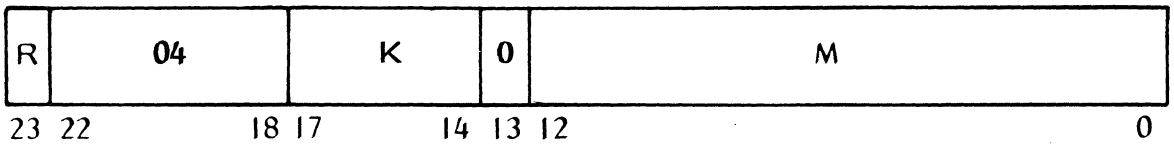
Seven special assembler mnemonics exist to aid the programmer. These are: BG, BGE, BE, BLE, BL, BNE, and BBC for K = 1000, 1100, 0100, 0110, 0010, 1010, 0001, respectively, (i.e., Branch Greater, Greater or Equal, Equal, Less Than or Equal, Less Than, Not Equal, and Bit Compare, respectively).

3.12.5 BOS Branch On State

Definition: Branch to M if State K Set

Cycles Required: 1

Instruction Format:



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

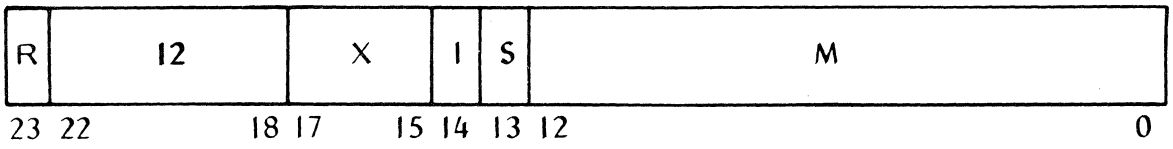
Assembler Format: BOS SW1, LABEL

3.12.6 BSM Branch Store Return At M

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

Cycles Required: 2

Instruction Format:



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 - 17. The states of the five indicators GT, EQ, LT, BE, and OV are stored in memory location Me in bit positions 22, 21, 20, 19, and 18 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

3.13 INPUT/OUTPUT INSTRUCTIONS

This section consists of 1 multifunction instruction: SPU.

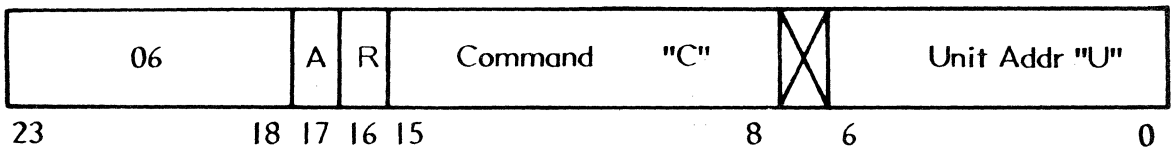
The FST-2 uses only one multifunction I/O instruction, SPU.

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 Section 6 for each 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 Section 6 and Appendix D for interpretation of indicators following an SPU command.

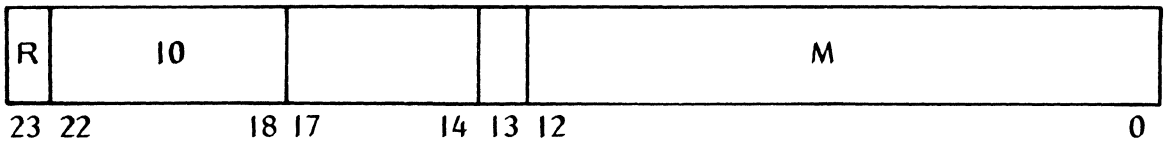
3.14 NO OPERATION INSTRUCTION

3.14.1 NOP No Operation

Definition: No operation

Cycles Required: 2

Instruction Format:



Description:

No operation of any kind will occur on the instruction.

Assembler Format: NOP * + 1

NOTE

Even though it has no operation upon execution, it may be used for indirection or instruction modification. Hence, it is treated as a memory reference instruction by the assembler and all rules and regulations of memory reference instruction apply to NOP.

SECTION 4

FST-2 INSTRUCTION EXECUTION AND TIMING

4.1 INTRODUCTION

This section describes the execution for each of the FST-2 instructions. A timing diagram and data flow diagram for each instruction form the basis for each of the instruction descriptions. Since the logic design for the FST-2 CPU was derived from these diagrams, they constitute the most important form of information necessary to the understanding of the CPU operation. There are several memory reference instructions (MRI) which may be relocatable (see Section 3). When the signal SELR/ is activated, operand bit 23 equals 1. The relocation register is then addded to the operand and the sum becomes the effective memory address. Not all applicable relocatable MRI will show this flow in their respective flow diagrams.

4.2 PHASE TIMER

The Phase Timer for the FST-2 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 4-1 is used to shift the Phase Timer. The system clock consists of negative going pulses 87.5n 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 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 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.

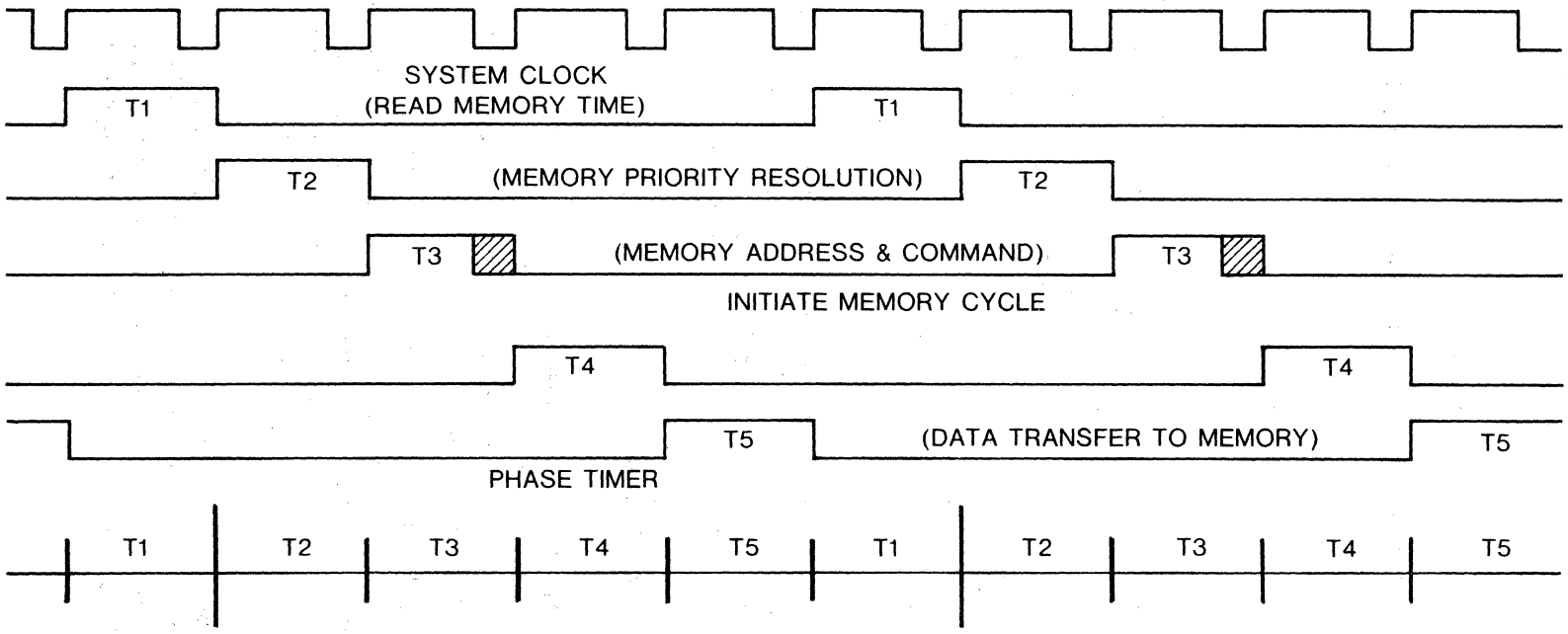


Figure 4-1. System Clock and Phase Timer

4.3 TIMING DIAGRAMS

Timing diagrams are used to illustrate the sequence of events required in the execution of the FST-2 instruction. Figure 4-1 illustrates the relationship between the timing diagram and the occurrence of the phase time signals. Figure 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 4-2 there are five 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 five 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. SELR is either present during all five phases or absent during all five phase times. (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-2 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 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 4-2.)

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 0-19 (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), GO23 (Gate a Clock Pulse to the Relocation bit in operand), GOA1417 (Gate a Clock Pulse to the Alternate operand bits 14 thru 17). This signal transfers zeroes into the alternate operand (bits 14 thru 17). 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.

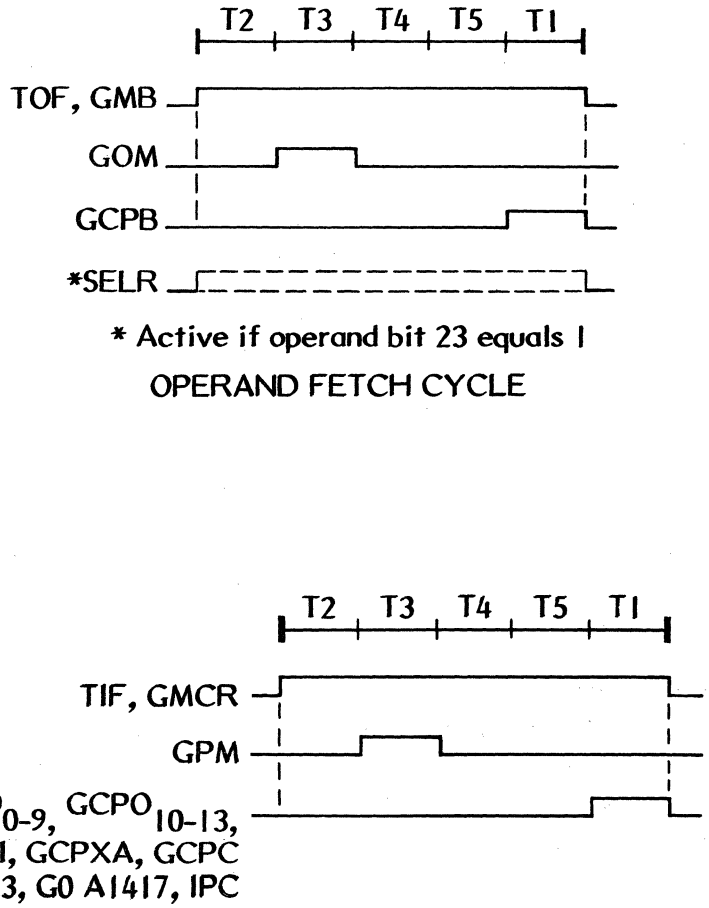


Figure 4-2. Operand and Instruction Fetch Cycle

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-2 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. If the signal SELR is active, operand bit 23 equals 1, the relocation register is then added to the operand and the sum becomes the effective memory address. 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

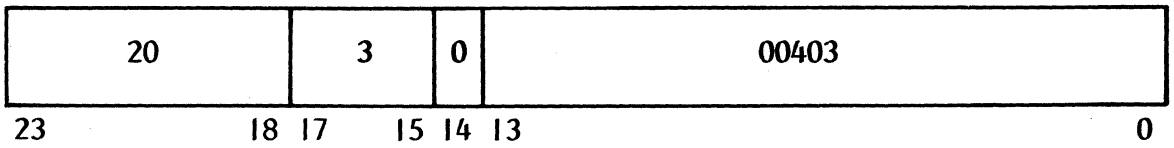
Operand addresses may be modified by indexing in which the contents of a specified index register are added to the operand address before the memory fetch, and by indirection in which the memory fetch produces not data, but another address. The order of modification is first index, then indirection.

4.4.3.1 INDEXING. Index register modification of an address is available to all commands with the exception of those which use bits 15, 16, or 17 for a purpose other than indexing, such as: 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), LDX (Load Index), LRA (Load RR from A), LAR (Load A from RR), and CLA (Clear Accumulator).

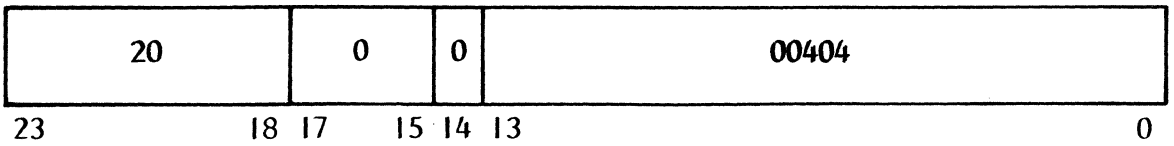
An indexing operation is specified and directed by bits 17, 16, and 15 of the command register word. These bits indicate the address of the specific index register to be used (1 through 7). If bits 17, 16, and 15 are a "0", indexing has not been specified. When any one or more of these three bits is 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 Figures 4-3 and 4-3A).

Prior to the execution of an indexable instruction, the contents of the index register specified by bits 15, 16, and 17 of the instruction word are added to the operand address field of the instruction word. The resultant sum replaces the original operand in the command register.

For example, if index register 3 contains 00001 and the command register contains the following instruction:



the 00403 and 00001 are added together, the sum replacing the 00403, so that the instruction in the command register changes to:



The instruction shown immediately above is the instruction actually executed. Note that the index address has been changed to 0. Index address 0 is interpreted as "no indexing" rather than specifying index register 0. It follows then that INDEX REGISTER 0 CANNOT BE SPECIFIED FOR ADDRESS MODIFICATION.

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), which generates control signal GBUB, Figure 4-3. 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), which generates control signal GUA, 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 GCPO 0-9 (Gate Clock Pulse to Operand field, 0 through 9) and GCPO 10-13 (Gate Clock Pulse to Operand field bits 10 through 13) as shown in Figure 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-2 Commands with the exception of SPU, BAT, BOS, and BOI and all augmented commands. 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 exceptions listed above.

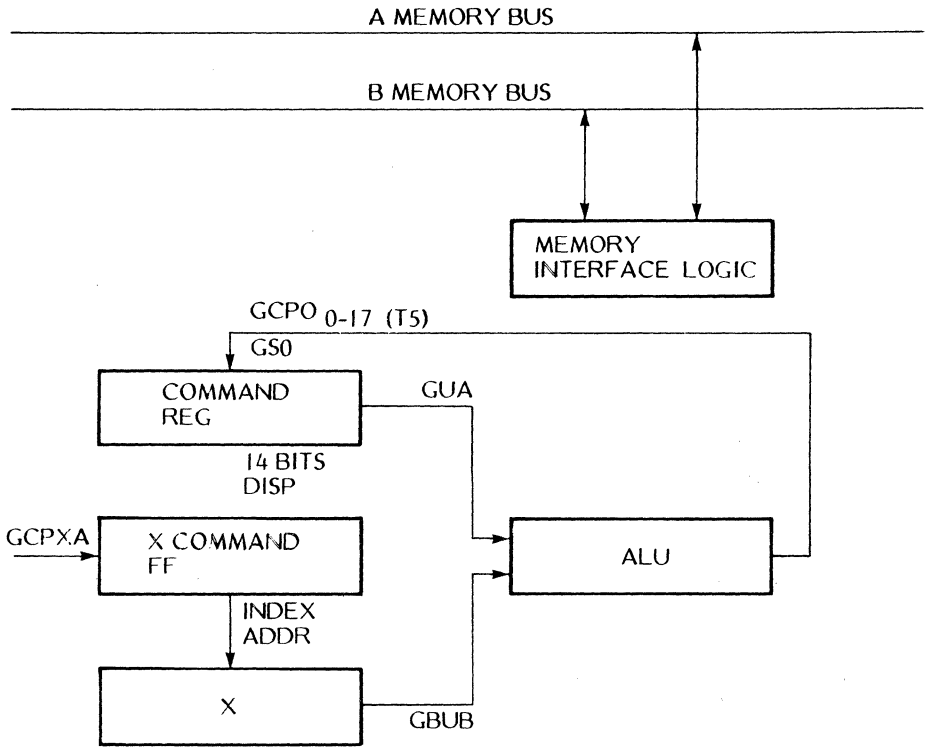


Figure 4-3. Indexing Flow Diagram

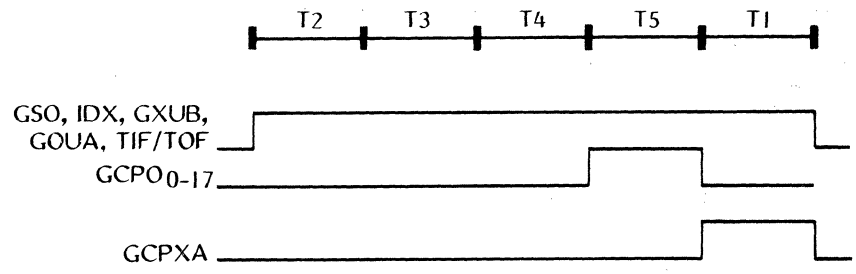
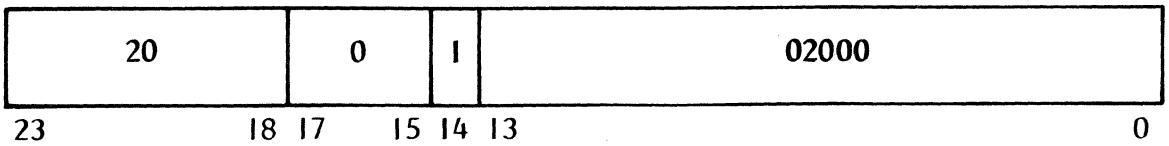


Figure 4-3A. Indexing Timing Diagram

(See Figures 4-4 and 4-4A). 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 4-4A), 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 GCPO 0-9, GCPO 10-13, GCPI (Gate Clock Pulse to Indirect bit), GCP 14-17 (Gate Clock Pulse Alternate Operand bits 14 thru 17), GCPO23 (Gate Clock Pulse to relocation bit in operand), clock the data from memory into the command register. Note that the original command (bits 22 through 18) is not altered.

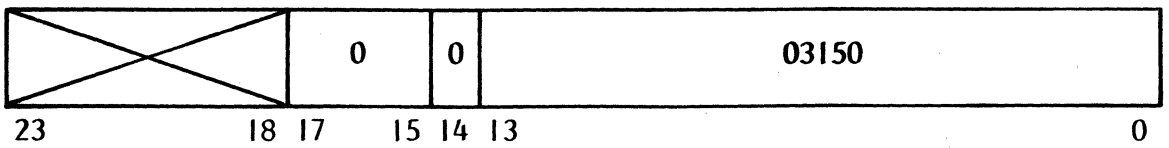
For example:



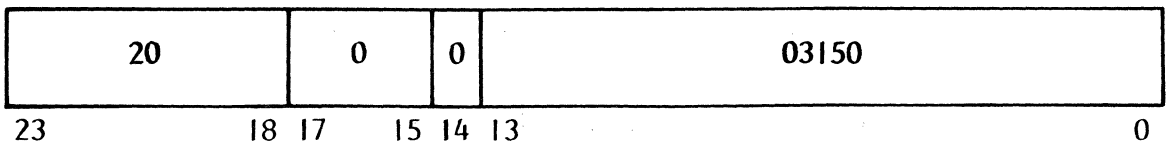
is a typical instruction specifying indirect address modification. The indirect address modification operation occurs as follows:

The instruction operand address is used to fetch a new operand address, indirect flag and index address bits from memory.

For example, if memory location 02000 contained



then the above ADD instruction would be equivalent to:



The FST-2 always performs indexing prior to indirect address modification. Only one level of indirection is allowed.

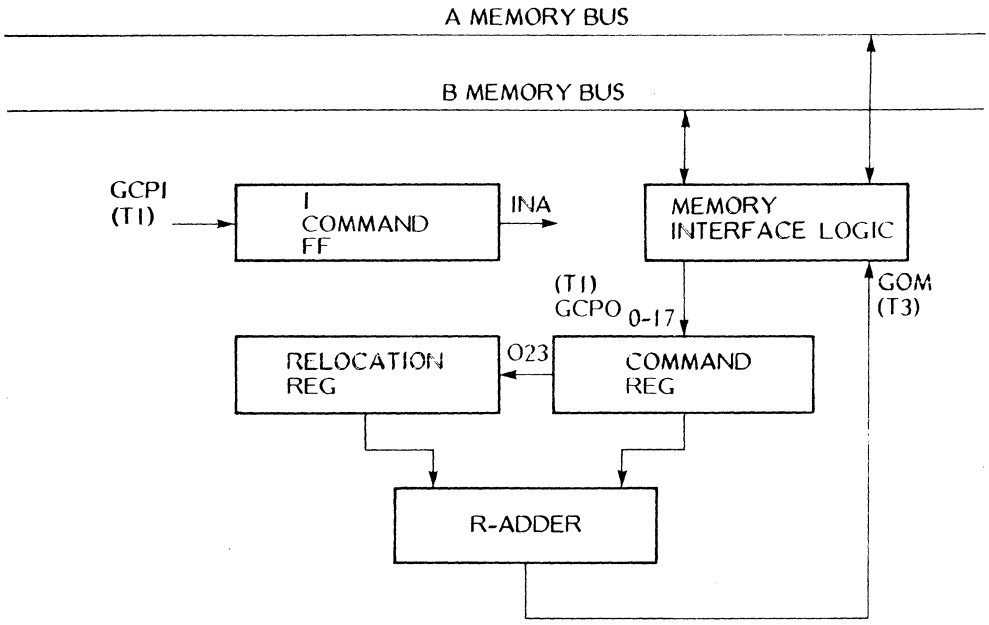
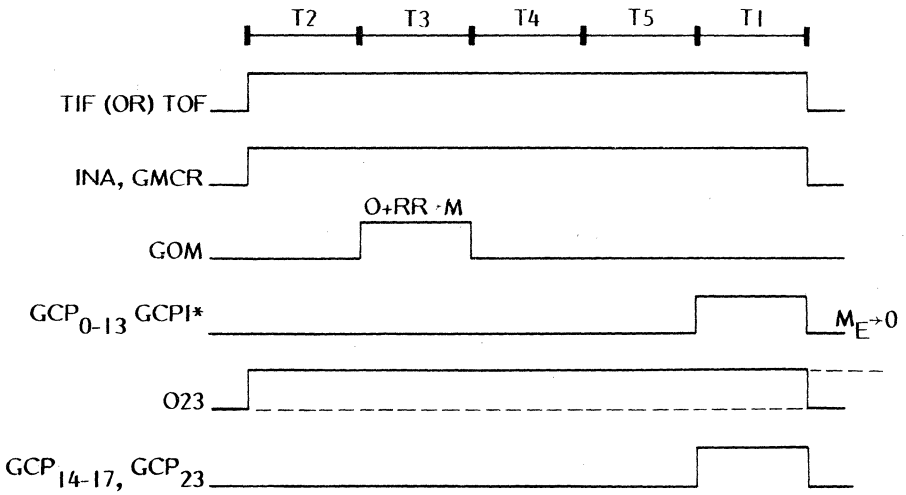


Figure 4-4. Indirect Address Modification Flow Diagram



* CLOCK M18-M23 TO STATUS REG IF INSTRUCTION IS BRU INDIRECT
(M18→OV,M19→BE,M20→LT,M21→EQ,M22→GT)

Figure 4-4A. Indirect Address Modification Timing Diagram

4.5 INSTRUCTION EXECUTION AND TIMING

The execution sequence for each instruction in the FST-2 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. However, the tables given for LDA and ATX cover indexing, indirection, and relocation.

4.5.1 ADD Instruction

Figures 4-5 and 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 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 4-7 and 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 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.

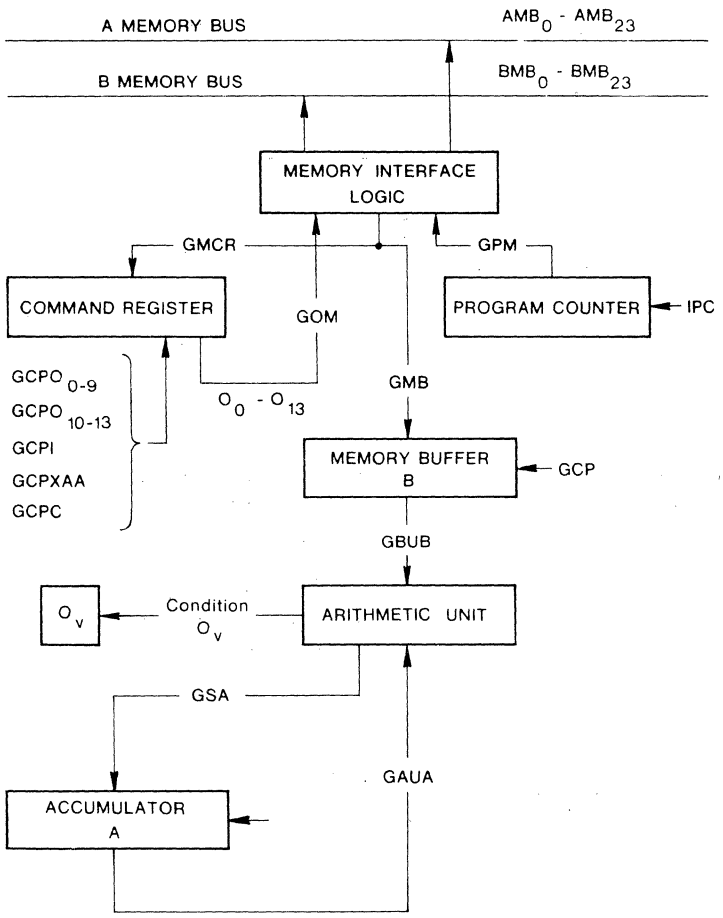


Figure 4-5. ADD Instruction Flow Diagram

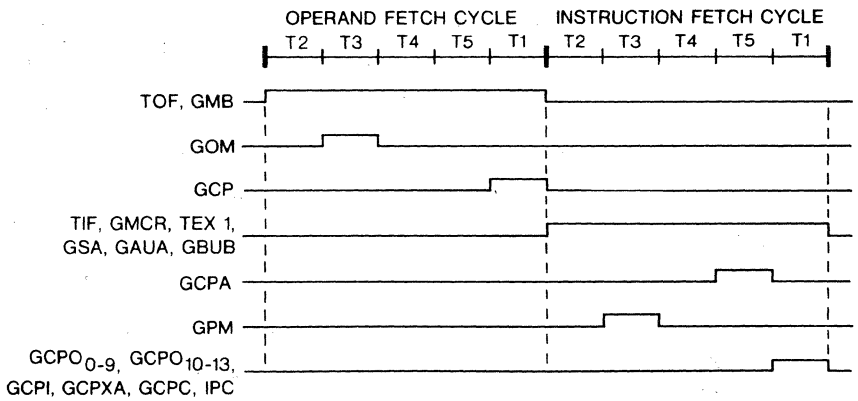


Figure 4-6. ADD Instruction Timing Diagram

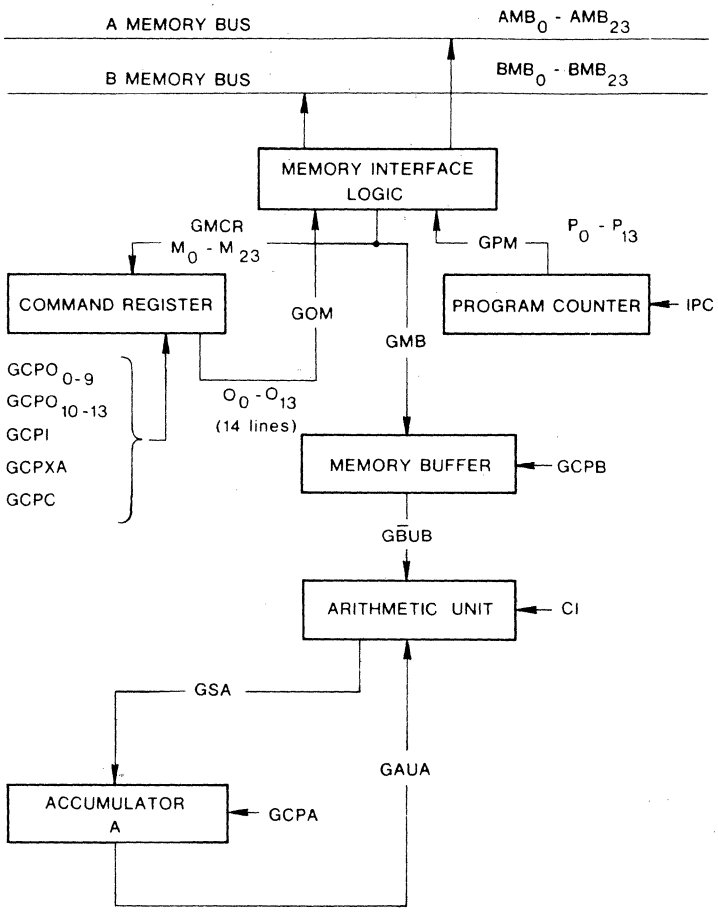


Figure 4-7. SUB Instruction Flow Diagram

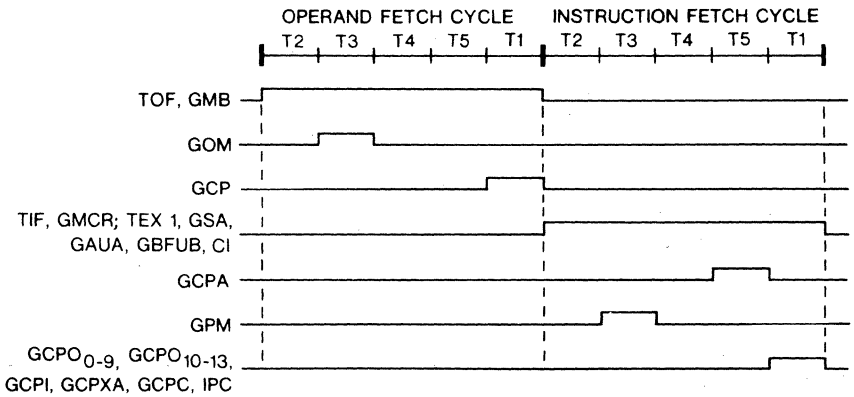


Figure 4-8. SUB Instruction Timing Diagram

4.5.3 EOR Instruction

Figures 4-9 and 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 4-11 and 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 resulting in 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.

4.5.5 LDA Instruction

Figures 4-13 and 4-14 are the flow and timing diagrams for the LDA instruction. Table 4-1 gives details of indexed, indirect, and relocated commands not described here. 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 4-15 and 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) clocks the operand from AU into E.

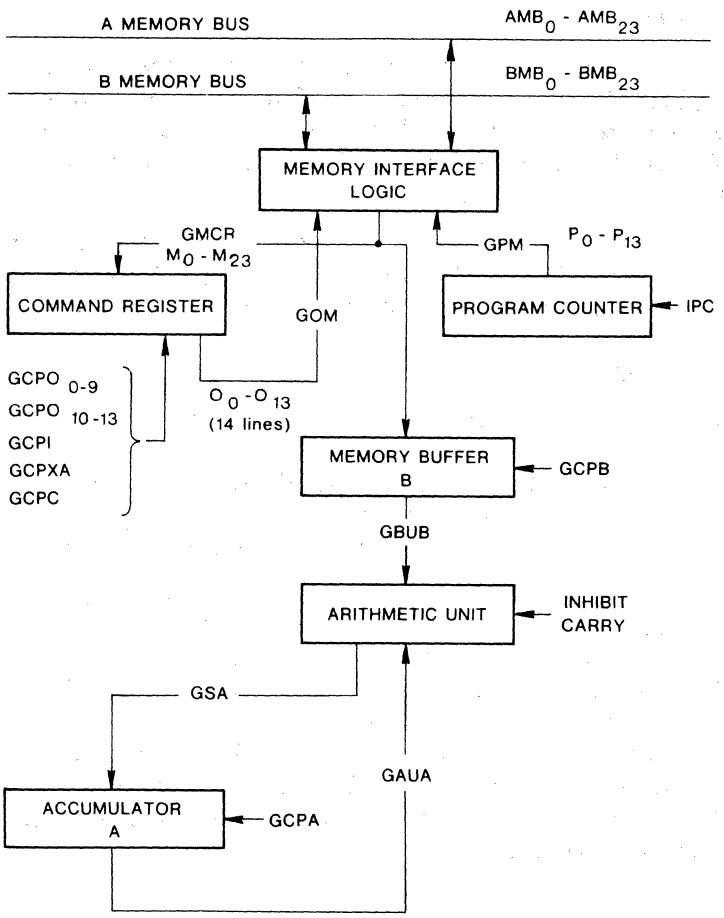


Figure 4-9. EOR Instruction Flow Diagram

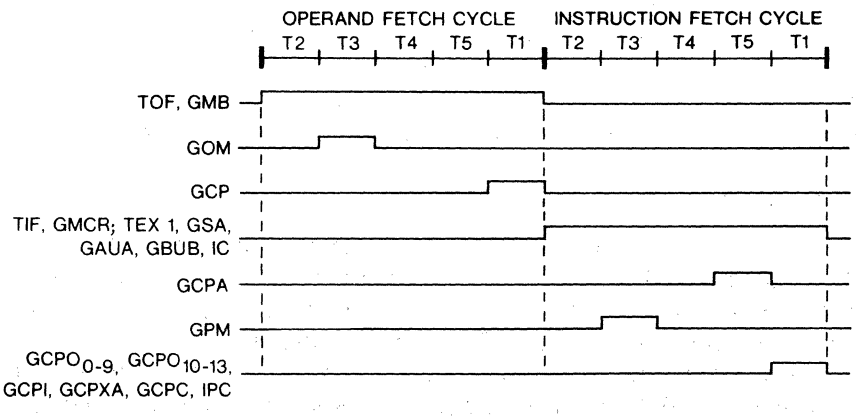


Figure 4-10. EOR Instruction Timing Diagram

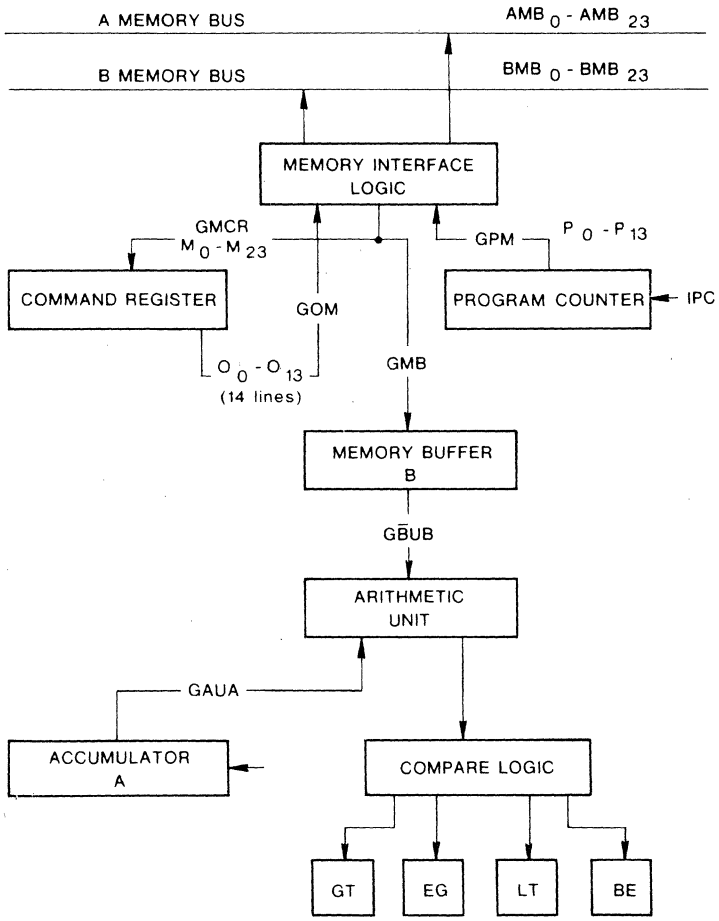
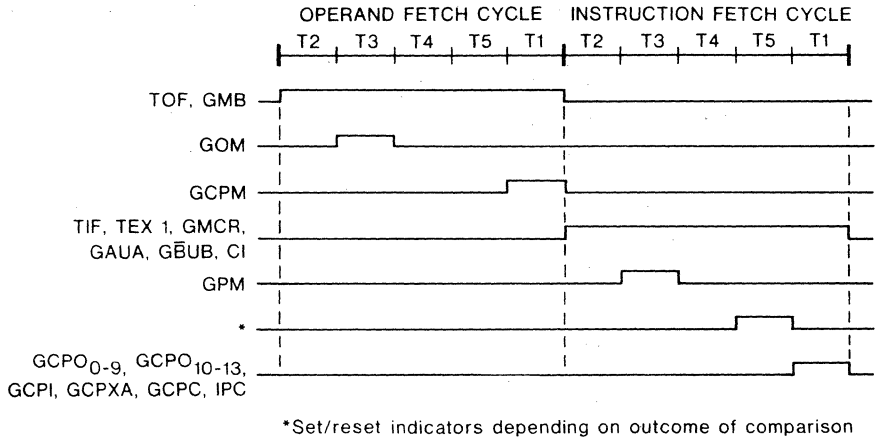


Figure 4-11. CAM Instruction Flow Diagram



*Set/reset indicators depending on outcome of comparison

Figure 4-12. CAM Instruction Timing Diagram

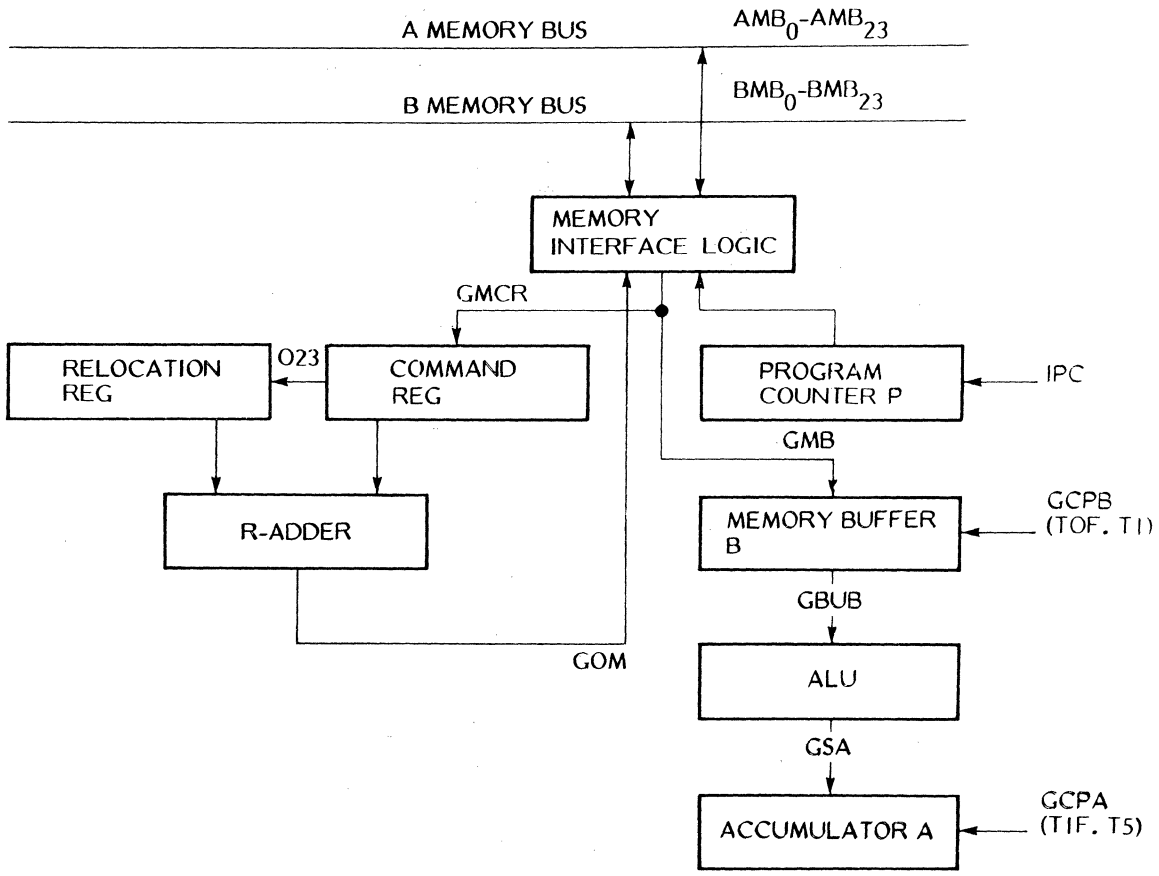


Figure 4-13. LDA Instruction Flow Diagram

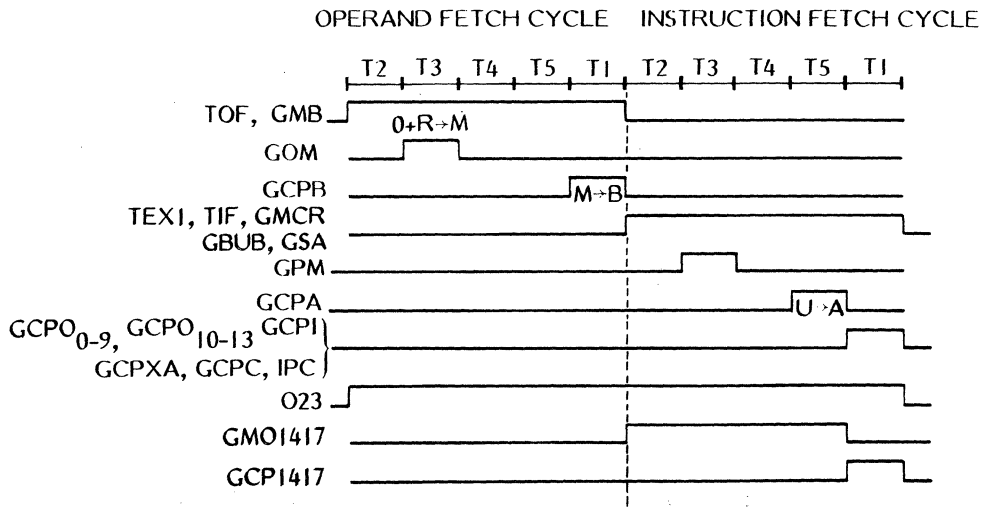


Figure 4-14. LDA Instruction Timing Diagram

TABLE 4-1 LDA INSTRUCTION DETAILS

	OPCODE	FST-2 MODE	FST-1 MODE
1	240(<4)XXXX 2 CYCLES DIRECT	RR INHIBITED SELR/=1	RR INHIBITED SELR/=1 SEE NOTE 2
2	241(<4)XXXX 3 CYCLES INDEXED	RR INHIBITED SELR/=1 SEE NOTE 1	RR INHIBITED SELR/=1
3	240(≥4)XXXX M23=0 3 CYCLES INDIRECT	RR INHIBITED	RR INHIBITED MULT IND O.K.
4	240(≥4)XXXX M23=1 3 CYCLES INDIRECT	TOF CYCLE: O+RR→M SELR/=0	SAME AS 3
5	241(≥4)XXXX M23=0 4 CYCLES INDEXED INDIRECT	INDEX CYCLE: SEE NOTE 1 TOF CYCLE: SELR/=1 O+0→M	RR INHIBITED MULT IND O.K.
6	241(≥4)XXXX M23=1 4 CYCLES INDEXED INDIRECT	INDEX CYCLE: SEE NOTE 1 TOF CYCLE: SELR/=0 O+RR→M	RR INHIBITED MULT IND O.K.
7	640(<4)XXXX 2 CYCLES RELOCATED	TOF CYCLE: SELR/=0 O+RR→M	ILLEGAL

TABLE 4-1 LDA INSTRUCTION DETAILS (Continued)

	OPCODE	FST-2 MODE	FST-1 MODE
8	641(<4)XXXX 3 CYCLES RELOCATED INDEXED	INDEX CYCLE: SEE NOTE 1 TOF CYCLE: SELR/=0 O+RR→M	ILLEGAL
9	640(≥4)XXXX M23=0 3 CYCLES RELOCATED INDIRECT	INDIRECT CYCLE: SELR/=0 RR+O→M, M_E→O O23=0 TOF CYCLE: SELR/=1 O+0→M	ILLEGAL
10	640(≥4)XXXX M23=1 3 CYCLES RELOCATED INDIRECT	INDIRECT CYCLE: SELR/=0 RR + O→M, M_E→O O23 = 1 TOF CYCLE: SELR/=0 O + RR→M	ILLEGAL
11	641 (≥4)XXX M23=0 4 CYCLES RELOCATED INDEXED INDIRECT	INDEX CYCLE: SEE NOTE 1 INDIRECT CYCLE: SELR/=0 M_E→O: O23=0 TOF CYCLE: SELR/=1 O+0→M; M_E→B→A	ILLEGAL
12	641(≥4)XXXX M23 = 1 4 CYCLES RELOCATED INDEXED INDIRECT	INDEX CYCLE: SEE NOTE 1 INDIRECT CYCLE: SELR/=0 R+O→M M_E→O; O23=1 TOF CYCLE: SELR/=0	

NOTES:

1. IF O13 = 1 DURING THE INDEX CYCLE, CPU PADS 1'S TO BITS 14-17 THRU MUXA I2 CAUSING NEGATIVE INDEXING.

   GUA03/ = 1
GUA13/ = 0
GUA23/ = 1 } SELECTED MUXA I2

   GM1UAA/ = 0 FORCE 1'S TO BIT 14-17

2. IN FST-1 MODE
   O14-O17 REMAIN ZERO BECAUSE
   GMO1417/ = 1 (GATE CONT 1 I/O G0)
   THEREFORE, THE LDA OPERAND IS LIMITED TO 14 BITS.

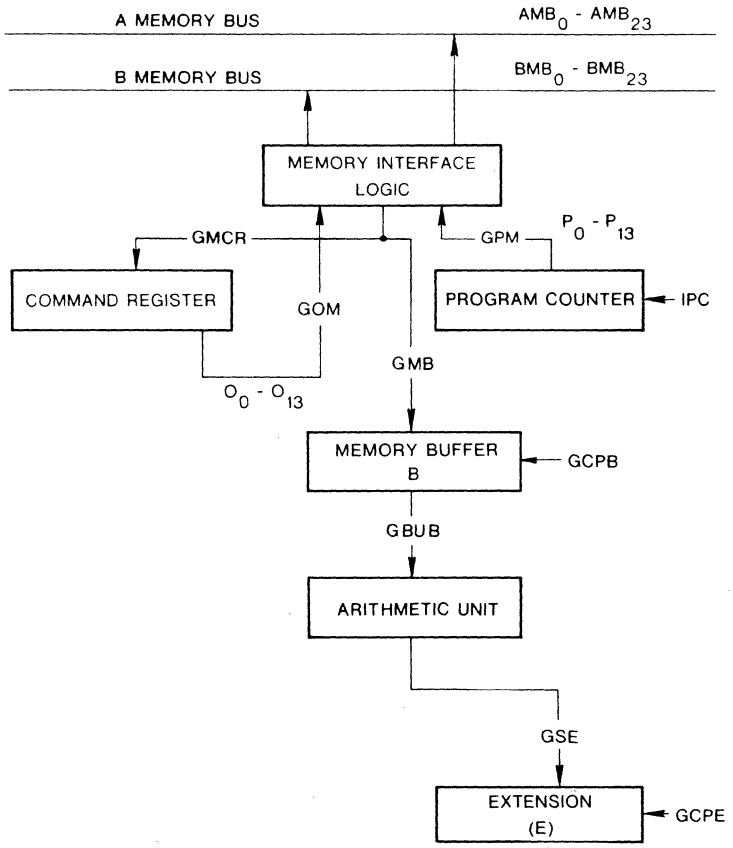


Figure 4-15. LDE Instruction Flow Diagram

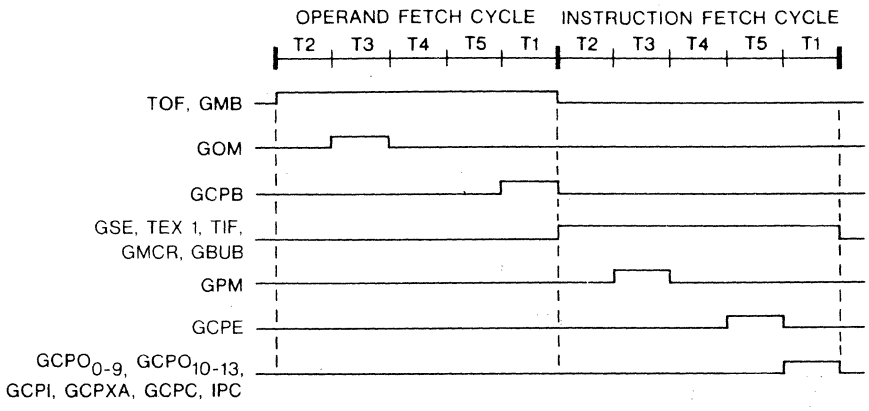


Figure 4-16. LDE Instruction Timing Diagram

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 4-17 thru 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 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.

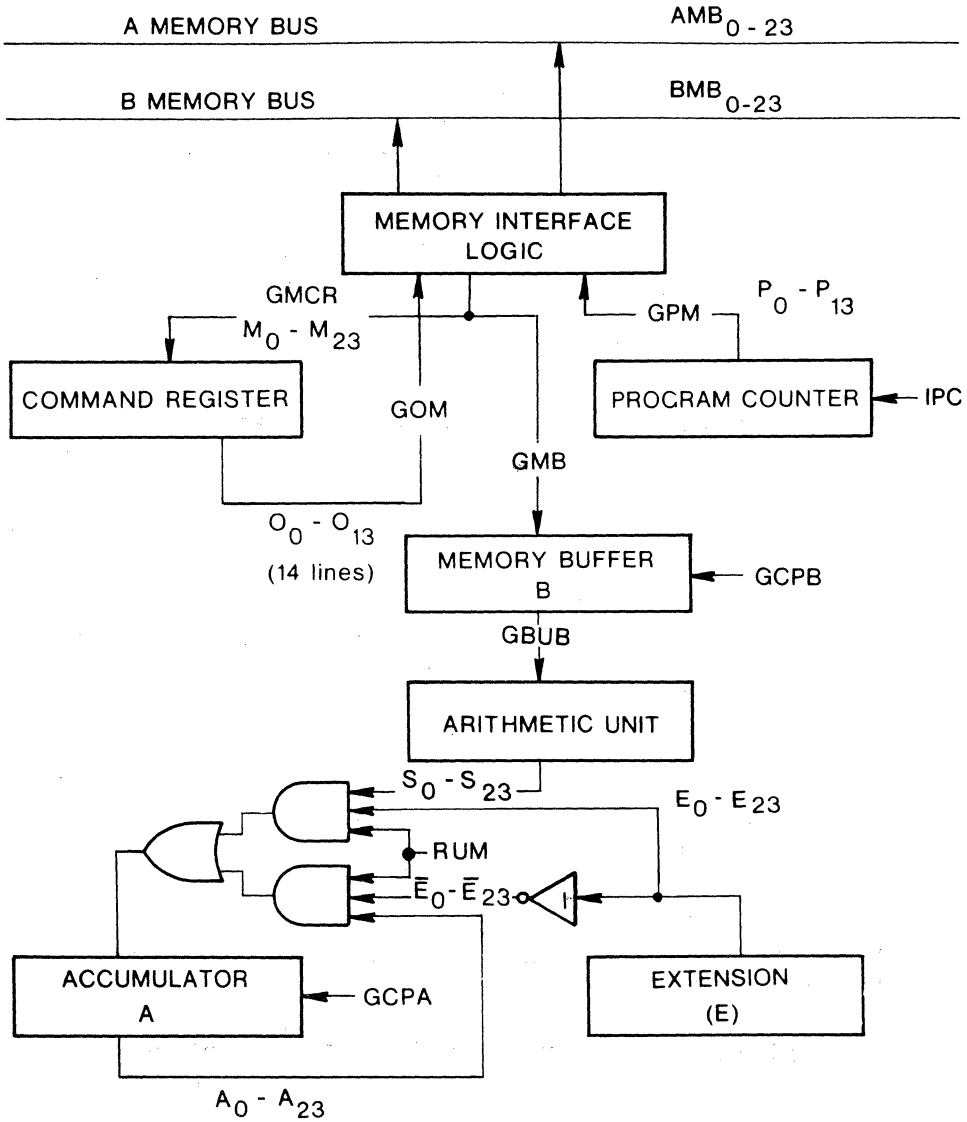


Figure 4-17. RUM Instruction Flow Diagram

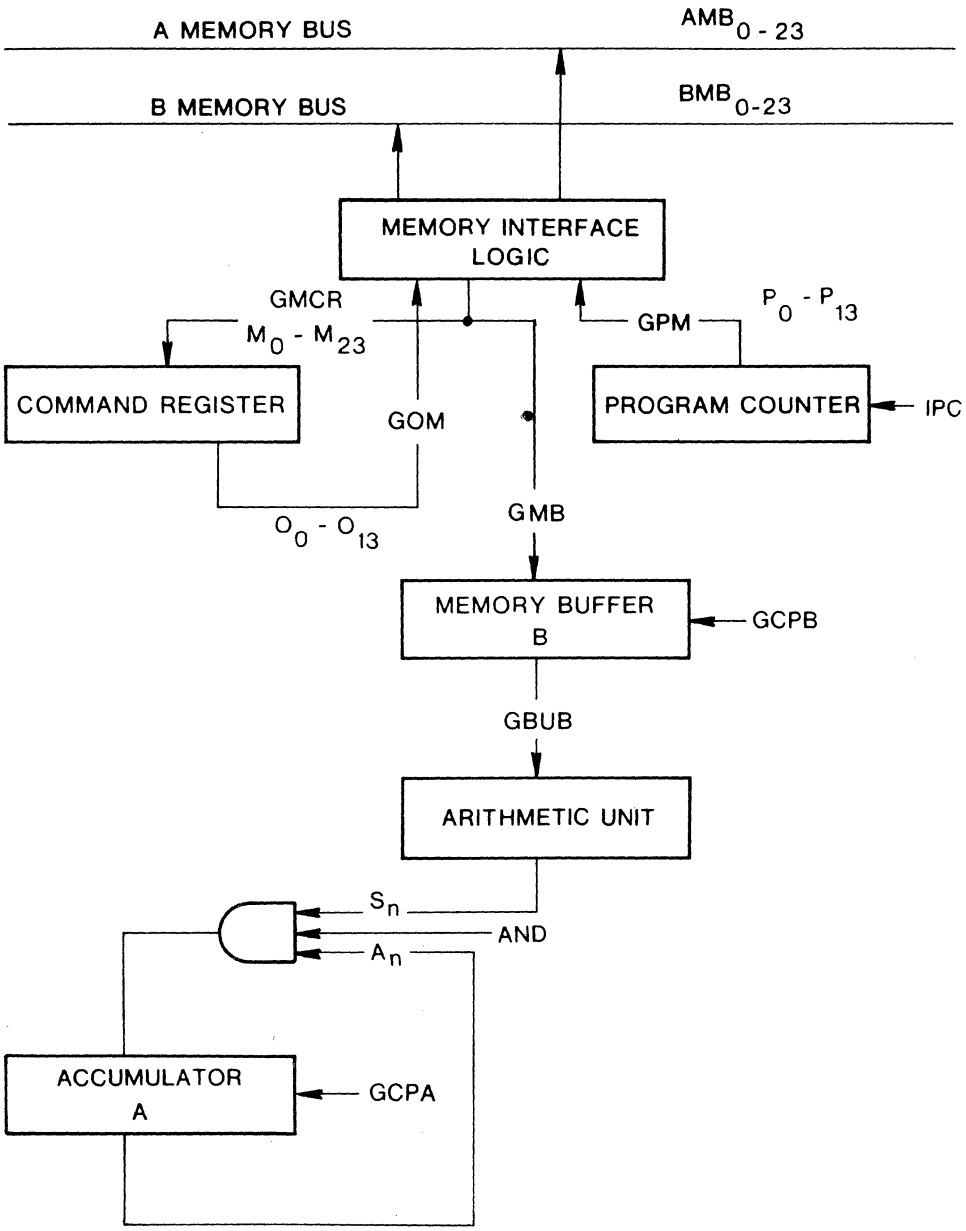


Figure 4-18. AND Instruction Flow Diagram

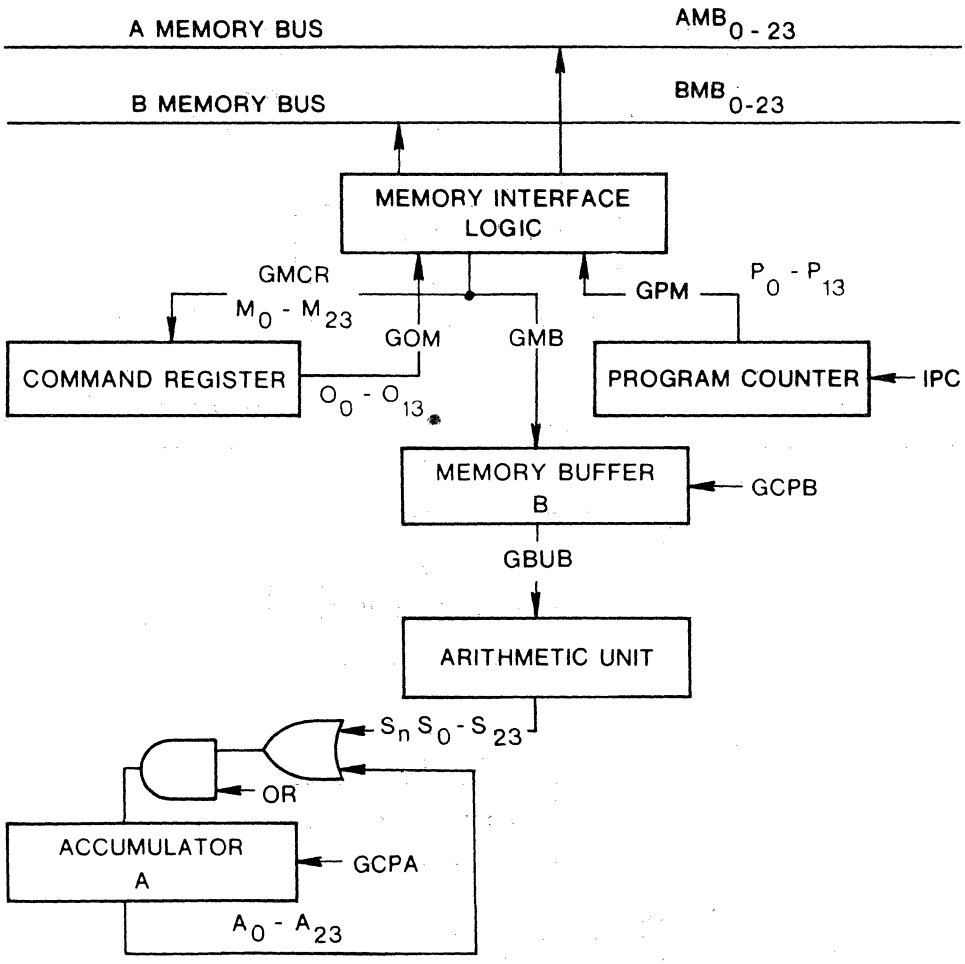


Figure 4-19. OR Instruction Flow Diagram

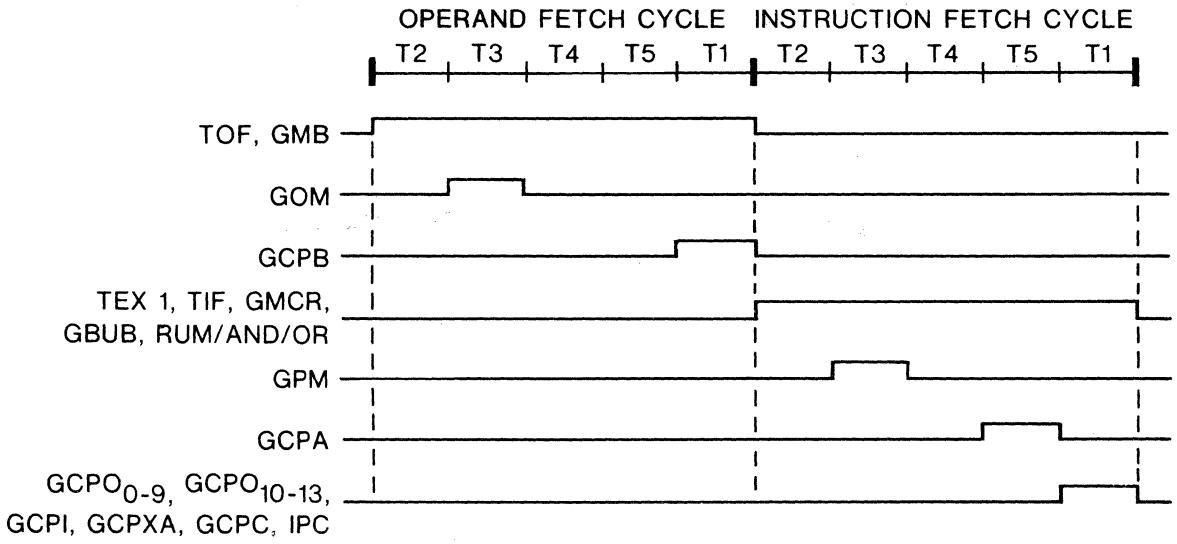


Figure 4-20. RUM/AND/OR Instruction Timing Diagram

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 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 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 4-21 and 4-22 are the flow and timing diagrams for the ATX instruction. Table 4-2 gives details of indirect and relocated commands not described here. 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 GCPO 0-9 and GCPO 10-13 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 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.

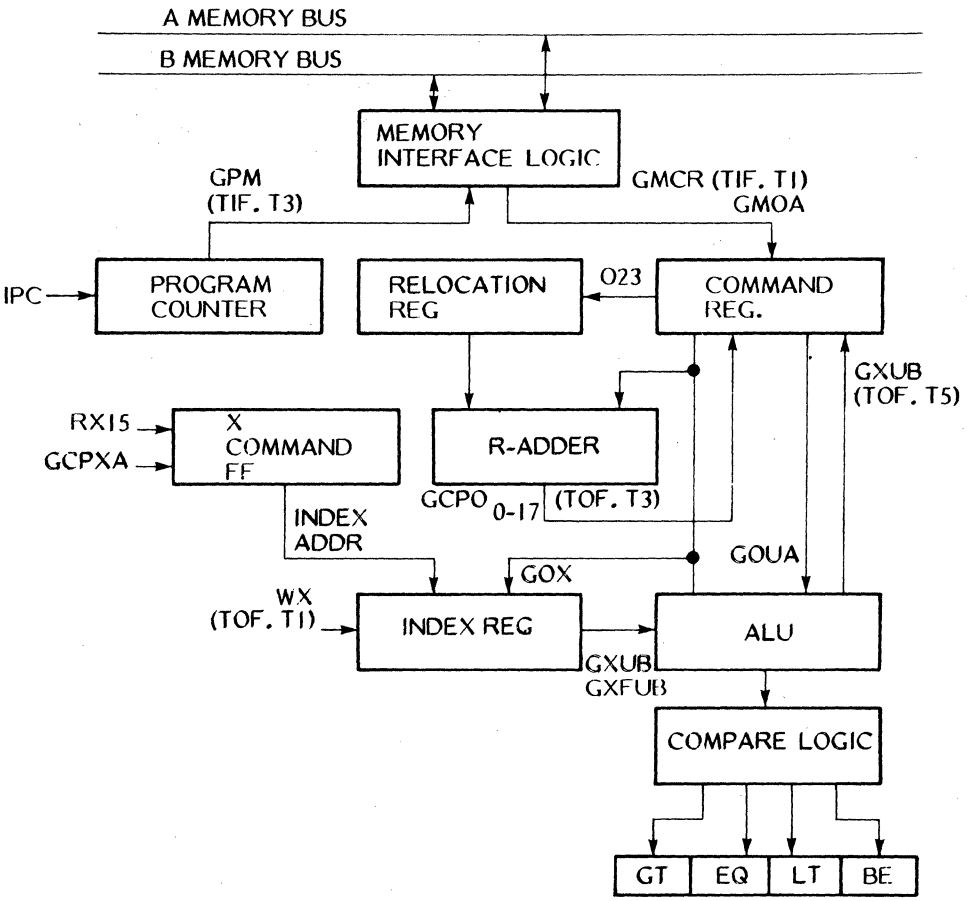


Figure 4-21. ATX Instruction Flow Diagram

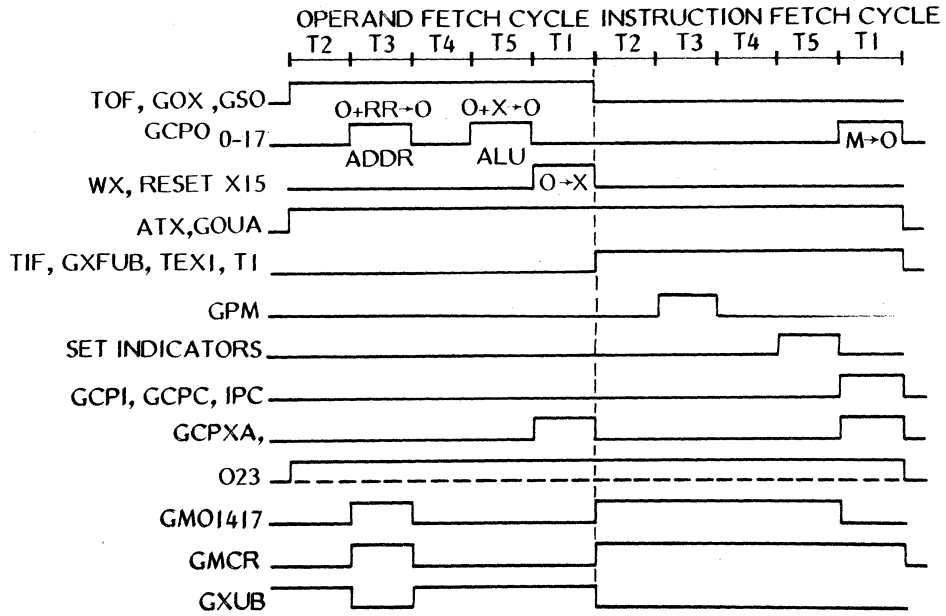


Figure 4-22. ATX Instruction Timing Diagram

TABLE 4-2 ATX INSTRUCTION DETAILS

	OPCODE	FST-2 MODE	FST-1 MODE
1	$[111(<4)XXXX]_8$ 2 CYCLES DIRECT	RR INHIBITED BIT 13 = 1 SIGNIFIES NEGATION SEE NOTE 1	RR INHIBITED SEE NOTE 2
2	$[111(\geq 4)XXXX]_8$ (M23=0) 3 CYCLES INDIRECT	RR INHIBITED 18 BIT OPERATION X1 + $M_E \rightarrow$X1 INS DEC GUA03/=GUA01/=0 18 BIT OPERATION NO NEGATIVE INDEXING	RR INHIBITED SAME AS 1 MULT-IND O.K.
3	$[111(\geq 4)XXXX]_8$ (M23=1) 3 CYCLES INDIRECT	IND CYCLE: RR INHIBITED $M_E \rightarrow 0$; SET O23=1 TOF CYCLE: O + RR→O X1+O→X1 GUA03/=GUA01/=0 18 BIT OPERATION NO NEGATIVE INDEXING	RR INHIBITED SAME AS 1 MULT-IND O.K.
4	$[511(\geq 4)XXXX]_8$ M13=0 2 CYCLES RELOCATED M13=1	RR SELECTED O + RR→O INS DECODE O23=0,D7-3=0, NO NEGATIVE ADDRESSING NOT APPLICABLE, ILLEGAL	ILLEGAL
5	$[511(\geq 4)XXXX]_8$ (M23=0) 3 CYCLES RELOCATED INDIRECT	IND CYCLE: RR SELECTED O + RR→M; $M_E \rightarrow$O O23→0 TOF CYCLE: RR INHIBITED X1+O→X1; GUA03/=GUA01/=0 SEE NOTE 1A	ILLEGAL
6	$[511(\geq 4)XXXX]_8$ (M23=1) 3 CYCLES RELOCATED INDIRECT	IND CYCLE: RR SELECTED O+RR→M; $M_E \rightarrow$O; O23=1 TOF CYCLE: RR SELECTED O+RR→O, X1+O→X1 GUA03/=GUA01/=0 18 BIT OPERATION SEE NOTE 1A	ILLEGAL

TABLE 4-2 ATX INSTRUCTION DETAILS (Continued)

NOTES:

1.A. FST-2, DIRECT, O13 = 0

TOF CYCLE: SELECT MUXA I_3

GUA01/=GUA03/=0

GUA11/=GUA13/=0

GUA21/=GUA23/=1

SELECT MUXB I_2

GBUB/=0, GBFUB/=1

14-BIT-OPERAND IS ADDED TO 18-BIT INDEX (ODD) THRU ALU. 18-BIT SUM IS CLOCKED BACK INTO OPERAND.

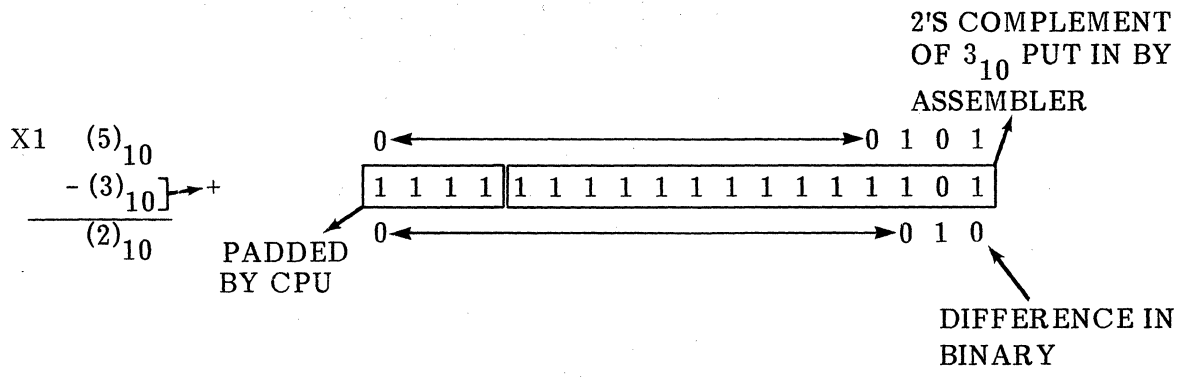
1.B. FST-2, DIRECT, O13 = 1 - NEGATIVE INDEXING

GUA03/=1
GUA13/=0 } SELECT MUXA I_2
GUA23/=1

GM1UAA/=0 FORCE 1'S TO X14-X17

X1(18 BITS) + $O_{INITIAL} \rightarrow O_{SUM}$

EXAMPLE



2. IN FST-1 MODE ATX WORKS AS NOTE 1.A. EXCEPT O14-O17 REMAIN 0'S (GMO1417/=1) AND THEREFORE X14-X17 ARE ALWAYS 0'S.

4.5.9 DTC Instruction

Figures 4-23 and 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 4-25 and 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.

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 LAX Instruction

Figures 4-27 and 4-28 are the flow and timing diagrams for the LAX instruction. Table 4-3 gives details not described here. The LAX instruction copies the least significant 18 bits of the selected Index Register, puts them into the accumulator, and requires one memory cycle for execution. The X to A data transfer overlaps the next Instruction Fetch cycle. The decoded command signal generates GXA which controls the gating of X to A. The contents of X0 through X 17 are loaded into A at T5 time with the gate clock pulse to Accumulator signal (GCPA).

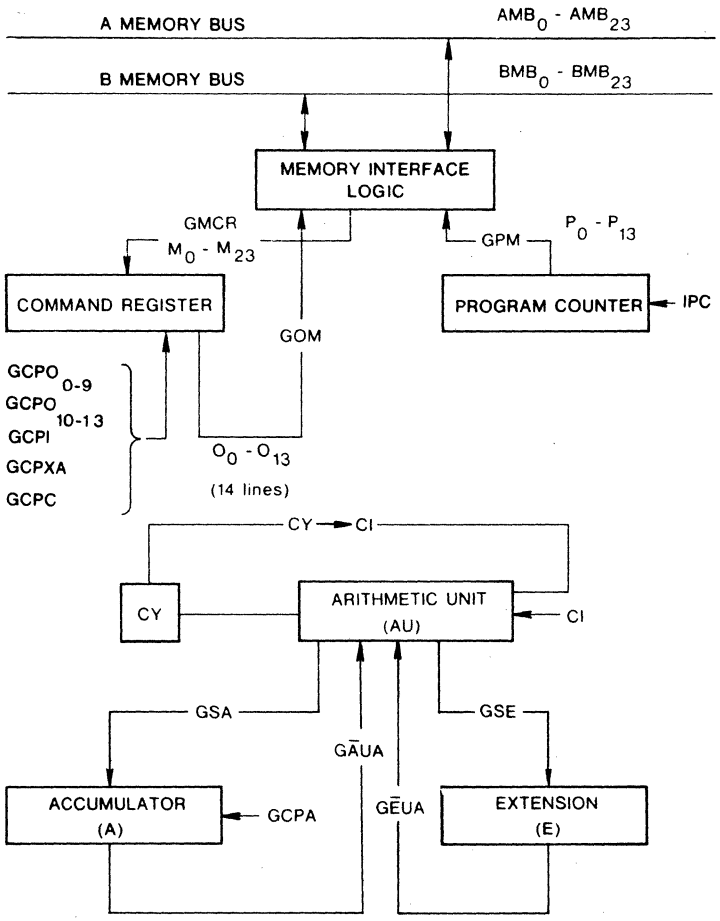


Figure 4-23. DTC Instruction Timing Diagram

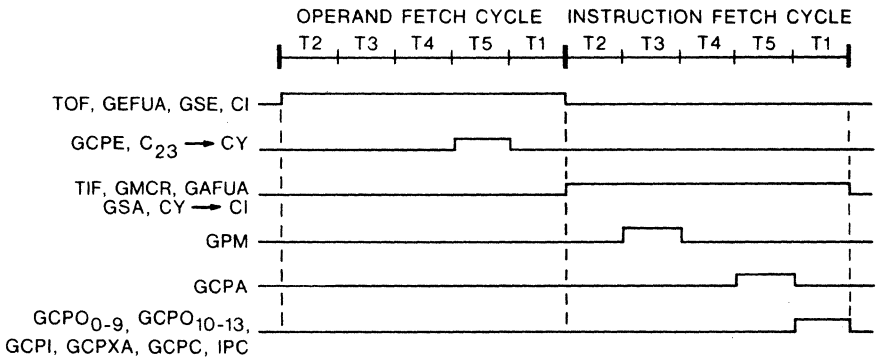


Figure 4-24. DTC Instruction Timing Diagram

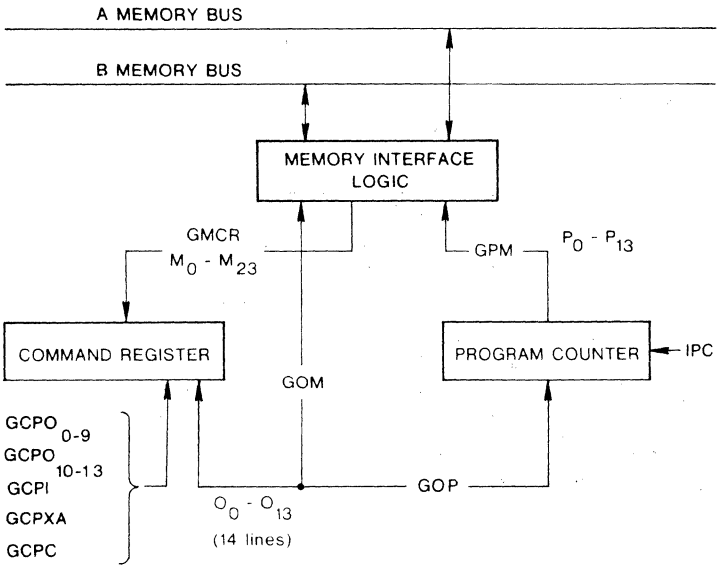


Figure 4-25. BSM Instruction Flow Diagram

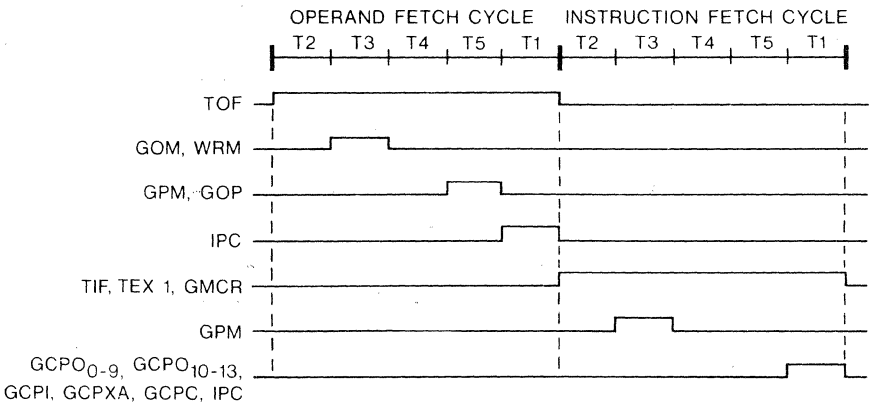


Figure 4-26. BSM Instruction Timing Diagram

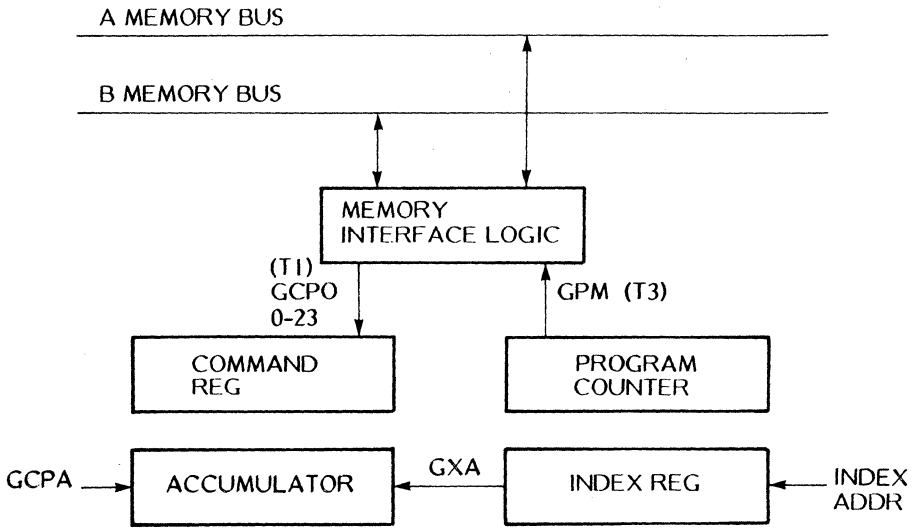


Figure 4-27. LAX Instruction Flow Diagram

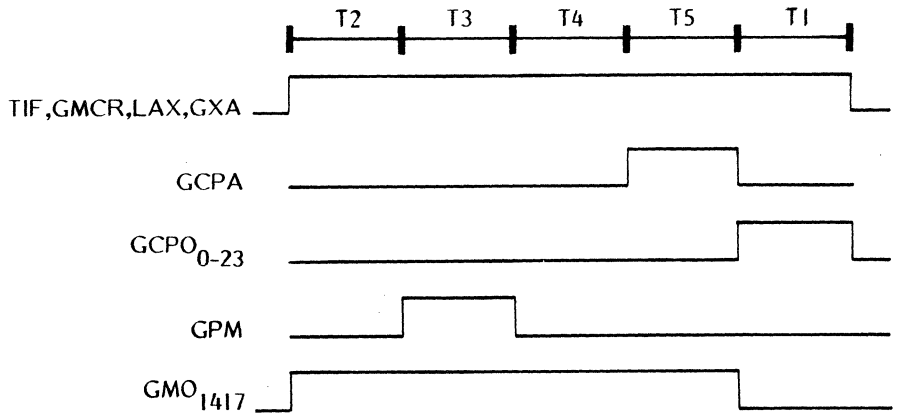


Figure 4-28. LAX Instruction Timing Diagram

TABLE 4-3 LAX INSTRUCTION DETAILS

	OPCODE	FST-2 MODE	FST-1 MODE
1	$(07100200)_8$ 1 CYCLE	18 BIT OPERATION	OPCODE IS 40100000 SEE NOTE 1. 18 BIT OPERATION. X14–X17 ASSUMED 0'S.

NOTES:

1. FST-1/FST-2 OPCODE MUXING AT CONFF I/O B5

2. INS DEC I/O C1, C9

4.5.12 STA Instruction

Figures 4-29 and 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 are gated to memory. The second memory cycle of STA is a normal instruction fetch cycle.

4.5.13 STE Instruction

Figures 4-31 and 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 4-33 and 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 GCPO 0-9 and GCPO 10-13 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.

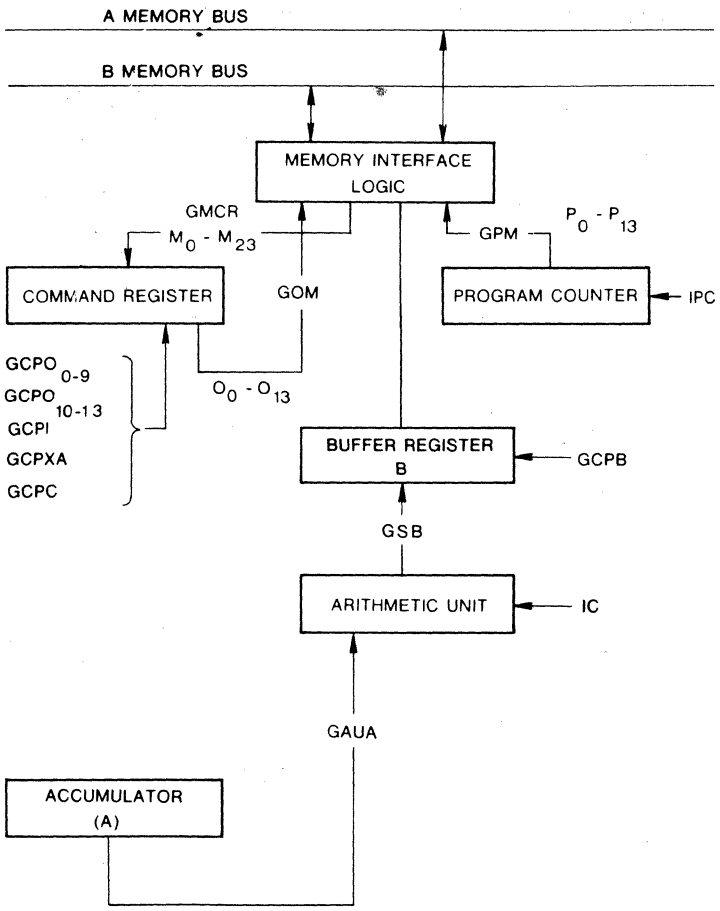


Figure 4-29. STA Instruction Flow Diagram

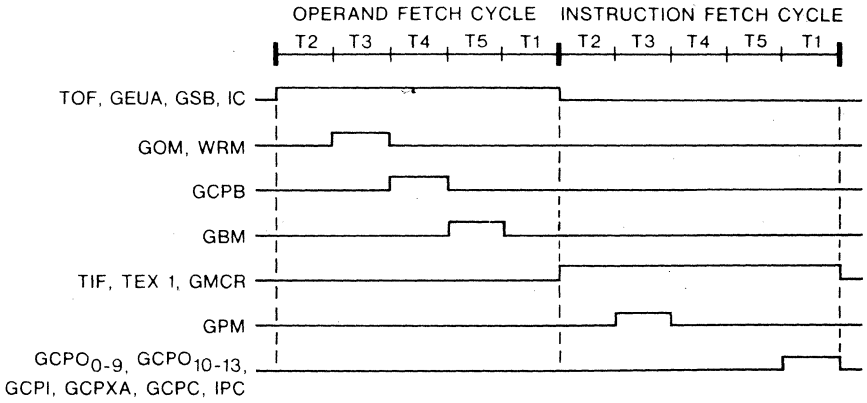


Figure 4-30. STA Instruction Timing Diagram

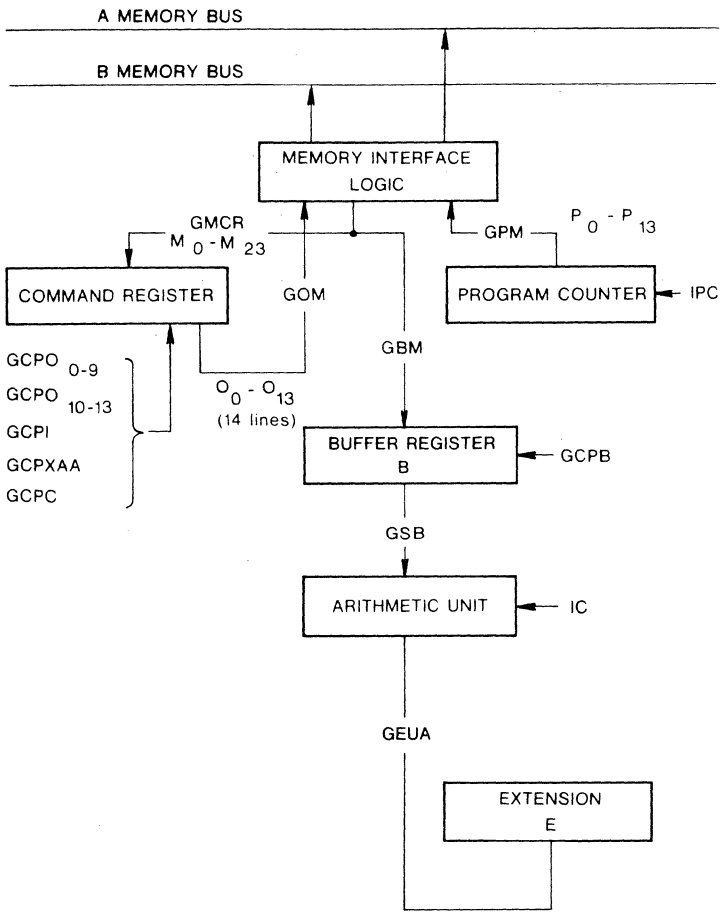


Figure 4-31. STE Instruction Flow Diagram

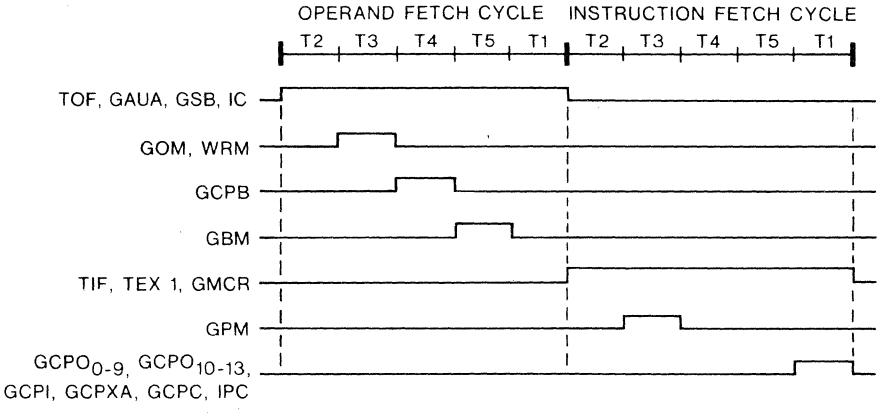


Figure 4-32. STE Instruction Timing Diagram

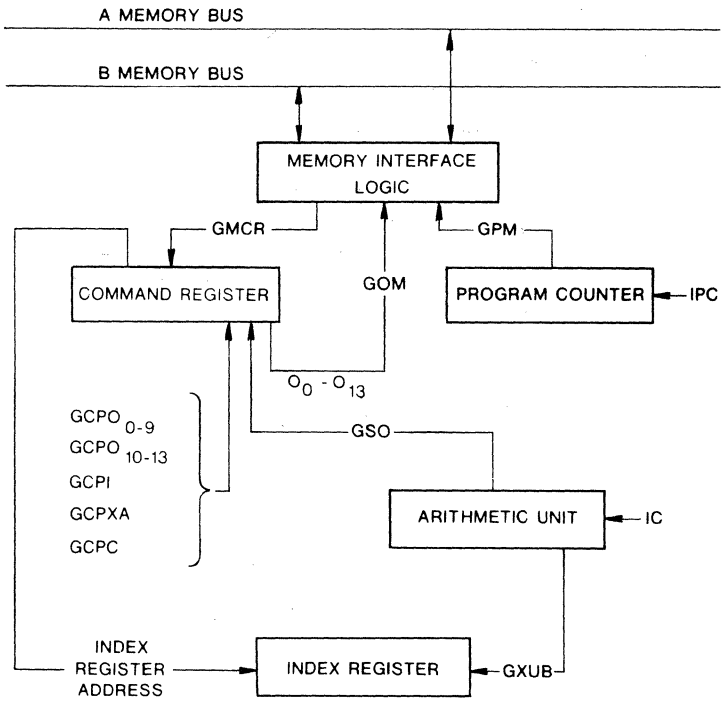


Figure 4-33. STX Instruction Flow Diagram

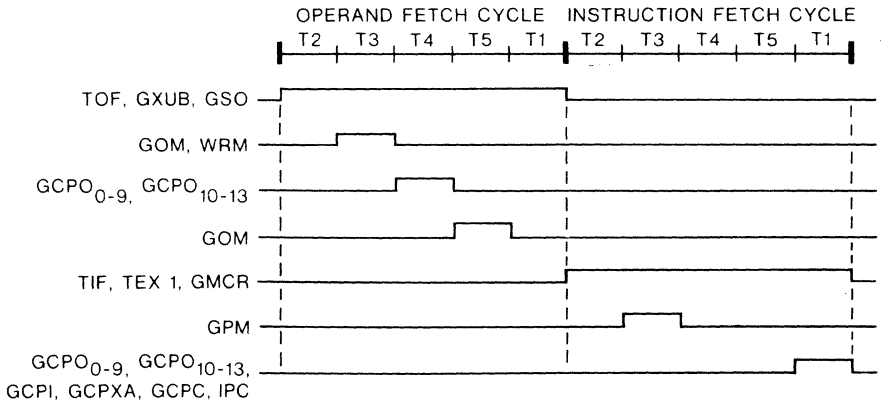


Figure 4-34. STX Instruction Timing Diagram

4.5.15 TCA Instruction

Figures 4-35 and 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.

4.5.16 LDX Instruction

Figures 4-37 and 4-38 are the flow and timing diagrams for the LDX instruction. Table 4-4 gives details of indirect and relocated commands not described here. 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 4-39 and 4-40 are the flow and timing diagrams for the LXA instruction. Table 4-5 gives details not described here. 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 4-41 and 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 4-43 and 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.

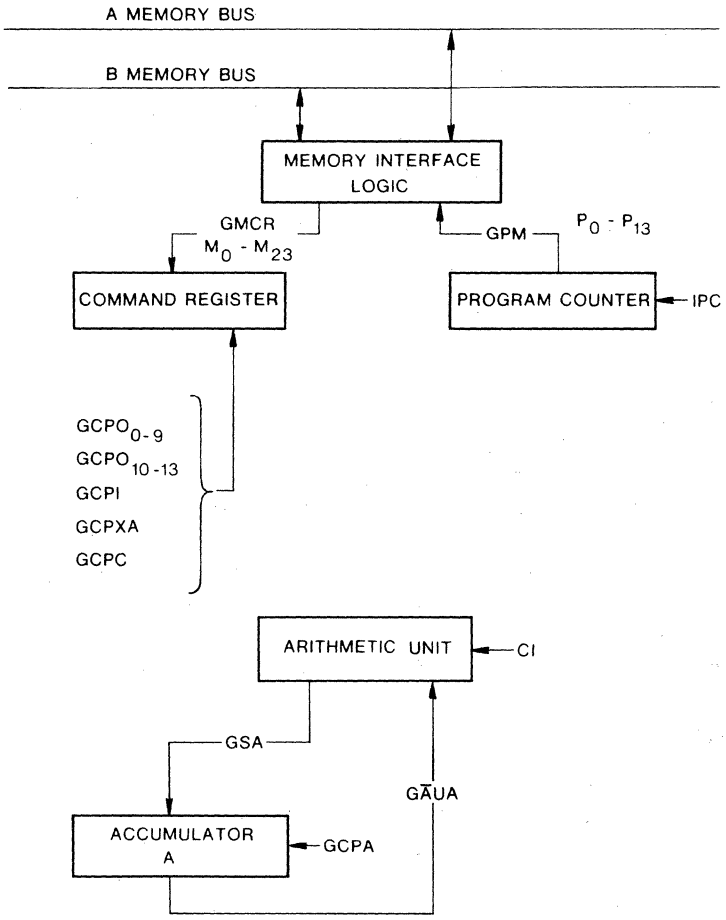


Figure 4-35. TCA Instruction Flow Diagram

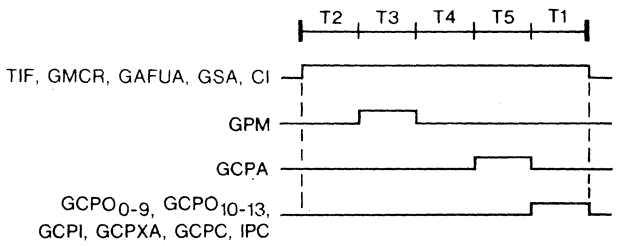


Figure 4-36. TCA Instruction Timing Diagram

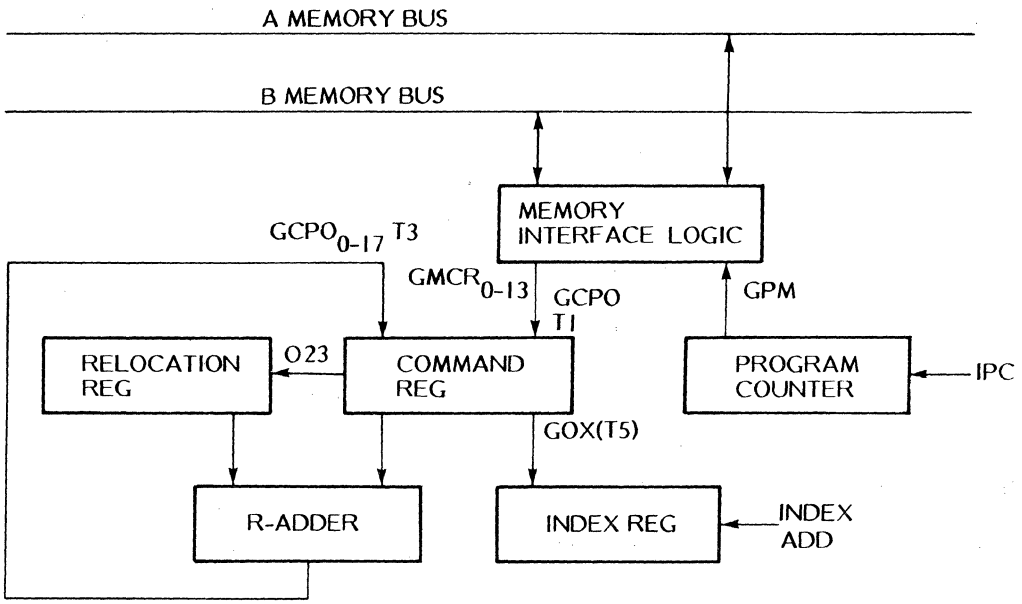


Figure 4-37. LDX Instruction Flow Diagram

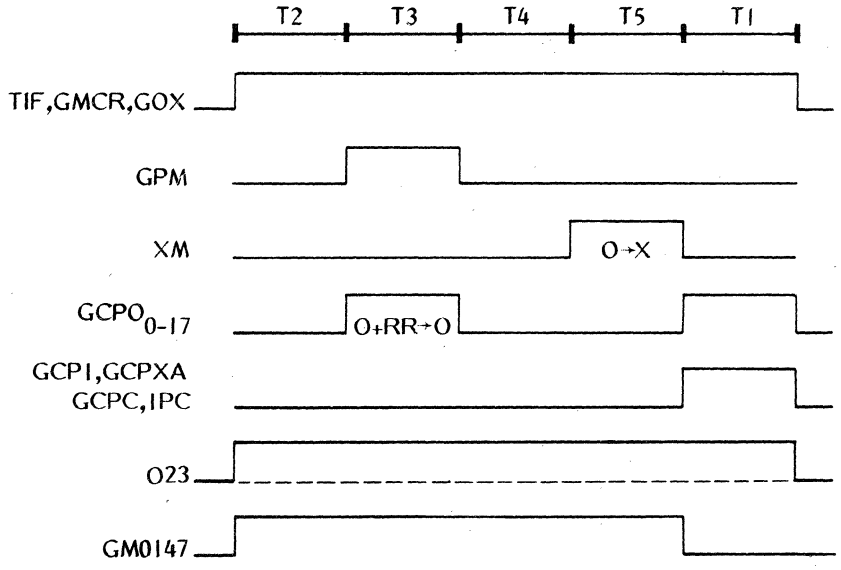


Figure 4-38. LDX Instruction Timing Diagram

	OPCODE	FST-2 MODE	FST-1 MODE
1	$[051(<4)XXXX]_8$ 1 CYCLE IMMEDIATE	RR INHIBITED O0-O12 DATA O13 = 1 SIGNIFIES NEGATION. SEE NOTE 2	RR INHIBITED 14 BIT OPERATION SEE NOTE 1
2	$[051(\geq 4)XXXX]_8$ (M23 = 0) 2 CYCLES INDIRECT	RR INHIBITED 18 BITS OF DATA LOADED INTO X. $M_E \rightarrow X$ 1; XSUB/=1 (INS DEC) NO NEGATIVE INDEXING	RR INHIBITED MULTI INDIRECTION O.K. TIF WORKS SAME AS 1
3	$[051(\geq 4)XXXX]_8$ (M23 = 1) 2 CYCLES INDIRECT	IND. CYCLE: RR INHIBITED SET O23 = 1. TIF CYCLE: O + RR→O O → X XSUB/=1 (INS DEC) NO NEGATIVE INDEXING	RR INHIBITED MULTI INDIRECTION O.K. TIF WORKS SAME AS 1
4	$[451(<4)XXXX]_8$ M13=0 1 CYCLE M13=1 RELOCATED	RR SELECTED, SELR/=0 O + RR→O O23 = 0; XSUB/=1; NO NEGATIVE INDEXING NOT APPLICABLE, ILLEGAL	ILLEGAL
5	$[451(\geq 4)XXXX]_8$ (M23 = 0) 2 CYCLES RELOCATED INDIRECT	IND CYCLE: SELR/=0; O + RR→M; $M_E \rightarrow O$. TIF CYCLE: SELR/=1; XSUB/=1 (NO NEGATION) O 18 BITS→X1	ILLEGAL
6	$[451(\geq 4)XXXX]_8$ (M23 = 1) 2 CYCLES RELOCATED INDIRECT	IND CYCLE: SELR/=0; O + RR→M; $M_E \rightarrow O$; O23=1. TIF CYCLE: SELR/=0; XSUB/=1 (NO NEGATION) O + RR→O O 18 BITS→X1	ILLEGAL

TABLE 4-4 LDX INSTRUCTION DETAILS (Continued)

NOTES:

1. IN FST-1 MODE, SINCE O14-O17 REMAIN ZERO ALL THE TIME (GMO14-17/=1), X14-X17 ARE ALWAYS FILLED WITH ZEROS.

2A. FST2, DIRECT AND BIT 13 = 1, CPU PADS 1'S TO X14 - X17. SUBSEQUENT USE OF THE REG WILL CAUSE

NEGATIVE INDEXING (MINUS UP TO 8K)

ACTION INS DEC: XSUB/(DEVICE E4-PIN 6, I/O F5) = 0

2 BIT SLICE: DEVICE E5 PIN 1 = E5-PIN 2 = 1
X14 - X17 ARE WRITTEN WITH 1'S

2B. FST-2, DIRECT AND BIT 13 = 0,

18 BITS ARE LOADED INTO INDEX

X14-X17 BECOME 0'S BECAUSE
O14-O17 ARE 0'S

TABLE 4-5 LXA INSTRUCTION DETAILS

	Opcode	FST-2 Mode	FST-1 Mode
1	$(0710\ 0000)_8$ (A23=0) 1 CYCLE	RR INHIBITED. 0→B. INDEX REG IS LOADED WITH 18 BITS FROM A THRU O. SEE NOTE 2.	RR INHIBITED. 0→B INDEX REG IS LOADED WITH 18 BITS. X14-X17 PADDED WITH 0'S.
2	$(0710\ 0000)_8$ (A23 = 1) 1 CYCLE	RR IS SELECTED. RR→B; B + A→O; O→X O14-O17 GATED WITH 0'S. SEE NOTE 1	SEE NOTE 1 SAME AS ABOVE

NOTES:

1. INS DEC

SELLOW = 0;

GUA03/=GUA13/=GUA23/=1 TO SELECT

I/O AT MUXA, FORCING 0'S INTO O14-O17.

2. MUXA, I7 IS SELECTED FOR BITS 0-17.

DEVICE A8-PIN 12=1, GUA01/=GUA11/=GUA21/=0.

SELLOW = 1 (F8-1)

GUA03/=GUA13/=GUA23/=0

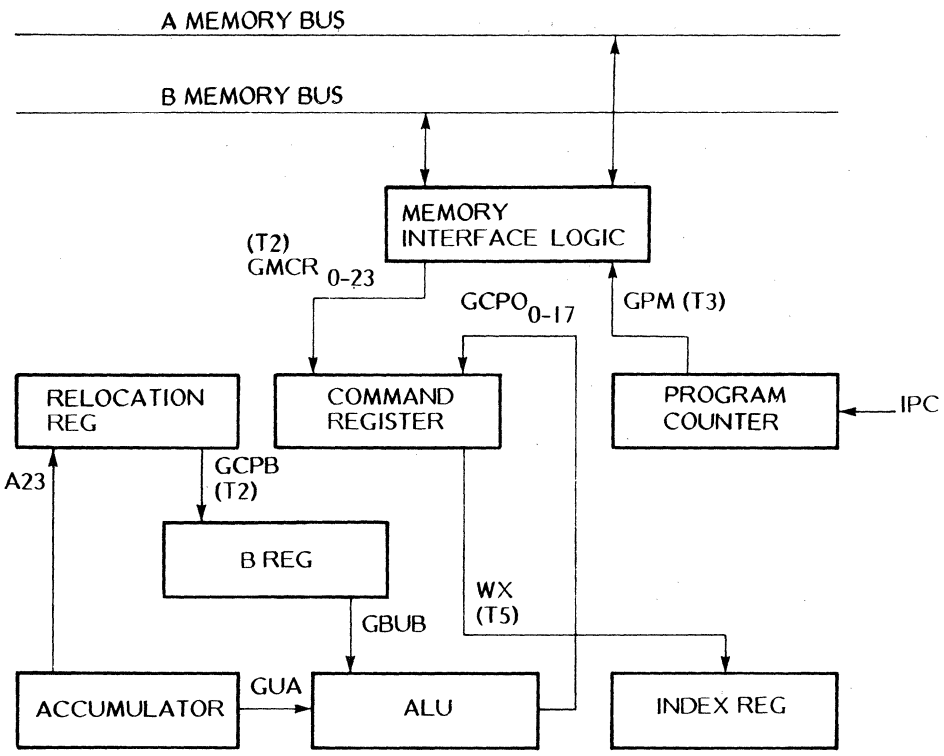


Figure 4-39. LXA Instruction Flow Diagram

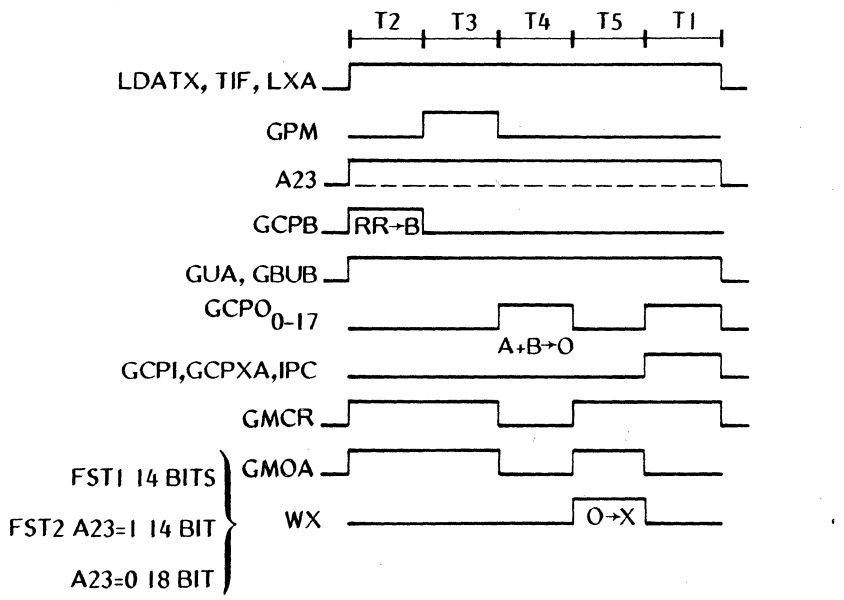


Figure 4-40. LXA Instruction Timing Diagram

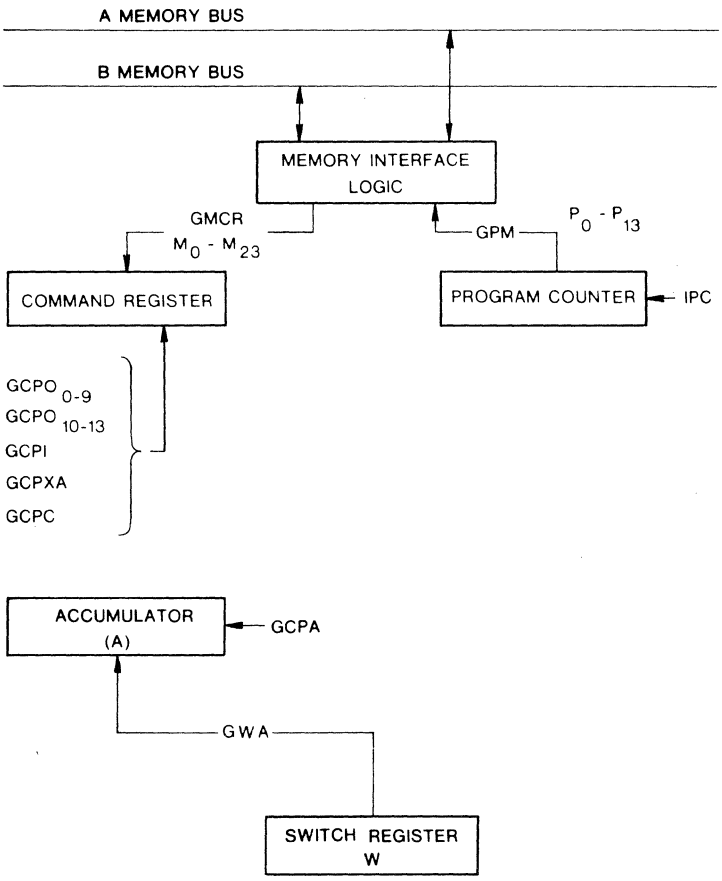


Figure 4-41. RSR Instruction Flow Diagram

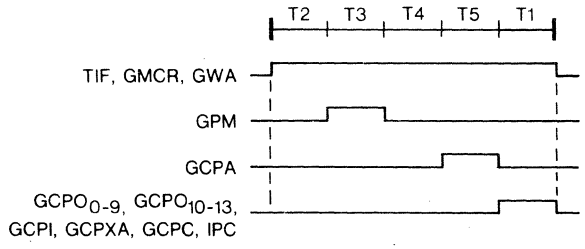


Figure 4-42. RSR Instruction Timing Diagram

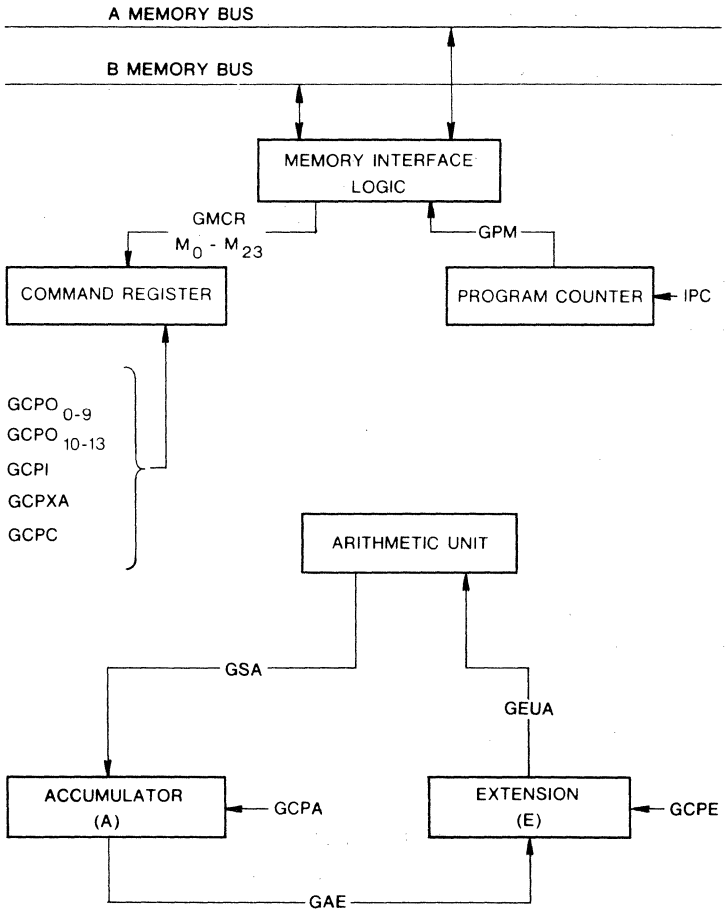


Figure 4-43. EXC Instruction Flow Diagram

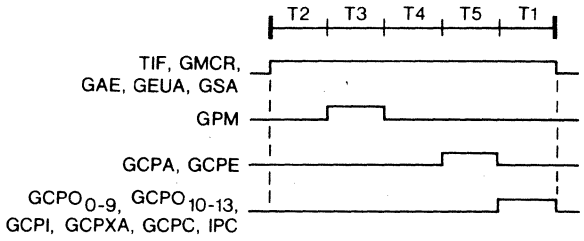


Figure 4-44. EXC Instruction Timing Diagram

4.5.20 RST Instruction

Figures 4-45 and 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), OV (OVerflow), and PD (Parity Disable) flip-flops. The RST instruction requires one memory cycle for execution. At T5 of RST, the outputs of the operand address register, bits O0 through O9 and O14, are gated to the switch register flip-flops, SW0 through SW7, and the Interrupt Enable, Overflow, and Parity Disable flip-flops, as illustrated in Figure 4-45. If O0 is a 1 during RST, SW0 is reset with the clock at T5. If O1 is a 1, SW1 is reset with the clock at T5, etc. The next instruction fetch occurs concurrently.

4.5.21 SST Instruction

Figures 4-47 and 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), OV (OVerflow), and PD (Parity Disable) flip-flops. At time T5, O0 is gated to the set input of SW0, O1 is gated to the set input of SW1 etc., with the exception of the IE flip-flop. O8 is gated to IE at T2 time instead of T5 as illustrated in Figure 4-47. If O0 is a 1 at T5 of SST, SW0 is set with the clock at T5. If O1 is a 1 at T5 of SST, SW1 is set with the clock at T5. If O2 is a 1, SW2 is set with the clock at T5 and so on up to IE. IE is set by the clock at T2 if O8 is a 1. PD is set by the clock at T5 if O14 is a 1.

4.5.22 SPU Instruction

Figures 4-49 and 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.

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

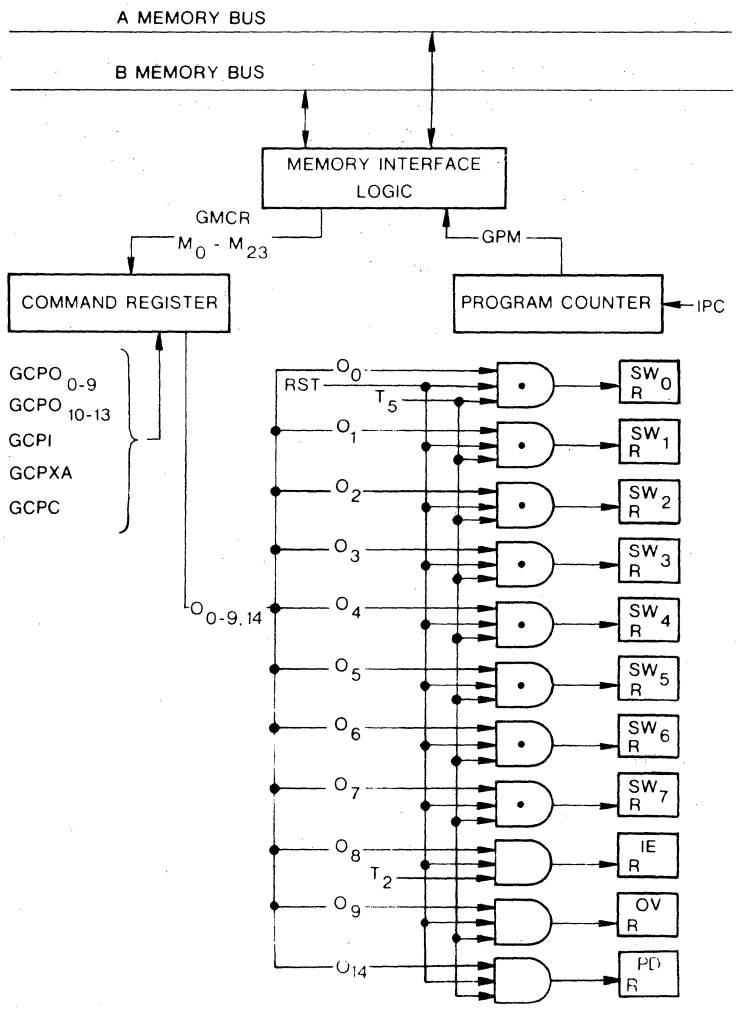


Figure 4-45. RST Instruction Flow Diagram

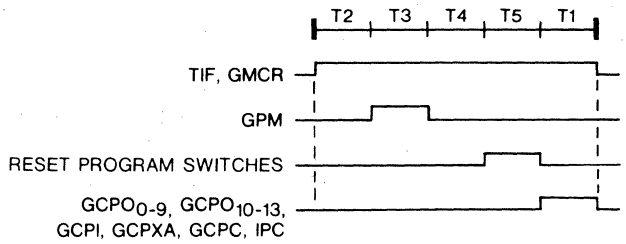


Figure 4-46. RST Instruction Timing Diagram

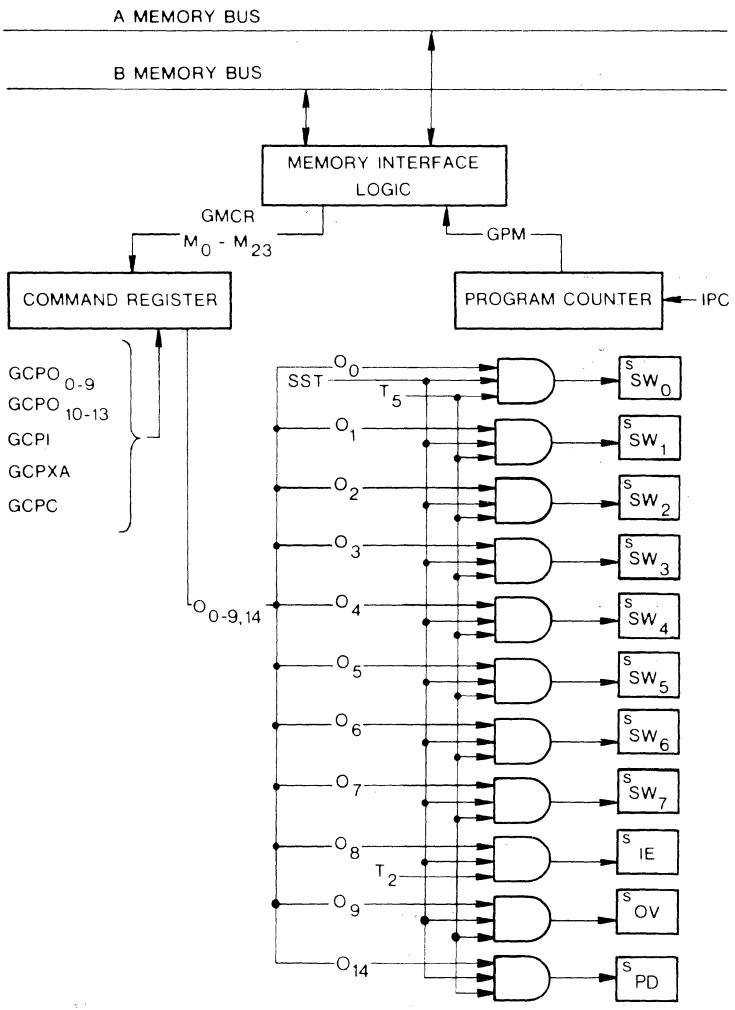


Figure 4-47. SST Instruction Flow Diagram

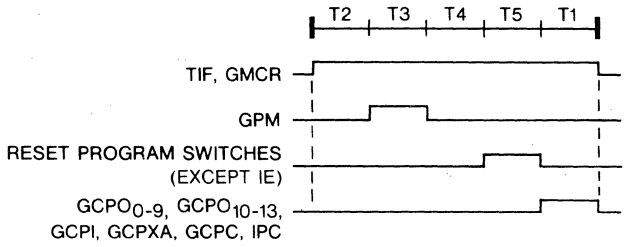


Figure 4-48. SST Instruction Timing Diagram

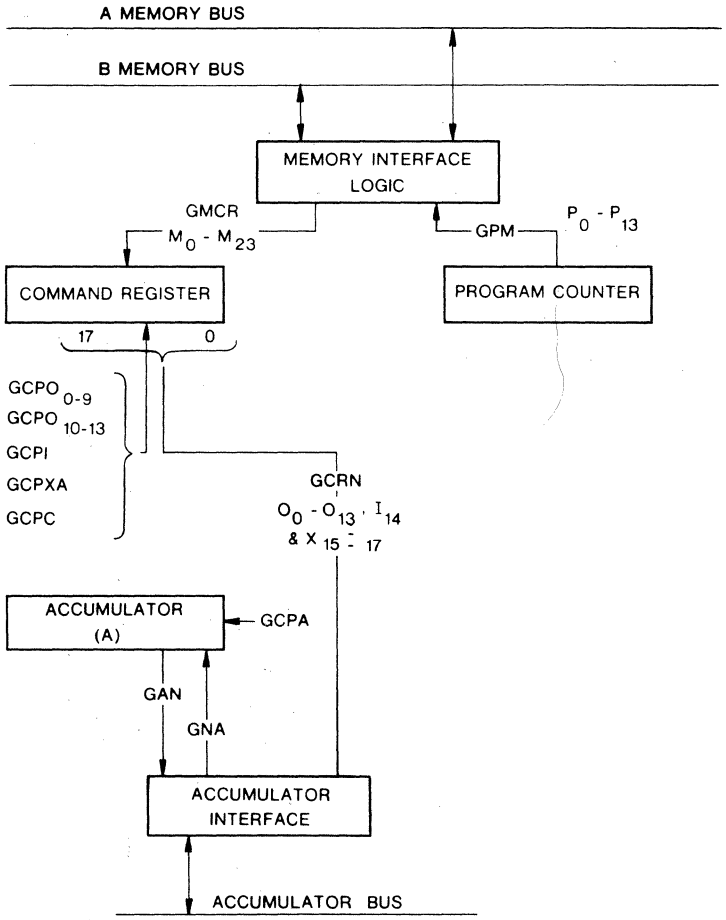


Figure 4-49. SPU Instruction Flow Diagram

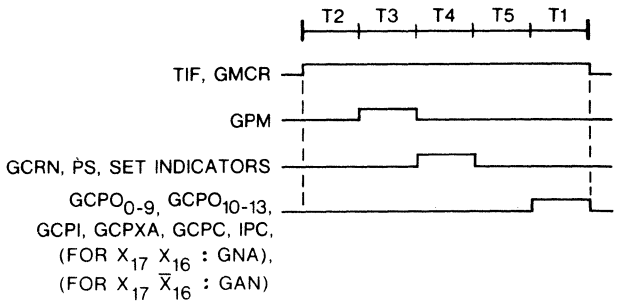


Figure 4-50. SPU Instruction Timing Diagram

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. Details of SPU commands are given in Section 6 and Appendix D.

4.5.23 BRU Instruction

Figures 4-51 and 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 4-53 and 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.

4.5.25 BAT Instruction

Figures 4-55 and 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.

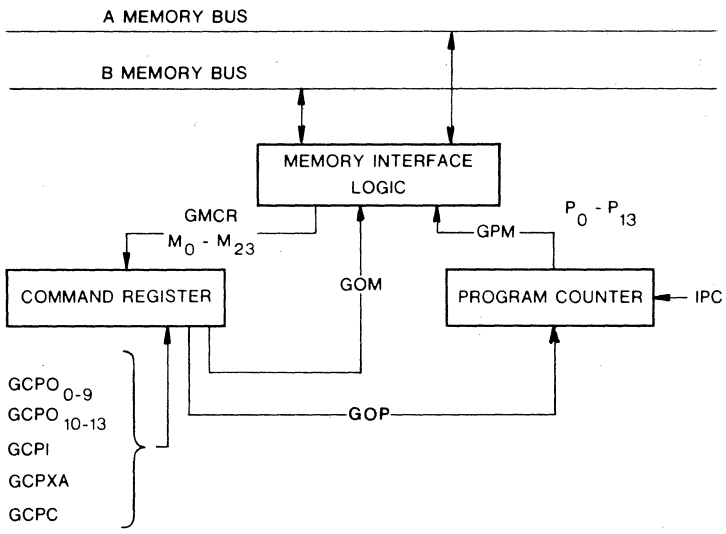


Figure 4-51. BRU Instruction Flow Diagram

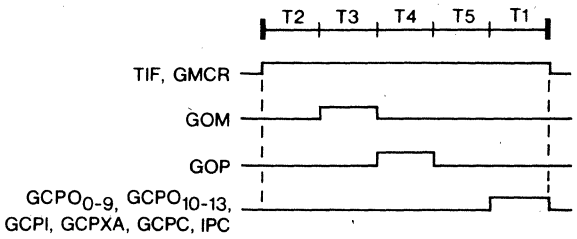


Figure 4-52. BRU Instruction Timing Diagram

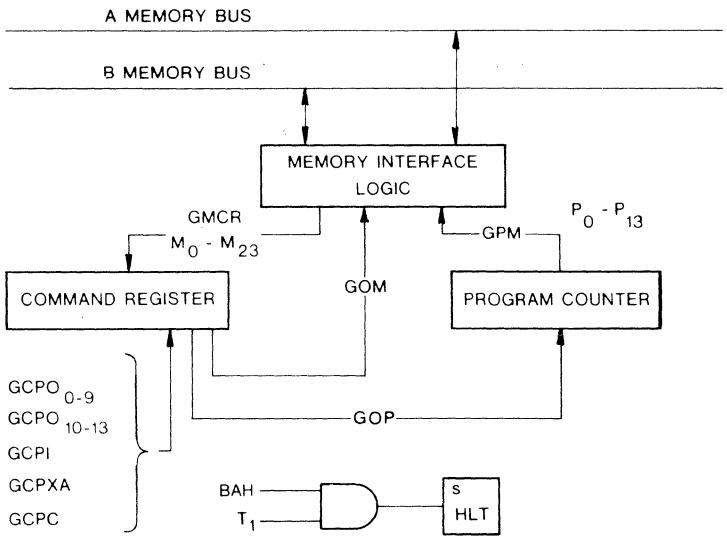


Figure 4-53. BAH Instruction Flow Diagram

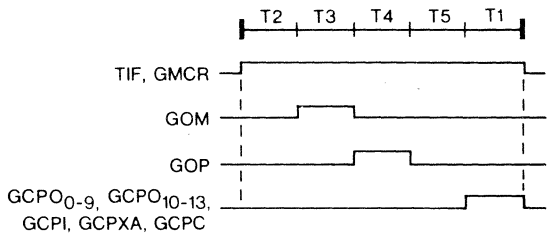


Figure 4-54. BAH Instruction Timing Diagram

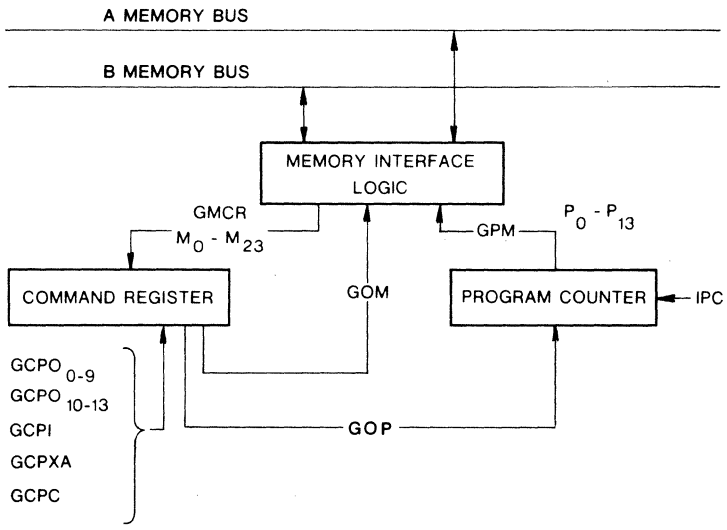


Figure 4-55. BAT Instruction Flow Diagram

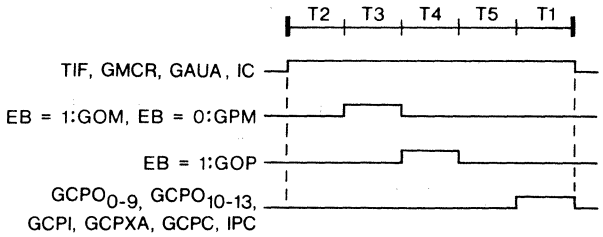


Figure 4-56. BAT Instruction Timing Diagram

4.5.26 BOS Instruction

Figures 4-57 and 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 14-17 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.

4.5.27 BOI Instruction

Figures 4-59 and 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 4-61 and 4-62 are the flow and timing diagrams for the shift instruction. The FST-2 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-2 programmer. The timing for all of these instruction 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 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.

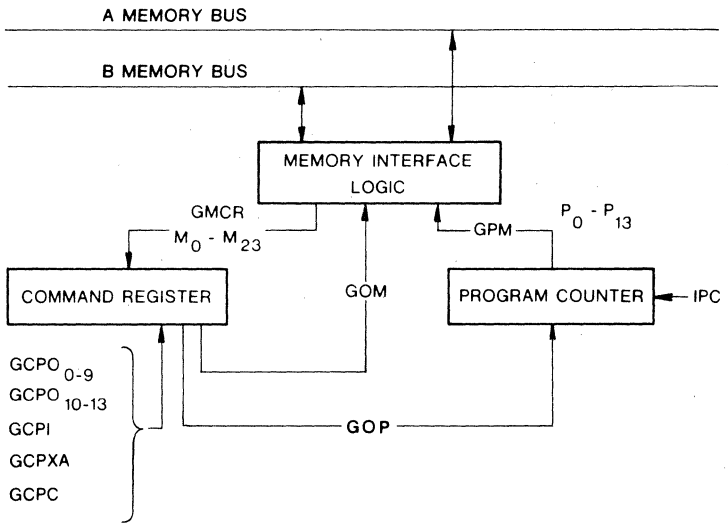


Figure 4-57. BOS Instruction Flow Diagram

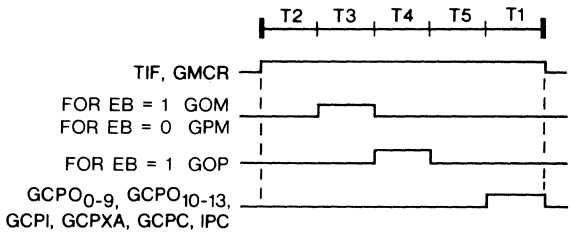


Figure 4-58. BOS Instruction Timing Diagram

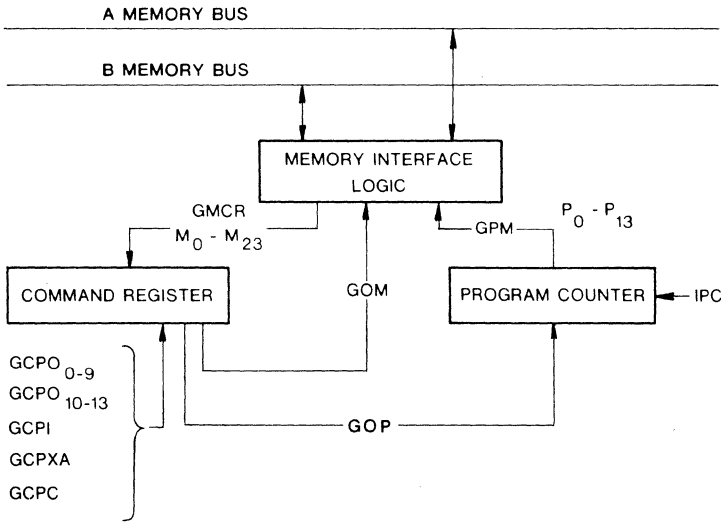


Figure 4-59. BOI Instruction Flow Diagram

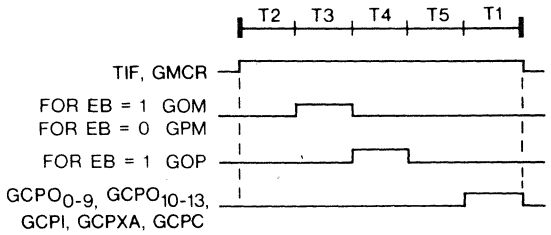


Figure 4-60. BOI Instruction Timing Diagram

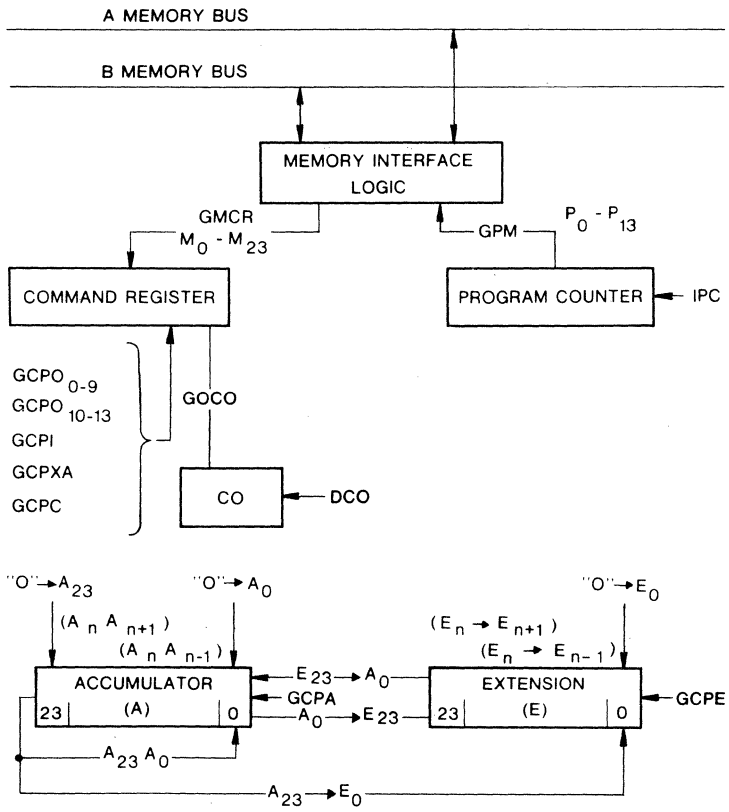


Figure 4-61. SHIFT Instruction Flow Diagram

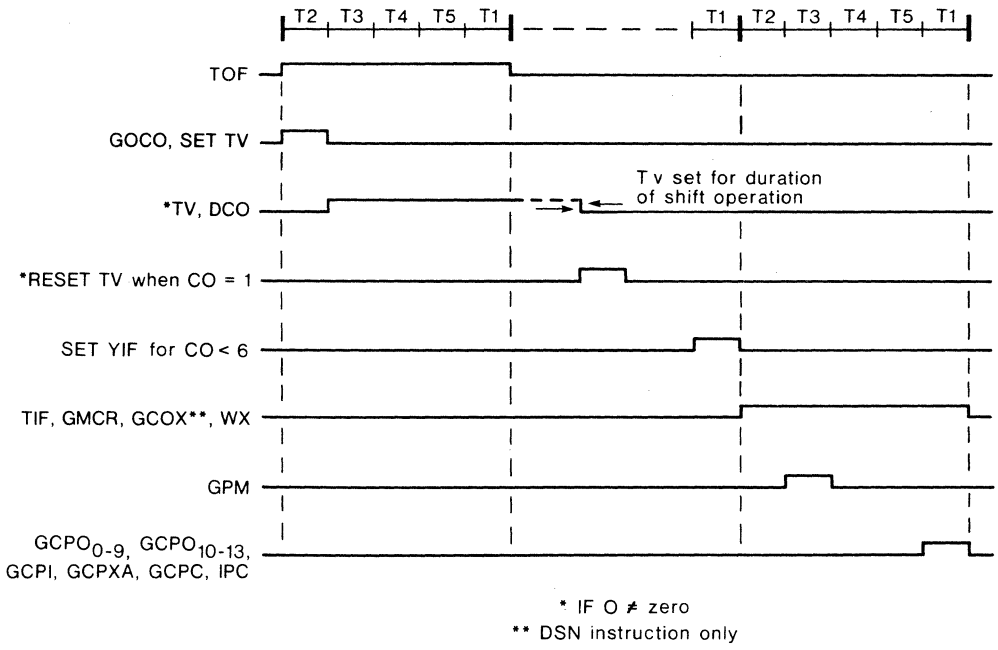
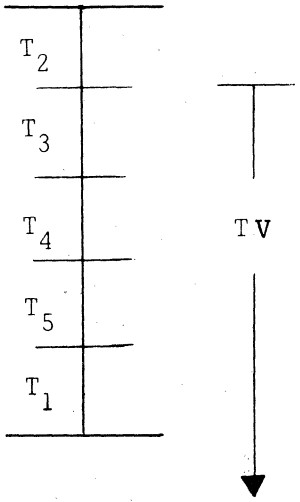


Figure 4-62. SHIFT Instruction Timing Diagram



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 GCPE	GCPA GCPE	$A_{23} \rightarrow E_0$	"0" $\rightarrow E_0$	"0" $\rightarrow E_0$
						GCPA GCPE	GCPA GCPE	GCPA GCPE

Figure 4-63. SHIFT Instruction Gating Table

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 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 differ. 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 4-64 and 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 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 4-66 and 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.

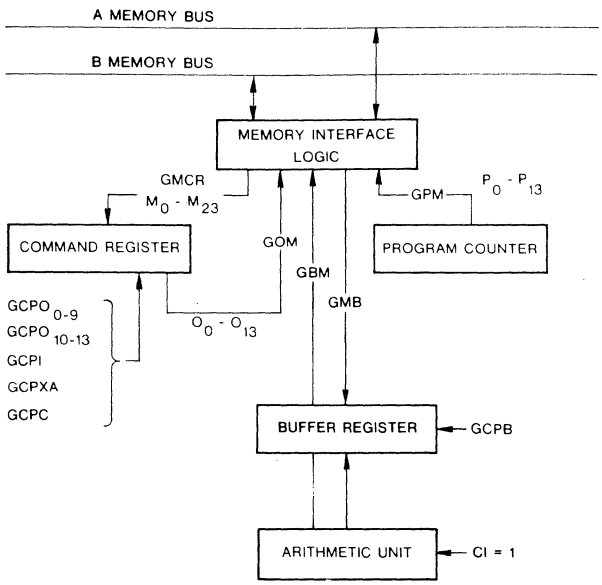


Figure 4-64. AOM Instruction Flow Diagram

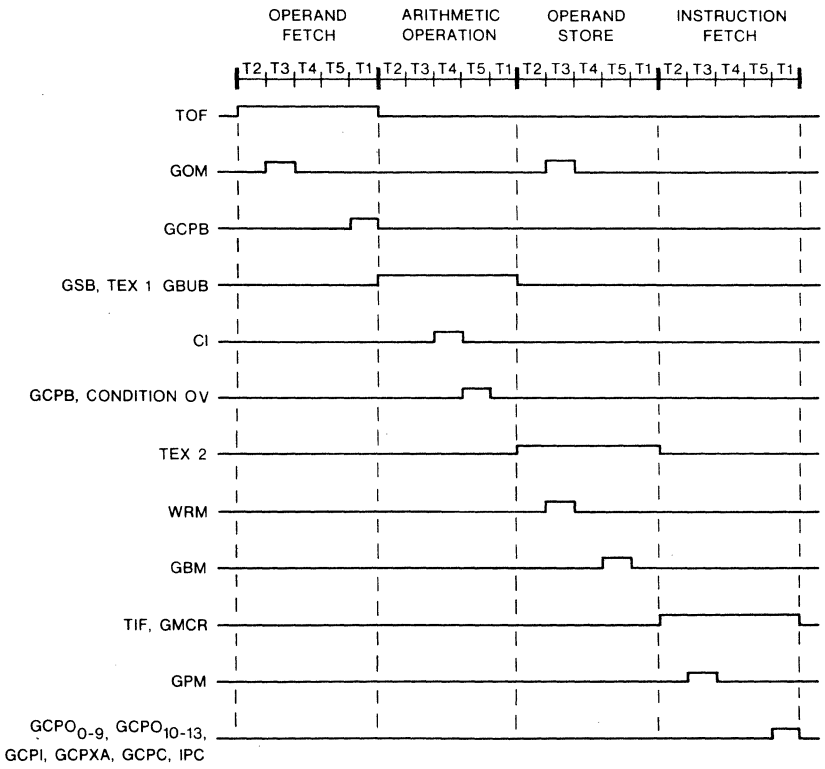


Figure 4-65. AOM Instruction Timing Diagram

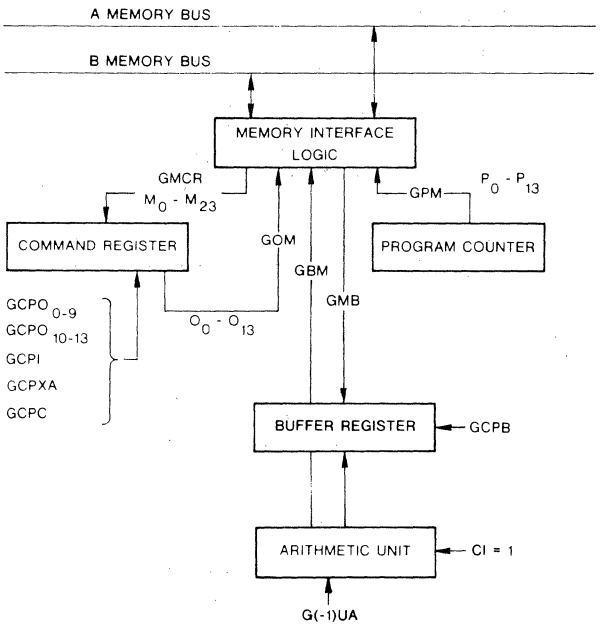


Figure 4-66. SOM Instruction Timing Diagram

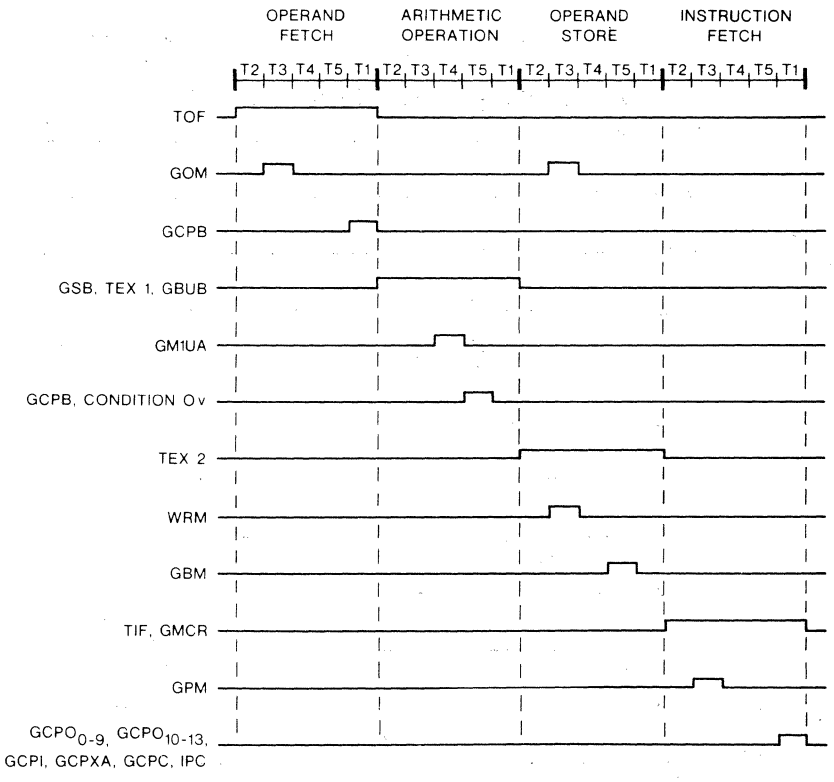


Figure 4-67. SOM Instruction Timing Diagram

4.5.31 DADD Instruction

Figures 4-68 and 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 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 4-70 and 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 with consideration given to 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.

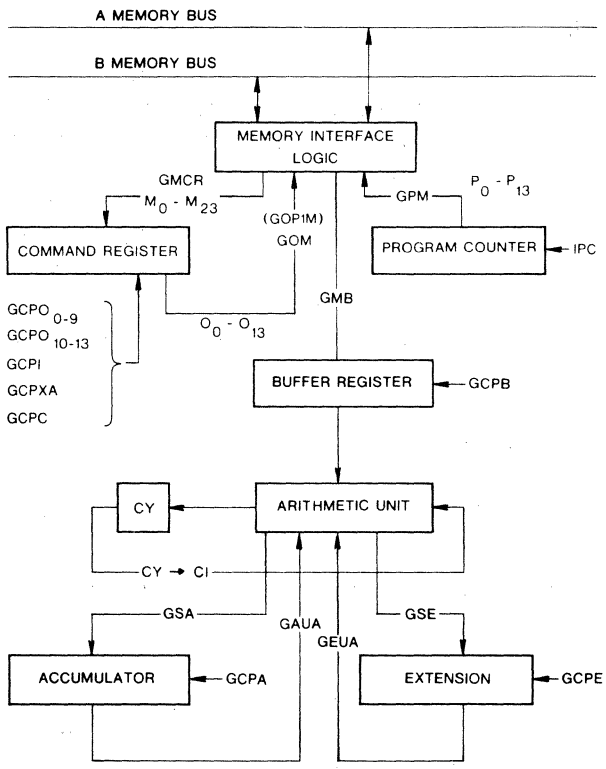


Figure 4-68. DADD Instruction Flow Diagram

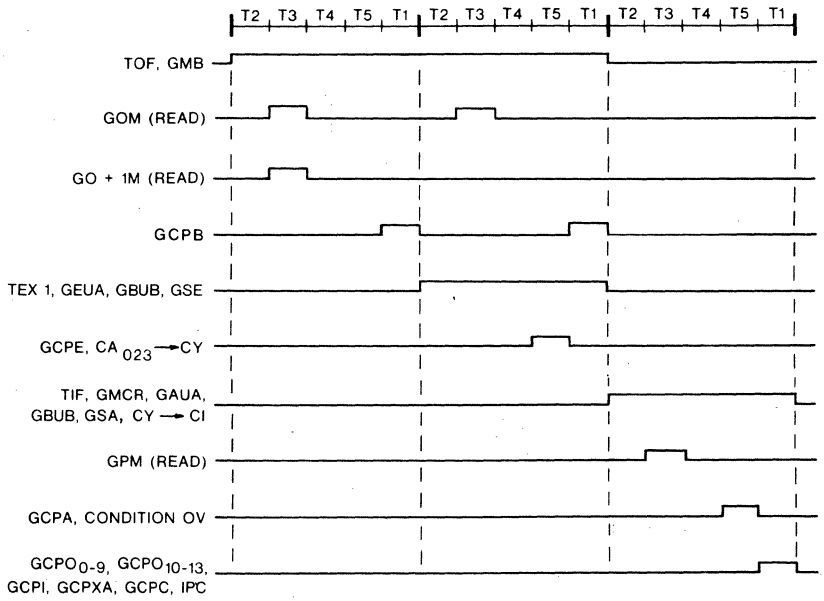


Figure 4-69. DADD Instruction Timing Diagram

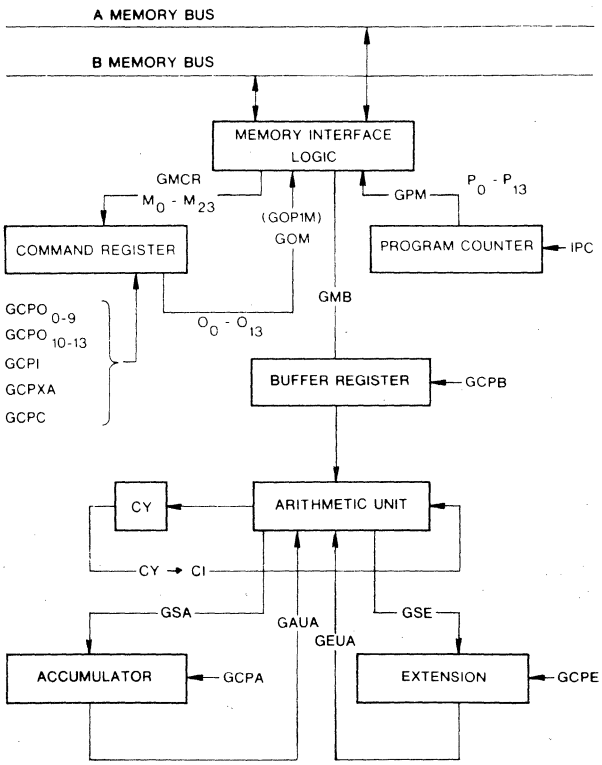


Figure 4-70. DSUB Instruction Flow Diagram

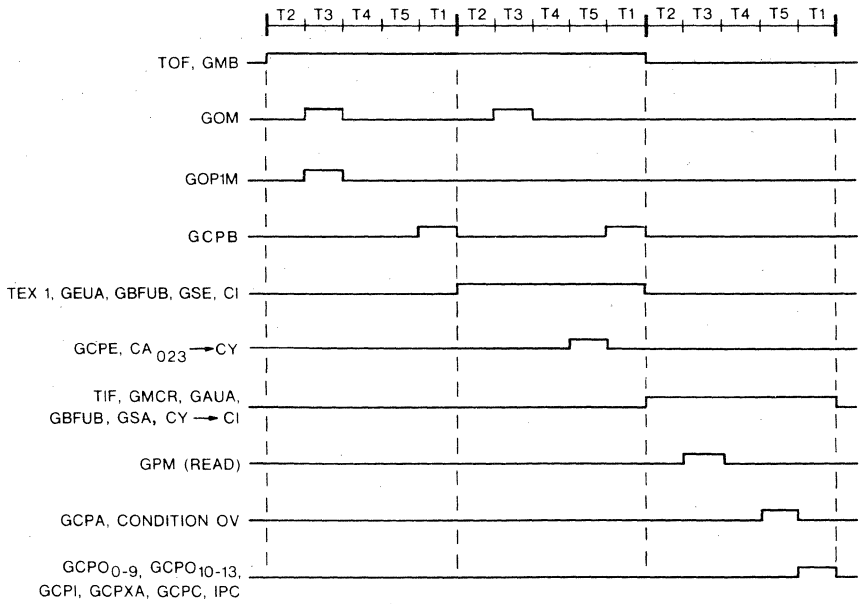


Figure 4-71. DSUB Instruction Timing Diagram

4.5.33 DLD Instruction

Figures 4-72 and 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.

4.5.34 DST Instruction

Figures 4-74 and 4-75 are the flow and timing diagrams for the DST instructions. 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 Plus 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 4-76 and 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. Zeroes 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 register (E0) is a "1," GCPA gates the new sum into A. Otherwise, the contents of A and E 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)

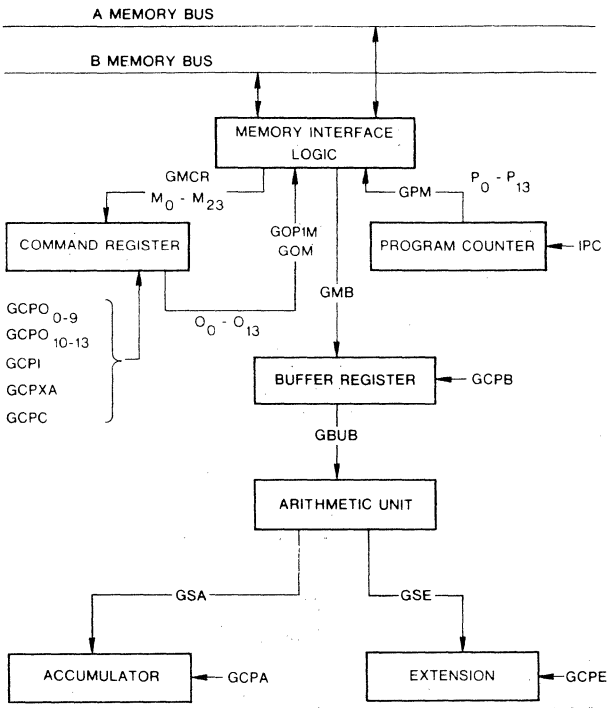


Figure 4-72. DLD Instruction Flow Diagram

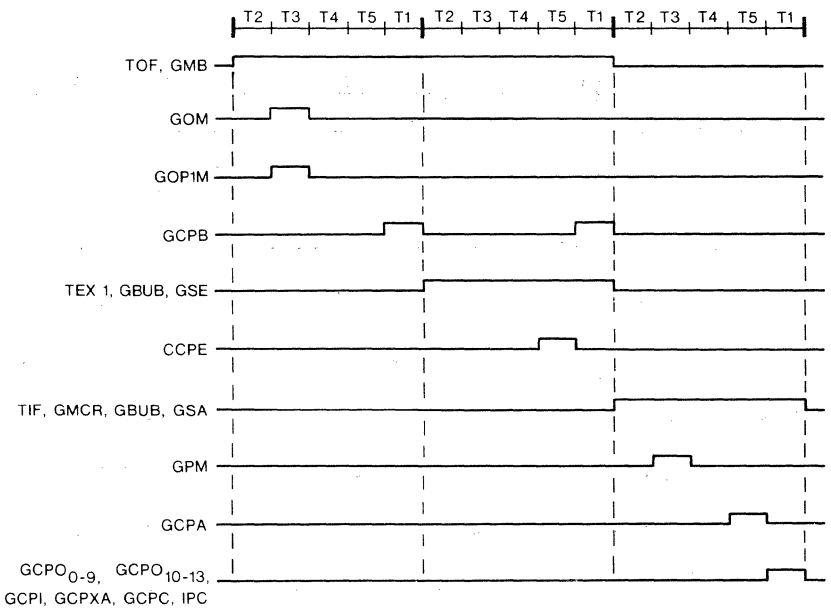


Figure 4-73. DLD Instruction Timing Diagram

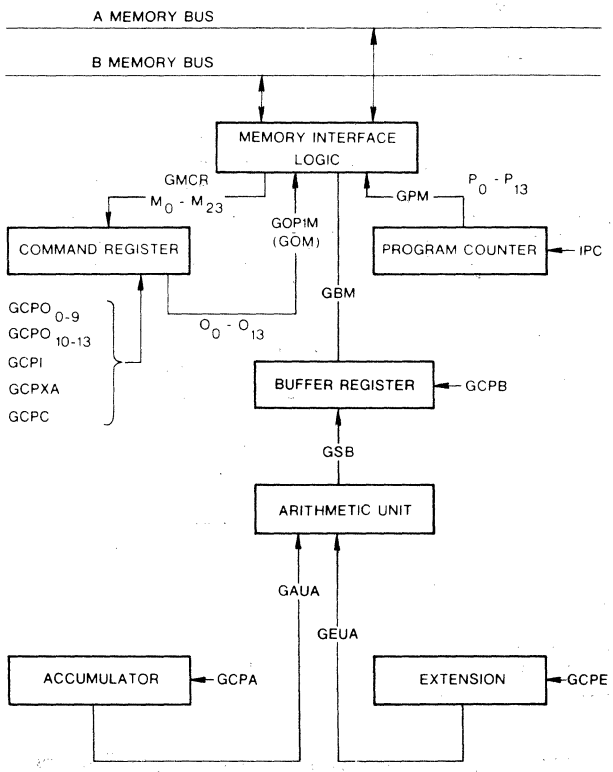


Figure 4-74. DST Instruction Flow Diagram

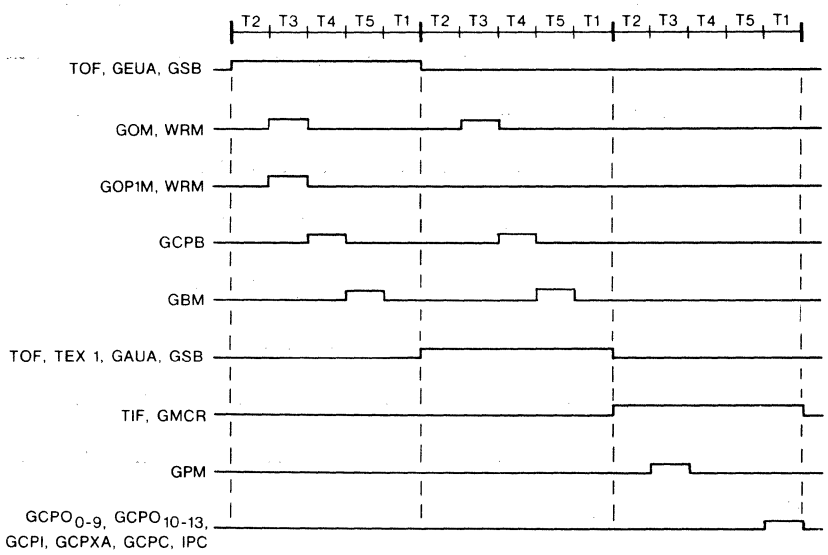


Figure 4-75. DST Instruction Timing Diagram

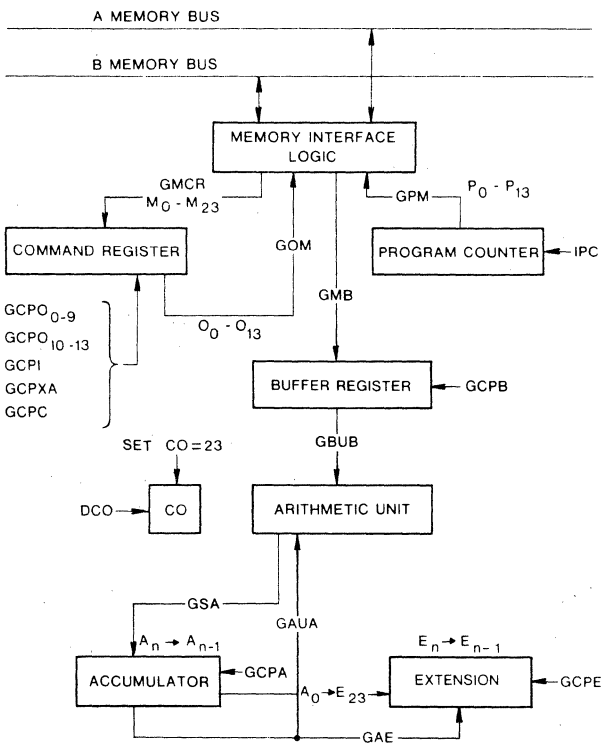


Figure 4-76. MUL Instruction Flow Diagram

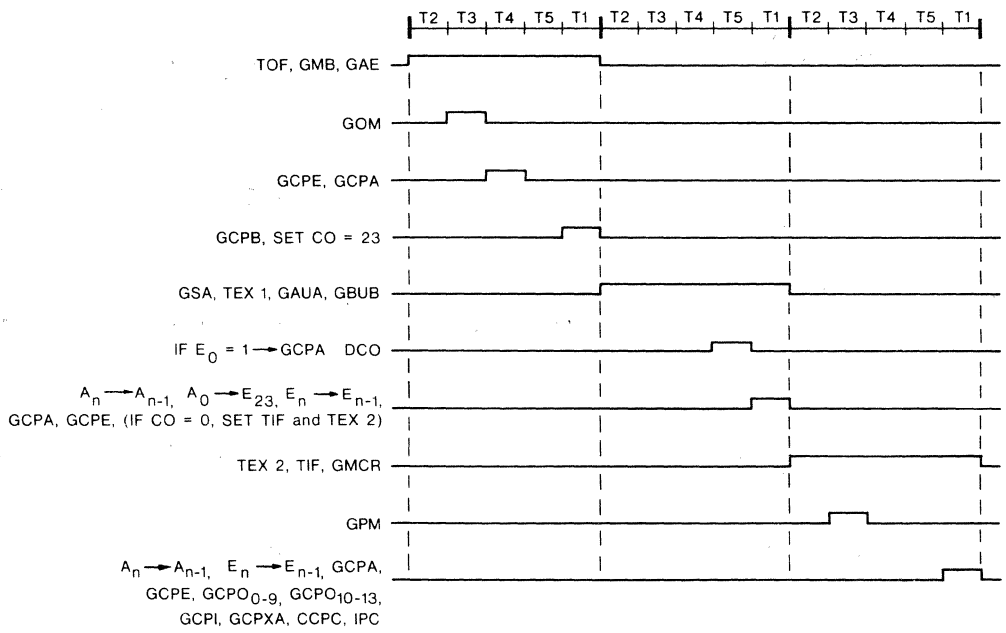


Figure 4-77. MUL Instruction Timing Diagram

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 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.

	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 4-78. Simplified Multiplication Example

4.5.36 DIV Instruction

Figures 4-79 and 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.

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 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 EO 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.

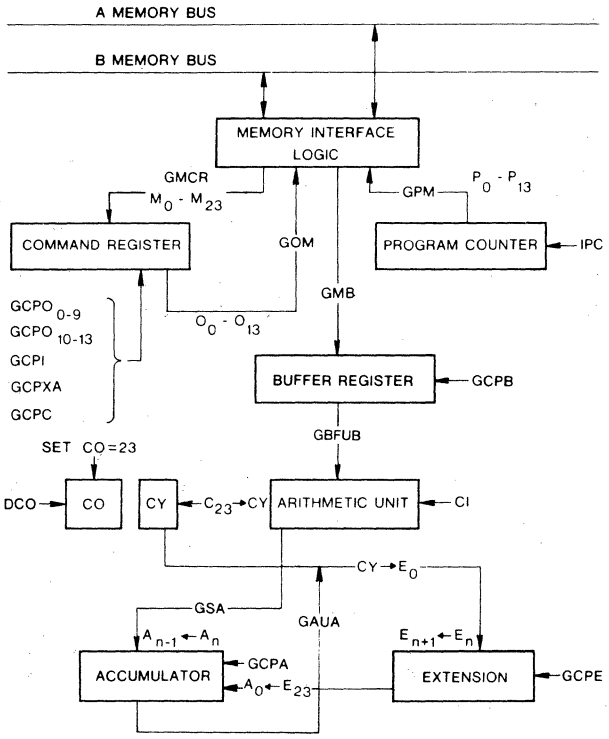


Figure 4-79. DIV Instruction Flow Diagram

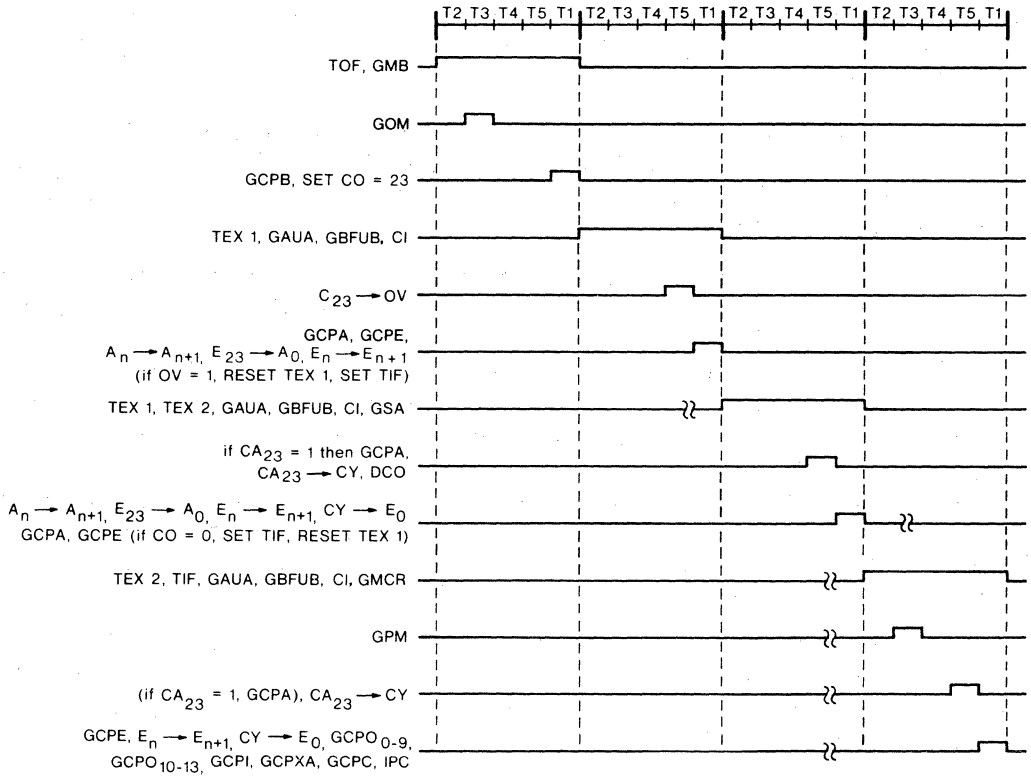


Figure 4-80. DIV Instruction Timing Diagram

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 Significant 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:

$A,E/(Me) \leq 2^{23} -1$

$A/(Me) < 1$

These bounds are consistent 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 4-81. Simplified Division Example

Figures 4-84 and 4-85 are the flow and timing diagrams for the LAR instruction and CLA sub-instruction. Table 4-7 gives details not described here. The contents of the Relocation Register (RR) are not changed. The data path from (RR) is through the Buffer (B) Register to the Arithmetic unit. From B, the data is gated to the Arithmetic unit via signal GBUB (Gate B Register to Arithmetic Unit B input). From ALU the data is transferred to the Accumulator via signal GSA (Gate-Sum to Accumulator). GCPA is energized at T5 clocking the results into A. The CLA instruction is identical except that operand Bit 2 causes GBUB to be blocked. As a result, a zero is gated to the B input of the ALU. GSA gates the sum of zero to the accumulator.

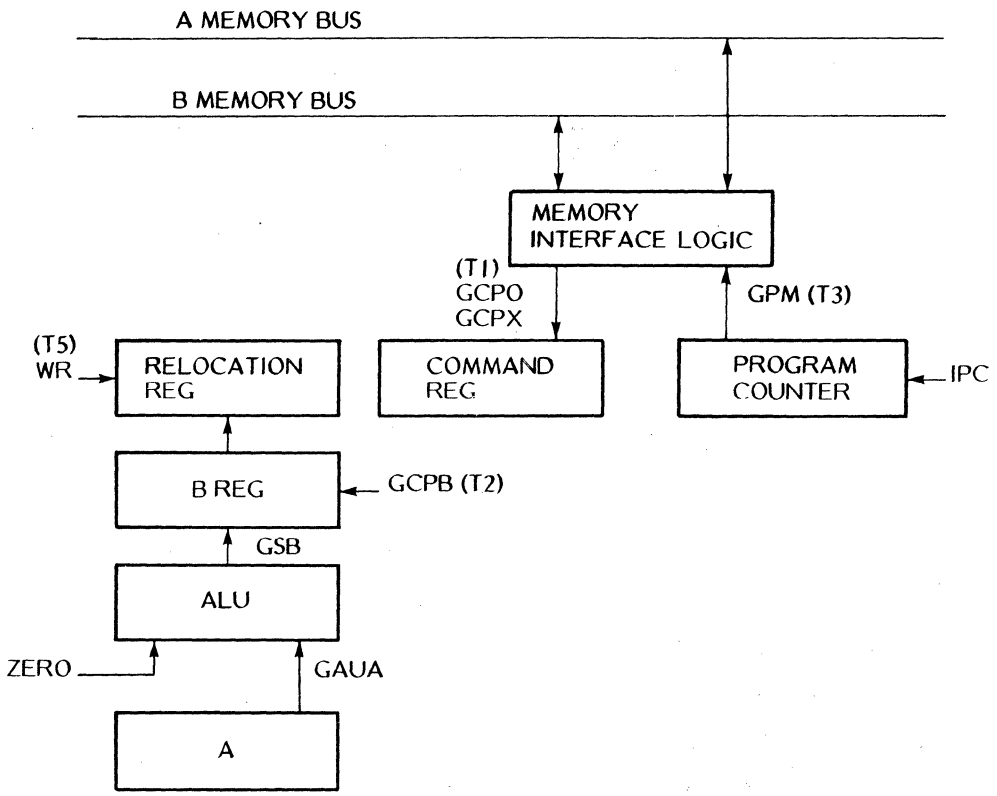


Figure 4-82. LRA Instruction Flow Diagram

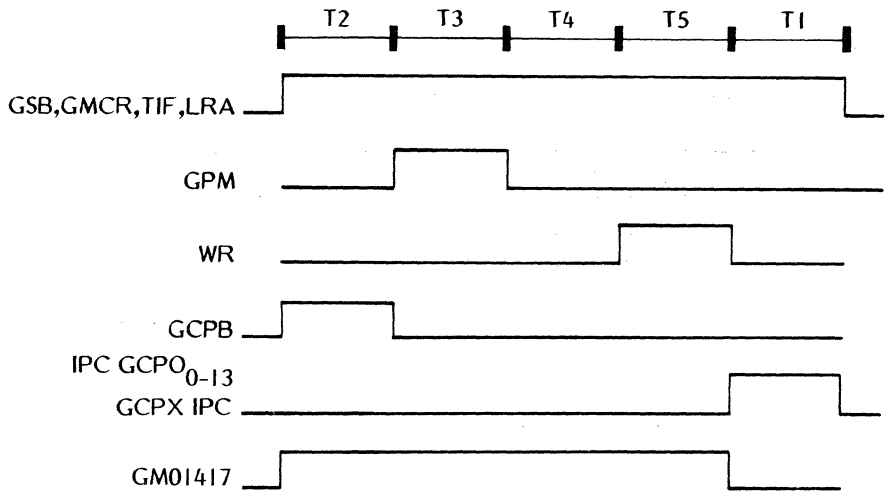


Figure 4-83. LRA Instruction Timing Diagram

TABLE 4-6 LRA INSTRUCTION DETAILS

	OPCODE	FST-2 MODE	FST-1 MODE
1	$(07000400)_8$ 1 CYCLE	SEE NOTE 1 18 BITS OPERATION	EVEN THOUGH INHIBITED BY ASSEMBLER, H/W WORKS THE SAME AS FST-2

NOTES:

1. AT ALU, MUXA I7 IS SELECTED, THUS AT INS DEC:

   DEVICE A8 Pin 12 = 0; GUA01/=GUA11/=GUA21=0;
   SELLOW = 1
   AND GUA03/=GUA13/=GUA23/=0.

2. AT ALU, MUXB I3 IS SELECTED TO FORCE ALL ZEROES.
   INS DEC: GBUB11/=GBUB01/=1

4.5.38 LAR Instruction and CLA Sub-Instruction

Figures 4-84 and 4-85 are the flow and timing diagrams for the LAR instruction and CLA sub-instruction. Table 4-7 gives details not described here. The contents of the Relocation Register (RR) are not changed. The data path from (RR) is through the Buffer (B) Register to the Arithmetic unit. From B, the data is gated to the Arithmetic unit via signal GBUB (Gate B Register to Arithmetic Unit B input). From ALU the data is transferred to the Accumulator via signal GSA (Gate-Sum to Accumulator). GCPA is energized at T5 clocking the results into A. The CLA instruction is identical except that operand Bit 2 causes GBUB to be blocked. As a result, a zero is gated to the B input of the ALU. GSA gates the sum of zero to the accumulator.

TABLE 4-7 LAR INSTRUCTION DETAILS

	OPCODE	FST-2 MODE	FST-1 MODE
1	$(07000600)_8$ 1 CYCLE	SEE NOTE1 18 BITS OPERATION	EVEN THOUGH INHIBITED BY ASSEMBLER, H/W WORKS SAME AS FST-2

NOTES:

1. AT ALU, MUXA, I0 is selected.

   AT INS DEC: GUA01/=GUA11/=GUA21/=1
   SELLOW = 1;
   GUA03/=GUA13/=GUA23/=1

2. AT ALU, MXUB, I0 IS SELECTED.

INS DEC: DEVICE A8 PIN 6 = 0

GBUB01/=GBUB11/=11

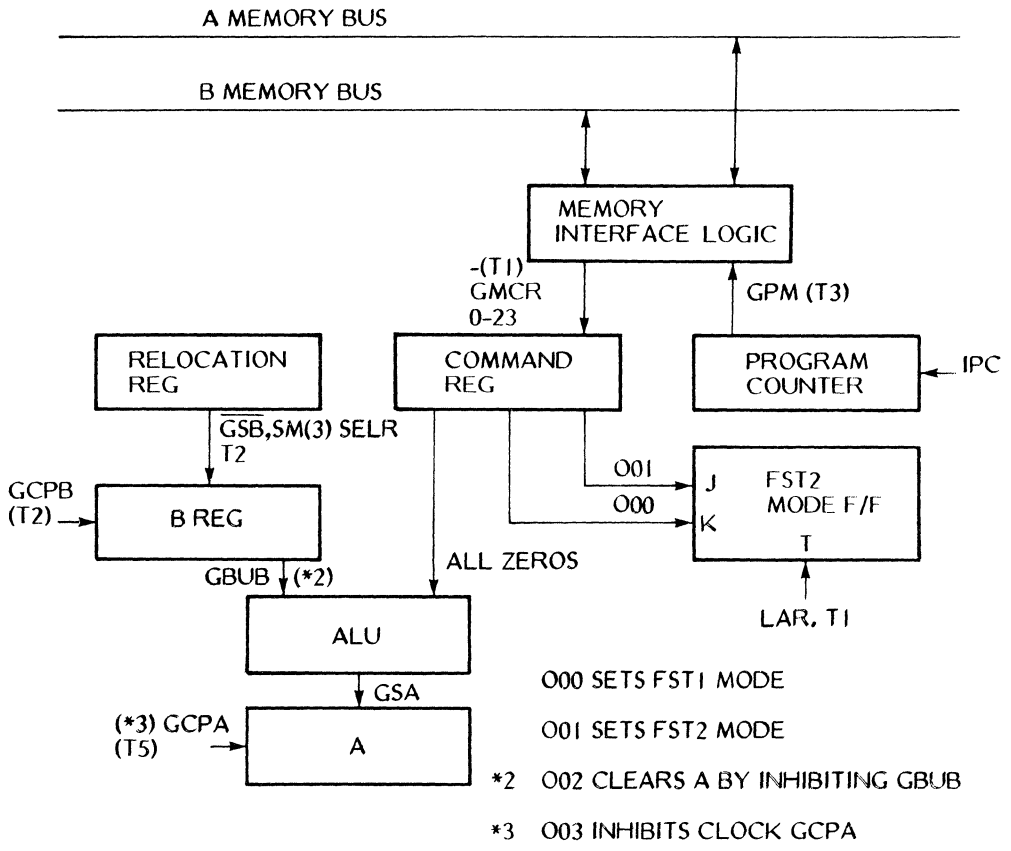


Figure 4-84. LRA Instruction and CLA Sub-instruction Flow Diagram

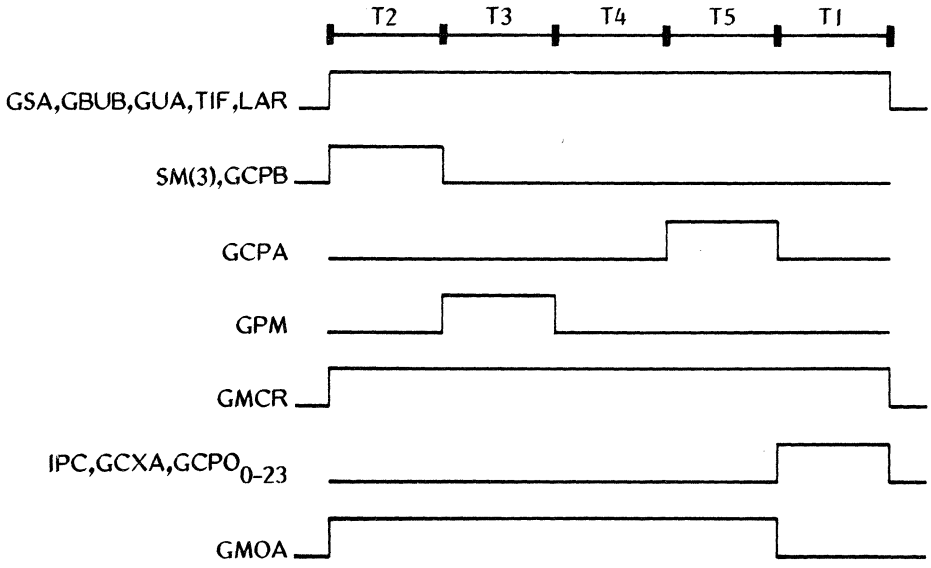


Figure 4-85. LAR Instruction and CLA Sub-instruction Timing Diagram

SECTION 5

MEMORY SYSTEM

5.1 INTRODUCTION

The memory system provided with the FST-2 is a random access storage unit. It uses dynamic MOS semiconductor RAMs as storage elements.

The memory is modular in 16K words by 25 bits. Its minimum size is 16K words. Its maximum size is (192 x 1024) 196K words. The storage elements are mounted on up to 24 identical memory boards. Each memory board contains 8K words by 25 bits, 2 memory boards make up a module.

The memory is divided into an A memory and a B memory. The A memory contains all even words, the B memory all odd words. Each 16K module consists of one A memory board and one B memory board.

The memory system is physically located in a backplane A2 module above the CPU backplane A1. Figure 5-1 shows the backplane card map. Addresses, data-in, and data-out and control signals are transferred by hard-wired A and B buses between memory and CPU backplanes.

Memory addresses, data-in and data-out are time-shared on 24 leads of each bus, originating on the three data bus cards of the CPU. Control signals come from the CPU clock board and the Memory Control B board in the CPU.

Both A and B memories are independent and can be accessed simultaneously. For example, A memory could execute a CPU instruction fetch while B memory executes a DMA transfer to or from a peripheral.

The 25 bits of each memory word represent 24 bits of data and an even parity bit.

5.2 PRINCIPLES OF OPERATION

5.2.1 Circuit Description

Each memory board, 97340206, contains 8K words of 25 bits. They are stored in 50 memory devices, each 4096 x 1 bit large. See Figure 5-2 for a schematic of the memory board. Addresses and data enter the board via the same pins (ADAIxx), output data leave the board on pins ADAOxx which are wired together with the ADAIxx pins.

Addresses

The least significant 12 address bits, ADAI00-11, are clocked into the address register (F2, F8). Bit 12, ADAI12, determines whether the chip containing the lower 4K or the one containing the upper 4K is selected. It is clocked into the chip

	ROW	NAME	PART NO.	NOTES
A	0	AMEM 60 (SPARE MEM)		
	2	AMEM 54 8K x 25	97340206	OPTIONAL
	4	AMEM 50 8K x 25	97340206	OPTIONAL
	6	AMEM 44 8K x 25	97340206	OPTIONAL
	8	AMEM 40 8K x 25	97340206	OPTIONAL
B	0	AMEM 34 8K x 25	97340206	OPTIONAL
	2	AMEM 30 8K x 25	97340206	OPTIONAL
	4	AMEM 24 8K x 25	97340206	OPTIONAL
	6	AMEM 20 8K x 25	97340206	OPTIONAL
	8	AMEM 14 8K x 25	97340206	OPTIONAL
C	0	AMEM 10 8K x 25	97340206	OPTIONAL
	2	AMEM 04 8K x 25	97340206	OPTIONAL
	4	AMEM 00 8K x 25	97340206	
	6	AMEM/BMEM (SPARE MEM.)		
	8	BMEM 00 8K x 25	97340206	
D	0	BMEM 04 8K x 25	97340206	OPTIONAL
	2	BMEM 10 8K x 25	97340206	OPTIONAL
	4	BMEM 14 8K x 25	97340206	OPTIONAL
	6	BMEM 20 8K x 25	97340206	OPTIONAL
	8	BMEM 24 8K x 25	97340206	OPTIONAL
E	0	BMEM 30 8K x 25	97340206	OPTIONAL
	2	BMEM 34 8K x 25	97340206	OPTIONAL
	4	BMEM 40 8K x 25	97340206	OPTIONAL
	6	BMEM 44 8K x 25	97340206	OPTIONAL
	8	BMEM 50 8K x 25	97340206	OPTIONAL
F	0	BMEM 54 8K x 25	97340206	OPTIONAL
	2	BMEM 60 (SPARE MEM.)		
	4	SPARE		
	6	SPARE		
	8	FST2-488 BUS CTRLER	97430007	OPTIONAL
G	0	FST2-488 BUS BUFFER	97430006	OPTIONAL

NOTE: Board Number = Starting Address x 10,000 Octal.

Figure 5-1. Module Card Map FST-2 Memory, A2

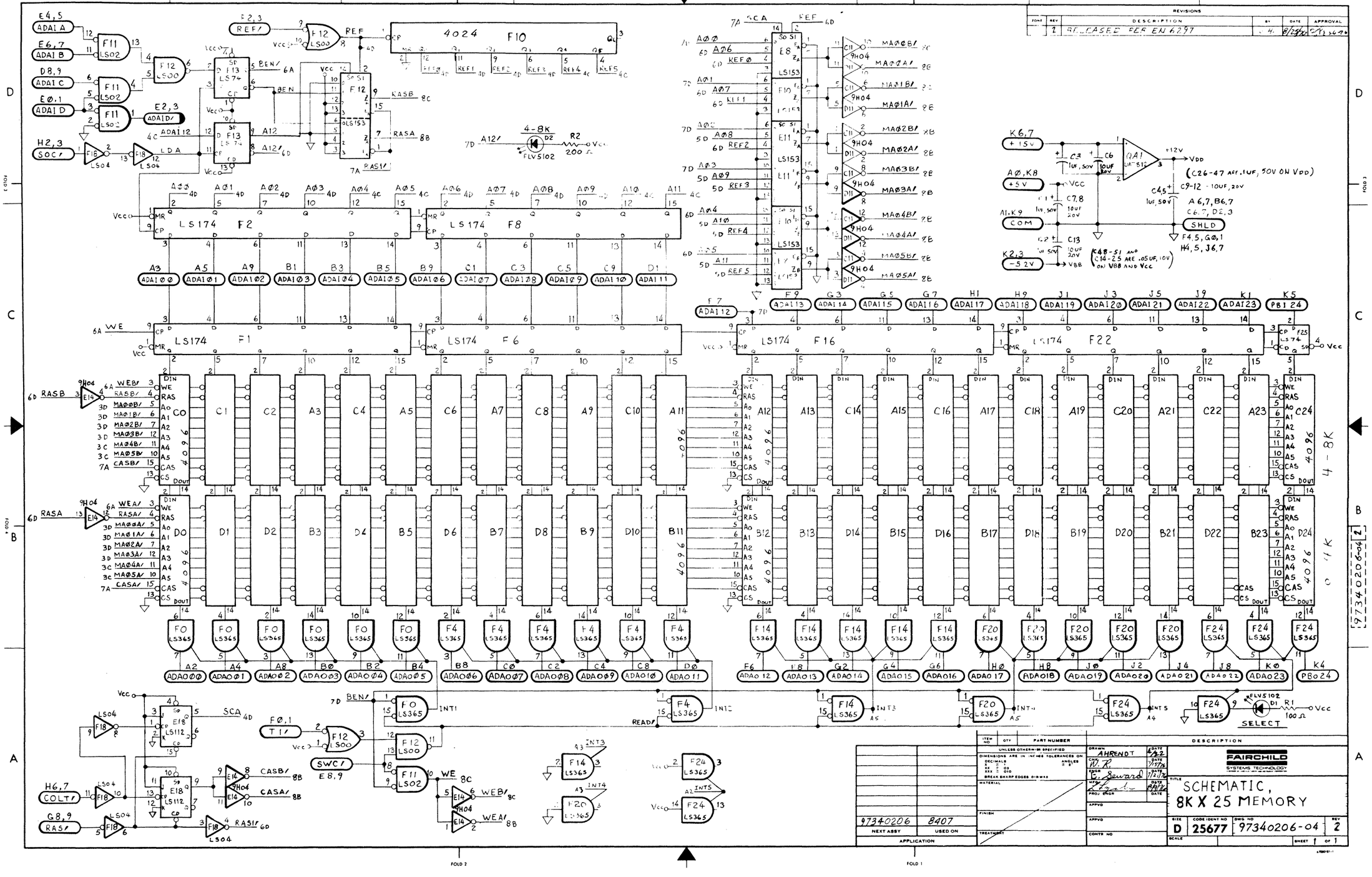


Figure 5-2. Memory Board Schematic

select flip-flop F13-9 and displayed as LED D2 which is lit with the upper 4K. Address bits 14 through 17 select one of the memory boards. The selected board has flip-flop F13-5 (BEN = board enable) set and LED D1 lit. All address registers are clocked with signal SOC (start of cycle). For a description of board selection and address translation, see paragraph 5.2.6.

Refresh addresses are generated in a 6 bit CMOS counter, F10.

The six least significant address bits represent the row address. The next six bits are the column address of the memory devices. Row address bits and column address bits enter the memory device on the same 6 pins. Therefore, they - and the refresh address - are multiplexed in devices E8, E10, E11, followed by buffer gates C11, D11, and then applied to the memory device matrix.

Data-In

Input data are loaded into a 25 bit register (F1, F6, F16, F22, F25) by clock WE (Write Enable), which consists of the start write pulse, SWC, ANDed with the board enable, BEN. The data-in register goes directly to the data inputs of the memory devices.

Data-Out

Output data of the memory devices are connected to the bus pins through 3-state gates which are enabled by a timing signal T1 if the board was selected and a read cycle occurred.

The control signals and their derivation are described in the following paragraphs. The memory device data sheet is shown as Figure 5-4.

5.2.2 Memory Write Cycle

See Figure 5-3 for the memory timing diagram. At time T3 the address is transferred over the bus and clocked with the trailing edge of SOC (start of cycle) into address, chip select and board select register. Signal RAS (row address strobe) clocks the row address into the memory device. Notice that multiplexer E12 generates an RAS for only those 25 memories whose board and chip select is true. This means that at any one time only 25 devices are active, and the remaining part of the system only draws standby power.

Signal COLT (column timing) has two functions: its leading edge generates SCA (select column address) in flip-flop E18-5. SCA switches the address multiplexer from row to column address. The trailing edge of COLT starts signal CAS (column address strobe), generated in flip-flop E18-9. The leading edge of CAS strobes the column address into the memory device.

Signal SWC (start write cycle) loads data-in into the data-in register with its leading edge. It also provides the write enable signal to the memory devices, and it blocks output data from being put on the bus during T1.

The trailing edge of RAS also terminates signals SCA and CAS and thus ends the memory access cycle.

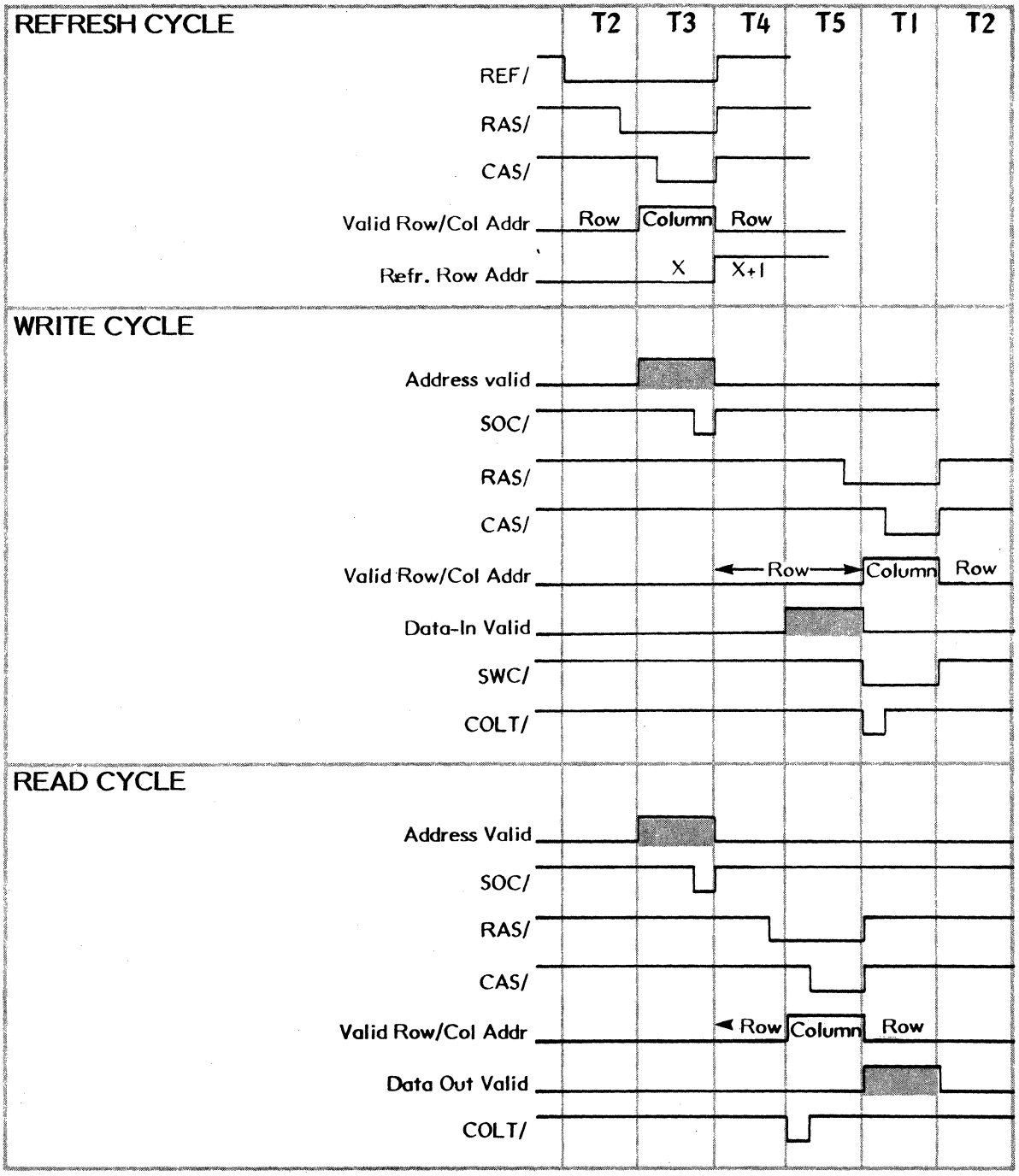


Figure 5-3. FST-2 Memory Timing Diagram

5.2.3 Memory Read Cycle

The read cycle (see Figure 5-3) is identical to a write cycle, with the following exceptions:

(1) Signals RAS and COLT are earlier by 1 T-time with respect to SOC. This reflects the fact that write data are available from the CPU at T5, but that data-out (memory contents) has to be made available to the CPU already at T1.

(2) Signal SWC (which indicates a write) is not generated.

(3) At time T1 the output data of the selected board and chip are gated onto the bus.

5.2.4 Memory Refresh Cycle

Each 15 CPU cycles, i.e., each 15 x 1.75 = 26.25 us one row of every memory device in the system is refreshed. The refresh cycle is squeezed in between normal read or write. The refresh address is supplied by the refresh counter (F10) which is free-running and which is incremented at the end of the refresh cycle.

A refresh cycle starts with the leading edge of signal REF. REF overrides board and chip select and gates RAS to all memory devices (in selector E12). It also selects the refresh counter outputs as the address source (in mux E8, E10, E11). Its trailing edge increments the refresh counter.

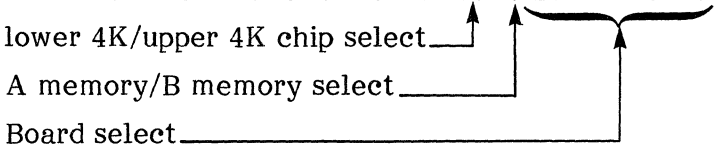
5.2.5 Parity

The 25th bit of each memory word is an even parity bit. If a non-existing memory location is addressed, the bus will stay at a "1" level during T1 when output data are expected. As 25 ones represent odd parity, a parity error is generated. For a description of the parity generation and check circuits, see Section 2 (CPU Clock A Board Description).

5.2.6 Address Translation and Board Selection

The fact that the overall memory is split into an A half (even addresses) and a B half (odd addresses) causes a discrepancy in the addressing of words at the board level and at the system level. The following table shows the translation of addresses:

Address bit at	x													x	x			
Board level	0	1	2	3	4	5	6	7	8	9	10	11	12	-	A	B	C	D
System level	12	1	2	3	4	5	6	7	8	9	10	11	13	0	14	15	16	17



Board selection is done by backplane wiring of the 4 inputs ADAIA,B,C,D of a NOT-AND gate (F11,F12). These inputs come directly from bits 14-17 of the bus or from inverted bits 14-17 of the bus. In order to provide the inversion, each memory board can invert signal ADAID (pins E2,3).

The following table shows the straightforward binary decoding of the board selects for the 12 sets of boards which are characterized by their starting address (in decimal and octal) and their backplane location.

Starting Addr.		Card Slot		Address Bit			
dec.	octal	A-Memory	B-Memory	17	16	15	14
0	0	C4	C8	0	0	0	0
16K	40K	C2	D0	0	0	0	1
32K	100K	C0	D2	0	0	1	0
48K	140K	B8	D4	0	0	1	1
64K	200K	B6	D6	0	1	0	0
80K	240K	B4	D8	0	1	0	1
96K	300K	B2	E0	0	1	1	0
112K	340K	B0	E2	0	1	1	1
128K	400K	A8	E4	1	0	0	0
144K	440K	A6	E6	1	0	0	1
160K	500K	A4	E8	1	0	1	0
176K	540K	A2	F0	1	0	1	1

5.2.7 Memory Board Power

Input voltages to the memory system are +15V, +5V, 5.2V. The memory devices use +12V which is derived from the +15V supply by a series regulator.

4096×1 DYNAMIC RANDOM ACCESS MEMORY

Fairchild Isoplanar Silicon Gate MOS

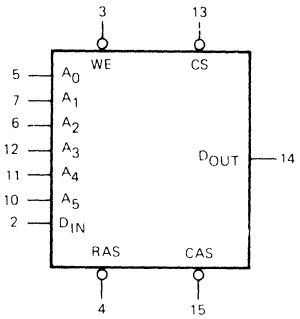
GENERAL DESCRIPTION – The 4096 is a 4096 x 1-bit dynamic Random Access Memory organized as 4096 one-bit words. This device is designed utilizing the single transistor dynamic memory cell.

A unique address multiplexing and latching technique permits the packaging of the 4096 in a standard 16-pin ceramic Dual In-line Package or Flatpak. The use of this package allows construction of highly dense memory systems utilizing widely available automated testing and insertion equipment.

The 4096 features direct TTL compatibility, on-chip address, data input and data output latches, TTL-level clocks with extremely low capacitance and a range of access times from 200 ns (4096-2) to 350 ns (4096-5). The 4096 is manufactured using the n-channel Isoplanar process and is available in the commercial (0°C to +70°C) or limited military (−55°C to +85°C) temperature ranges.

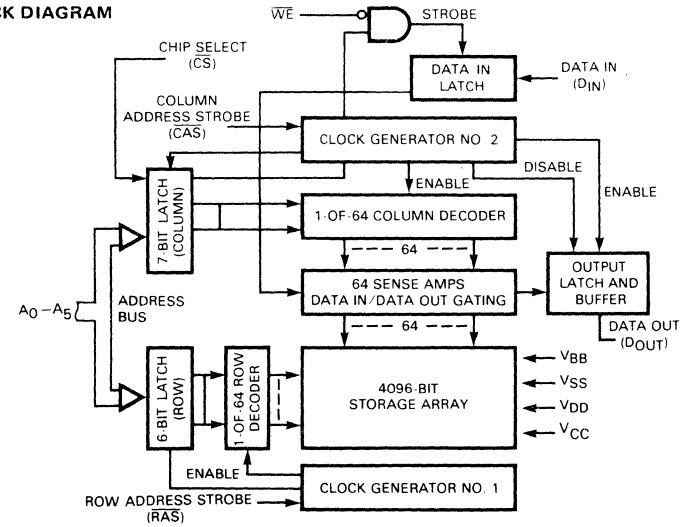
- **ALL INPUTS TTL-COMPATIBLE, INCLUDING CLOCKS**
- **ON-CHIP LATCHES FOR ADDRESSES, CHIP SELECT, DATA INPUT, DATA OUTPUT**
- **THREE-STATE TTL-COMPATIBLE OUTPUT**
- **CHIP SELECT DECODING DOES NOT ADD TO ACCESS TIME**
- **READ OR WRITE CYCLES: 4096-2: 300 ns, 4096-3: 360 ns, 4096-4: 420 ns, 4096-5: 500 ns**
- **ACTIVE POWER: 4096-2: <431 mW, 4096-3: <378 mW, 4096-4: <341 mW, 4096-5: <315 mW**
- **STANDBY POWER: <25 mW**
- **EXTENDED TEMPERATURE RANGE OPERATION (−55°C to +85°C)**
- **STANDARD 16-PIN CERAMIC PACKAGE**
- **STANDARD 16-PIN FLATPAK**

LOGIC SYMBOL

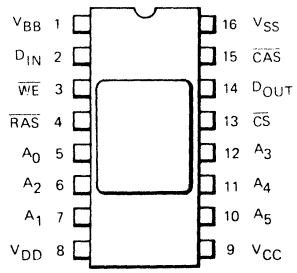


V_{SS} = Pin 16
V_{CC} = Pin 9
V_{DD} = Pin 8
V_{BB} = Pin 1

BLOCK DIAGRAM



CONNECTION DIAGRAM DIP (TOP VIEW)



NOTE:
The Flatpak version has the same pin-outs (Connection Diagram) as the Dual In-line Package.

Figure 5-4. Memory Device Data Sheet

FAIRCHILD MOS INTEGRATED CIRCUIT • 4096

PIN NAMES

An	Address Inputs	D_{OUT}	Data Output	
D_{IN}	Data Input	V_{CC}	+5 V	Power Supply
$\overline{CS}$	Chip Select Input	V_{SS}	0 V	Power Supply
$\overline{WE}$	Write Enable Input	V_{BB}	-5 V	Power Supply
$\overline{RAS}$	Row Address Strobe (Clock) Input	V_{DD}	+12 V	Power Supply
$\overline{CAS}$	Column Address Strobe (Clock) Input			

ABSOLUTE MAXIMUM RATINGS (Note q)

Voltage of any pin relative to V_{BB}	-0.5 V to +25.0 V
Operating Temperature	
Commercial Range	0°C to 70°C
Limited Military Range	55°C to 85°C
Storage Temperature (Ambient)	55°C to 150°C

ADDRESSING – The 12 address bits required to decode 1 of 4096 cell locations are multiplexed onto the 6 address pins and latched into the on-chip row and column address latches. The Row Address Strobe ($\overline{RAS}$) latches the 6 row address bits onto the chip. The Column Address Strobe ($\overline{CAS}$) latches the 6 column address bits plus Chip Select ($\overline{CS}$) onto the chip. Since the Chip Select signal is not required until well into the cycle, its decoding time does not add to the system access or cycle time.

DATA INPUT/OUTPUT – Data to be written into a selected cell is latched into an on-chip register by a combination of WE and $\overline{CAS}$. The last of these signals making its negative transition is the strobe for the Data In register. This permits several options in the write timing. In a write cycle, if the $\overline{WE}$ input is activated prior to $\overline{CAS}$, the Data In is strobed by $\overline{CAS}$ and the set-up and hold times are referenced to this signal. If the cycle is to be a read-write cycle or read-modify-write cycle, then the $\overline{WE}$ input will not go to a logic 0 until after the access time has elapsed. But now, because $\overline{CAS}$ is ready at a logic 0, the Data In is strobed in by $\overline{WE}$ and the set-up hold times are referenced to $\overline{WE}$.

At the beginning of a memory cycle the state of the Data Out Latch and buffer depend on the previous memory cycle. If during the previous cycle the chip was unselected, the output buffer will be in its open-circuit condition. If the previous cycle was a read, read-write, or read-modify-write cycle and the chip was selected, then the output latch and buffer will contain the data read from the selected cell. This output data is the same polarity (not inverted) as the input data. If the previous cycle was a write cycle ($\overline{WE}$ active low before access time) and the chip was selected, then the output latch and buffer will contain a logic 1. Regardless of the state of the output it will remain valid until $\overline{CAS}$ goes negative. At that time the output will unconditionally go to its open-circuit state. It will remain open circuit until after an access time has elapsed. At access time the output will assume the proper state for the type of cycle performed. If the chip is unselected, it will not accept a WRITE command and the output will remain in the open-circuit state.

INPUT/OUTPUT LEVELS – All inputs, including the two address strobes, will interface directly with TTL. The high impedance, low capacitance input characteristics simplify input driver selection by allowing use of standard logic elements rather than specially designed driver elements. Even though the inputs may be driven directly by TTL gates, pull-up or termination resistors are normally required in a system to prevent ringing of the input signals due to line inductance and reflections. In high speed memory systems, transmission line techniques must be employed on the signal lines to achieve optimum system speeds. Series rather than parallel terminations may be employed at some degradation of system speed.

The three-state output buffer is a low impedance to V_{CC} for a logic 1 and a low impedance to V_{SS} for a logic 0. The resistance to V_{CC} is 500 ohms maximum and 150 ohms typically. The resistance to V_{SS} is 200 ohms maximum and 100 ohms typically. The separate V_{CC} pin allows the output buffer to be powered from the supply voltage of the logic to which chips are interfaced. During battery standby operation, the V_{CC} pin may be unpowered without affecting the 4096 refresh operation. This allows all system logic except the $\overline{RAS}$ timing circuitry and the refresh address logic to be turned off during battery standby to conserve power.

REFRESH – Refresh of the cell matrix is accomplished by performing a memory cycle at each of the 64 row addresses every 2 milliseconds or less. Any read cycle refreshes the selected row, regardless of the state of the Chip Select. A write, read-write, or read-modify-write cycle also refreshes the selected row but the chip should be unselected to prevent writing data into the selected cell.

POWER DISSIPATION/STANDBY MODE – Most of the circuitry used in the 4096 is dynamic and draws power only as the result of an address strobe edge. Because the power is not drawn during the whole time the strobe is active, the dynamic power is a function of operating frequency. Typically, the power is 120mW at a 1 μS cycle time for the 4096DC with a worst case power of less than 341 mW at a 420 ns cycle time. To reduce the overall system power the Row Address Strobe ($\overline{RAS}$) must be decoded and supplied to only the selected chips. The $\overline{CAS}$ must be supplied to all chips (to turn off the unselected outputs). But those chips that did not receive a $\overline{RAS}$ will not dissipate any power on the $\overline{CAS}$ edges, except for that required to turn off the output. If the $\overline{RAS}$ is decoded and supplied to the selected chips, then the Chip Select input of all chips can be at a logic 0. The chips that receive a $\overline{CAS}$ but no $\overline{RAS}$ will be unselected (output open-circuited) regardless of the Chip Select input.

Figure 5-4. Memory Device Data Sheet (Continued)

RECOMMENDED DC OPERATING CONDITIONS: DC: T_A = 0°C to +70°C; DL, FL: T_A = −55°C to +85°C

SYMBOL	PARAMETER		MIN	TYP	MAX	UNITS	NOTES
V_{DD}	Supply Voltage	DC	11.4	12.0	12.6	V	2
		FL, DL	10.8	12.0	13.2		
V_{CC}	Supply Voltage		4.5	5.0	V_{DD}	V	2
V_{SS}	Supply Voltage		0	0	0	V	2, 12
V_{BB}	Supply Voltage		−5.5	−5.0	−4.5	V	2
V_{IH1}	Input HIGH Voltage Address Input, $\overline{CS}$		2.4	5.0	V_{DD}	V	2, 14
V_{IL}	Input LOW Voltage, All Inputs		−1.0	0	0.6	V	2, 14
V_{IH2}	Input HIGH Voltage, $\overline{RAS}$, $\overline{CAS}$, $\overline{WE}$		2.7	5.0	V_{DD}	V	2, 14

DC ELECTRICAL CHARACTERISTICS: (Over Temperature and Voltage Ranges)

SYMBOL	PARAMETER		PART NUMBER								UNITS	NOTES
			4096-2		4096-3		4096-4		4096-5			
			MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX		
I_{DD1}	Average V_{DD} Power Supply Current			35		30		27		25	mA	16
I_{CC}	V_{CC} Power Supply Current										mA	9
I_{BB}	Average V_{BB} Power Supply Current	DC		75		75		75		75	μA	
		FL, DL		150		150		150		150		
I_{DD2}	Standby V_{DD} Power Supply Current	DC		2.0		2.0		2.0		2.0	mA	
		FL, DL		2.5		2.5		2.5		2.5		
I_{IN}	Input Leakage Current (Any Input)			10		10		10		10	μA	10
I_{OUT}	Output Leakage Current			10		10		10		10	μA	11
V_{OH}	Output HIGH Voltage at I_{OUT} = −5 mA		2.4		2.4		2.4		2.4		V	
V_{OL}	Output LOW Voltage at I_{OUT} = 2 mA			0.4		0.4		0.4		0.4	V	
C_{IN1}	Input Capacitance (A_0 − A_5)			10		10		10		10	pF	
C_{IN2}	Input Capacitance ($\overline{RAS}$, $\overline{CAS}$, D_{IN}, $\overline{WE}$, $\overline{CS}$)			7		7		7		7	pF	
C_{OUT}	Output Capacitance (D_{OUT})			8		8		8		8	pF	

NOTES:

1. Stresses greater than those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.
2. All voltages referenced to V_{SS}.
3. Referenced to $\overline{RAS}$ leading edge.
4. Referenced to $\overline{CAS}$ leading edge.
5. Referenced to $\overline{CAS}$ trailing edge.
6. Write Command Hold Time is important only when performing normal random write cycles. During read-write or read-modify-write cycles, the Write Command Pulse Width is the limiting parameter.
7. Referenced to the $\overline{RAS}$ trailing edge.
8. Referenced to access time.
9. Depends upon output loading. The V_{CC} supply is connected only to the output buffer.
10. All device pins at 0 volts except V_{BB} at −5 volts and pin under test which is at +10 volts.
11. Output disabled by chip select input.
12. Output voltage will swing from V_{SS} to V_{CC} independent of differential between V_{SS} and V_{CC}.
13. These parameters are referenced to the $\overline{CAS}$ leading edge in random write cycle operation and to the $\overline{WE}$ leading edge in read-write or read-modify-write cycles.
14. Input voltages greater than TTL levels (0 to 5 V) require device operation at reduced speed.
15. Assumes t_{RCL} minimum.
16. Current is proportional to speed with maximum current measured at fastest cycle rate.
17. AC measurements assume ≈10 ns rise and fall times.

Figure 5-4. Memory Device Data Sheet (Continued)

RECOMMENDED AC OPERATING CONDITIONS: (over Temperature and Voltage Ranges)

SYMBOL	PARAMETER	PART NUMBER								UNITS	NOTES
		4096-2		4096-3		4096-4		4096-5			
		MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX		
t_{RC}	Random Read or Write Cycle Time	300		365		425		500		ns	3
t_{RAC}	Access Time from ROW Address Strobe		200		250		300		350	ns	3, 15
t_{CAC}	Access Time from Column Address Strobe		120		150		175		200	ns	4
t_{OFF}	Output Buffer Turn-off Delay	0	70	0	80	0	90	0	100	ns	4
t_{RP}	ROW Address Strobe Precharge Time	100		115		125		150		ns	
t_{RCL}	ROW to Column Strobe Lead Time	80		100		125		150		ns	3
t_{CPW}	Column Address Strobe Pulse Width	120		150		175		200		ns	
t_{AS}	Address Set-up Time	0		0		0		0		ns	3, 4
t_{AH}	Address Hold Time	50		60		70		80		ns	3, 4
t_{CH}	Chip Select Hold Time	70		80		90		100		ns	
t_{RCS}	Read Command Set-up Time	0		0		0		0		ns	4
t_{RCH}	Read Command Hold Time	30		35		40		45		ns	5
t_{WCH}	Write Command Hold Time	90		110		140		150		ns	4, 6
t_{WP}	Write Command Pulse Width	120		150		175		200		ns	
t_{CRL}	Column to ROW Strobe Lead Time	20		20		20	+20	20		ns	7
t_{CWL}	Write Command to Column Strobe Lead Time	120		150		175		200		ns	13
t_{DS}	Data In Set-up Time	0		0		0		0		ns	13
t_{DH}	Data In Hold Time	90		110		130		150		ns	13
t_{RFSH}	Refresh Period		2		2		2		2	ms	
t_{MOD}	Modify Time		10		10		10		10	μs	8

Notes on preceeding page.

TIMING DIAGRAMS

READ CYCLE

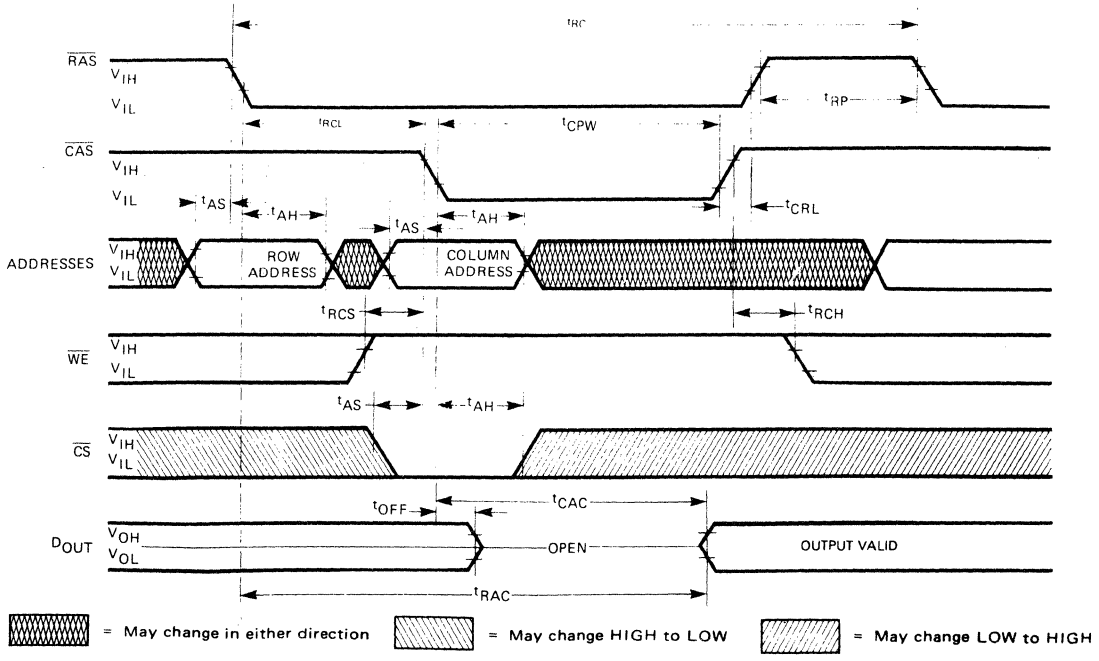


Figure 5-4. Memory Device Data Sheet (Continued)

TIMING DIAGRAMS (Cont'd)

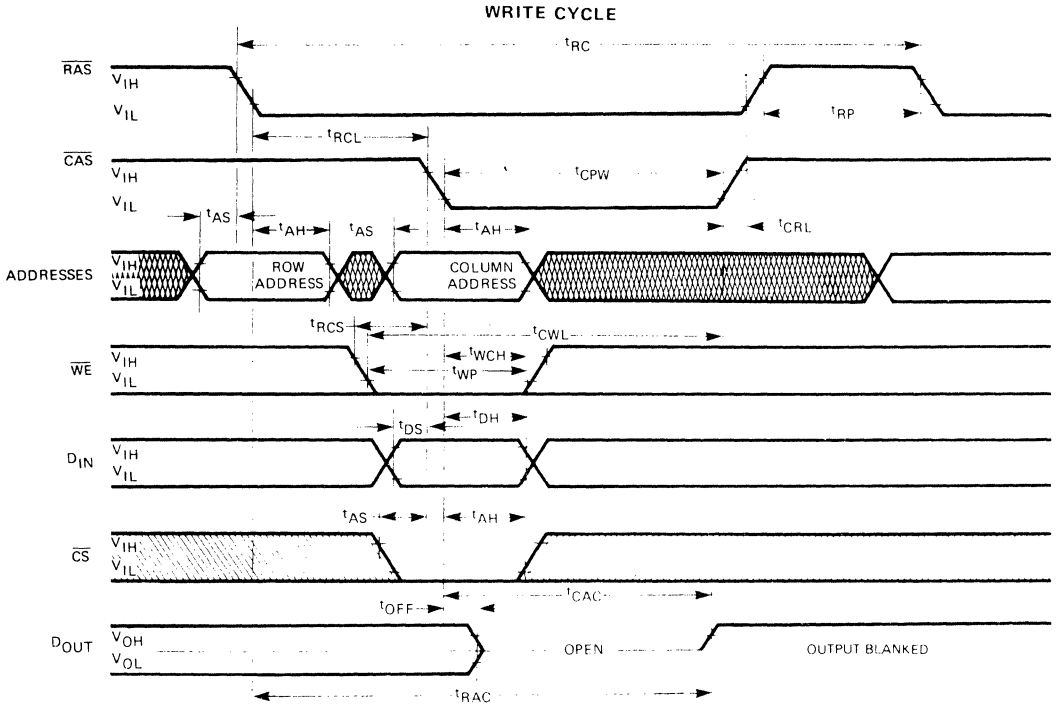


Figure 5-4. Memory Device Data Sheet (Continued)

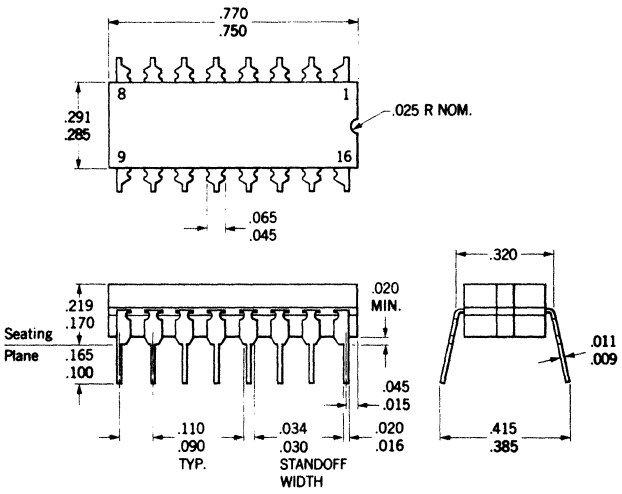
FAIRCHILD DYNAMIC RANDOM ACCESS MEMORY • 4096DC

ORDERING INFORMATION

PART NUMBER	PACKAGE	TEMPERATURE RANGE	ACCESS TIME
40962DC	CerDIP	0°C to +70°C	200 ns
40962DL	CerDIP	−55°C to +85°C	200 ns
40962FL	Flatpak	−55°C to +85°C	200 ns
40963DC	CerDIP	0°C to +70°C	250 ns
40963DL	CerDIP	−55°C to +85°C	250 ns
40963FL	Flatpak	−55°C to +85°C	250 ns
40964DC	CerDIP	0°C to +70°C	300 ns
40964DL	CerDIP	−55°C to +85°C	300 ns
40964FL	Flatpak	−55°C to +85°C	300 ns
40965DC	CerDIP	0°C to +70°C	350 ns
40965DL	CerDIP	−55°C to +85°C	350 ns
40965FL	Flatpak	−55°C to +85°C	350 ns

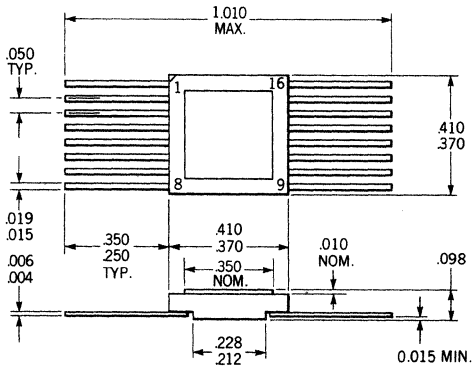
PACKAGE OUTLINES

16-Pin Ceramic Dual In-line
6D



NOTES:
1. Pins are tin-plated kovar
2. Pins are intended for insertion in hole rows on .300" centers
3. They are purposely shipped with "positive" misalignment to facilitate insertion
4. Board-drilling dimensions should equal your practice for .020 inch diameter pin
5. Hermetically sealed alumina package
6. Cavity size is .130 x .230
7. The .034-.030 dimension does not apply to the corner pins
8. Package weight is 2.2 grams

16-Pin 3/8-In. Flatpak
4D



NOTES:
All dimensions in inches
Pins are NiAu plated kovar
Cap is kovar
Base is Al_2O_3
Cavity size is .180 x .180
Package weight is 0.6 gram approx.

Figure 5-4. Memory Device Data Sheet (Continued)

SECTION 6

CPU PERIPHERAL INSTRUCTIONS

6.0 INTRODUCTION

This section describes in detail the operation of several FST-2 peripherals under the control of the SPU commands. See Appendix D for a complete set of SPU commands.

6.1 VIDEO KEYBOARD TERMINAL

A video keyboard terminal (VKT) is provided as standard equipment with the FST–2. This unit has a keyboard and a CRT display.

The characteristics of the VKT are given below:

Input/Output Speed	9600 BAUD/872 characters per second
Code	ASCII
Printable Characters	63
Characters per Line	73
Number of Lines	27

6.1.1 System Configuration

The VKT is interfaced to the FST-2 on the Accumulator Bus. The VKT controller contains two character buffers, one for input and one for output. With both the input and output buffered, input and output data transfers can be performed simultaneously at the maximum transfer rate.

The VKT controller utilizes the FST-2 interrupt system. When enabled, either a reader or a printer interrupt is generated, when a character is entered at keyboard or the printer (CRT) is ready to print. Either of these interrupts will cause program control to be transferred to an appropriate service program in memory.

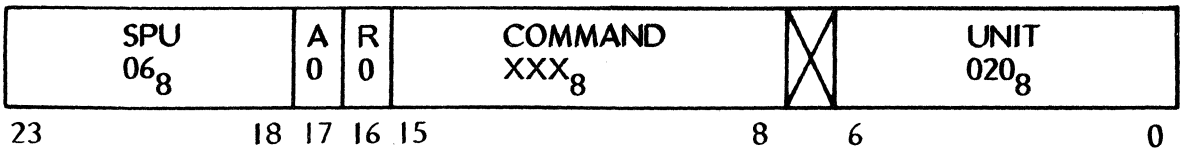
6.1.2 VKT Control

6.1.2.1 KEYBOARD COMMANDS (TTK). Six commands are available for testing and control of the console keyboard (TTK). These commands are variations of the Select Peripheral Unit (SPU) instruction. They differ only in the command code. The unit address remains constant. Each TTK command causes the general status of the controller to be signalled. General status is stored in the CPU indicators GT, EQ, LT, and BE where program tests may be conducted on the status. These indicators convey the following information when set:

GT	Idle
EQ	Idle with error
LT	Busy
BE	Not available

SPU SELECT PERIPHERAL UNIT

Instruction Format:



NOTE

Commands which the TTK controller is unable to accept are ignored.

STST STATUS TEST (06000020_8)

Description: Command Code 000_8

The Status Test command is a null command used to obtain the TTK status without changing the condition of the TTK controller.

Assembler Format: STST 20B

RDS READ STATUS (06611420_8)

Description: Command Code 023_8

The Read Status instruction execution returns controller status to the CPU Accumulator (A_{0-3}), as indicated below:

Bit 0 set: Read Error;
Bit 1 set: Read Interrupt In Process;
Bit 2 set: Read Interrupt Enabled;
Bit 3 set: SPU Command Code Error.

Assembler Format: RDS 20B (Read Status, TTK)

RDTT READ TT (06601420_8)

Description: Command Code 003_8

The Read TT (Keyboard) Data instruction execution transfers the contents of the TTK's Character Buffer into the CPU's Accumulator (A_{0-7}), providing:

(1) The TTK Controller is in the Idle or Idle Error State; and
(2) The TTK Buffer is Full.

Assembler Format: RDTT 20B (Read Keyboard)

PON TTK INTERRUPT ENABLE (06013020_8)

Description: Command Code 026_8

The TTK Interrupt Enable instruction sets the TTK interrupt control, enabling the TTK Controller to interrupt the CPU at the completion of an operation.

Assembler Format: PON 20B (Priority Interrupt ON)

POFF TTK INTERRUPT DISABLE (06011020_8)

Description: Command Code 022_8

The TTK Interrupt Disable instruction resets the TTK interrupt control, preventing the TTK Controller from interrupting the CPU.

Assembler Format: POFF 20B (Priority Interrupt OFF)

PCOMP TTK INTERRUPT COMPLETE (06001020_8)

Description: Command Code 002_8

The TTK Interrupt Complete instruction execution resets the Interrupt In Process control in the TTK Controller. Whenever the interrupt system is in use, the instruction must be executed at the termination of the interrupt service routine.

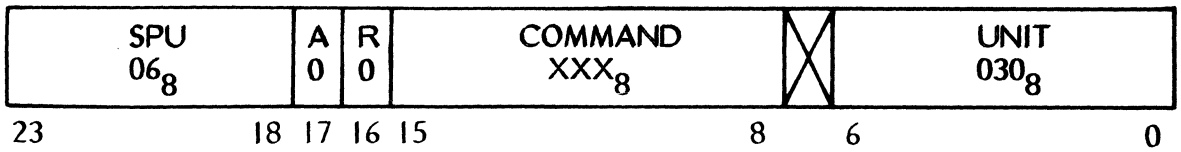
Assembler Format: PCOMP 20B (Priority Interrupt Complete)

6.1.2.2 Printer Commands (TTP)

Six commands are provided for the testing and control of the CRT console display (TTP). The unit address for the printer is 030_8; it remains constant for all commands for the printer. Each printer command causes the general status of the controller to be signalled to the CPU, where general status is stored in the indicators GT, EQ, LT, and BE. Program tests may be performed on these indicators to interpret device status. These indicators convey the following information when set:

GT	Idle
EQ	Idle with error
LT	Busy
BE	Not Available

Instruction Format:



NOTE

Commands which cannot be accepted by the printer controller are ignored.

STST STATUS TEST (06000030_8)

Description: Command Code 000_8

The Status Test command is a null command used to obtain the printer status without changing the condition of the printer controller.

Assembler Format: STST 30B

RDS READ STATUS (06611430_8)

Description: A=1, R=1, Command Code 023_8

The Read Status instruction execution returns controller status to the CPU Accumulator (A_{0-3}), as indicated below:

Bit 0 set: Print Error
Bit 1 set: Printer Interrupt In Process
Bit 2 set: Printer Interrupt Enabled
Bit 3 set: Printer Command Code Error

Assembler Format: RDS 30B (Read Status Printer)

WRIT SELECT AND PRINT (06421430_8)

Description: Command Code 043_8

The Select and Print (Write) instruction execution transfers the contents of the CPU Accumulator (A_{0-7}) to the Printer Buffer. The contents of the buffer is then transferred to the teletype, causing a character to be printed, providing:

(1) The Printer Buffer is empty,
(2) The Printer Controller is in the Idle or Idle With Error state.

Assembler Format: WRIT 30B (Select and Print)

PON PRINTER INTERRUPT ENABLE (06013030_8)

Description: Command Code 026_8

The Printer Interrupt Enable instruction sets the printer interrupt control, enabling the Printer Controller to interrupt the CPU when the printer is ready.

Assembler Format: PON 30B (Priority Interrupt On)

POFF PRINTER INTERRUPT DISABLE (06011030_8)

Description: Command Code 022_8

The Printer Interrupt Disable instruction resets the printer interrupt control, preventing the Printer Controller from interrupting the CPU.

Assembler Format: POFF 30B (Priority Interrupt Off)

PCOMP PRINTER INTERRUPT COMPLETE (06001030_8)

Description: Command Code 002_8

The Printer Interrupt Complete instruction execution resets the Interrupt In Process control in the Printer Controller. Whenever the interrupt system is in use, the instruction must be executed at the termination of the interrupt service routine.

Assembler Format: PCOMP 30B (Priority Interrupt Complete)

6.2 CARD READER

The FST-2 Card Reader System uses a 300 card per minute mechanism which reads cards serially by columns. This unit is interfaced to the FST-2 system with a controller which provides full word buffering between the card reader and memory.

There are eight commands which provide the programmer with the capability of testing the status of the Card Reader System (CRS), reading cards in either a binary or BCD mode, and for controlling card reader interrupts.

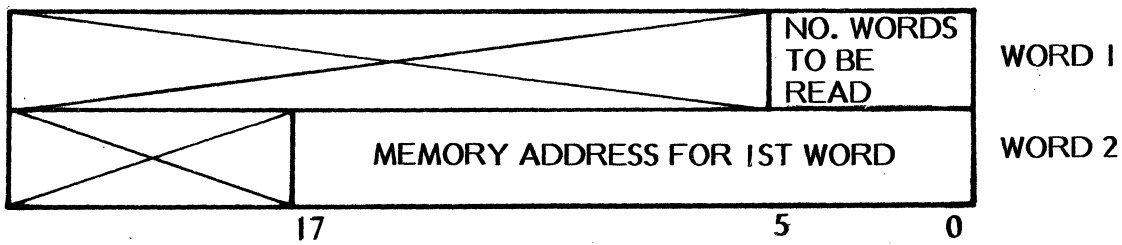
6.2.1 System Configuration

The CRS in interfaced to the FST-2 DMA (Direct Memory Access) System and the Accumulator Bus System. Commands are directed to the CRS with the SPU (Select Peripheral Unit) instruction. The CRS's unit address is 40_8. Data transfers from the CRS to memory are automatically handled by the DMA System. The CRS also uses the DMA System to access Data Control Words (DCW's) stored in memory.

When the CRS Interrupt System is enabled, it will automatically generate an interrupt at the end of a Card Reader operation. This interrupt will cause program control to be transferred to a card reader service program which is stored in memory.

6.2.2 Card Reader Instructions

6.2.2.1 INITIALIZATION. The CRS (Card Reader System) reads a variable number of words from a card and transfers these words into sequential memory locations using the DMA system. The Programmer must specify both the number of words to be transferred and the memory address where the 1st word read is to be stored. These parameters are passed to the CRS through two words stored in memory. Before initiating a read operation in the CRS, the programmer must store these words in memory. The format is as follows:



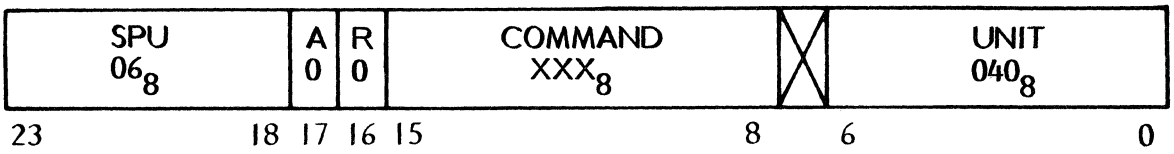
CARD READER DATA CONTROL WORDS

The 1st DCW specifies the total number of words to be transferred from the CRS to memory. The 2nd DCW specifies the memory address of the 1st word to be read by the CRS.

The location of the 1st DCW is transferred to the CRS during the execution of the Read Card instruction. The programmer must load the address of the 1st DCW into the accumulator, prior to the execution of the Read Card command. The accumulator-to-peripheral transfer is specified in the instruction if bit-17 is one and bit-16 is zero.

6.2.2.2 CARD READER STATUS TRANSFERS. The card reader returns its General Status to the CPU, as described below in the Status Test instruction description, for all Card Reader SPU instructions except the Error Test instruction. If the card reader cannot accept a command, the command is ignored, but the General Status, which indicates why the command was ignored, is returned to the CPU. After executing a Card Reader instruction, a branch on the indicators GT, EQ, LT, or BE will determine if the SPU command was accepted by the CRC and, if not, why.

6.2.2.3 CARD READER INSTRUCTIONS, FORMATS AND DESCRIPTIONS. The eight forms of the SPU instructions recognized by the CRC are illustrated and described below. The instruction format for each follows:



STST STATUS TEST (06000040_8)

Description: Command Code 000

The Card Reader Status Test instruction tests the General Status of the CRS without disturbing the state of the controller. The status of the controller is set into the GT, EQ, LT, and BE indicators, with the following meaning:

GT	Card Reader is Idle with no Error;
EQ	Card Reader is Idle with an Error;
LT	Card Reader Busy;
BE	Card Reader Not Available.

Assembler Format: STST 40B (Status Test Unit 40)

RD READ CARD BINARY (06401440_8)

Description: Command Code 003_8

The Read Card Binary instruction sets the CRS to the read binary mode. The CRS accepts the address of the DCW from the CPU accumulator and reads the number of words specified in the 1st DCW into consecutive memory locations, starting with the memory address specified in the 2nd DCW. In this mode, two card columns are packed into each word. The maximum number of words which can be read per Read Card Binary instruction is 40_{10}. Each instruction reads one card.

Assembler Format: RD 40B (Read Unit 40)

ARD READ CARD BCD (06403440_8)

Description: Command Code 007_8

The Read Card BCD instruction sets the CRS to the read Binary Coded Decimal mode. The CRS accepts the address of the DCW from the CPU accumulator and reads the number of words specified in the 1st DCW into consecutive memory locations, starting with the memory address specified in the 2nd DCW. In this mode the CRC packs 4 BCD characters into each word which is transferred to memory. The maximum number of words which can be read with this instruction is 20_{10}. Each instruction reads one card.

Assembler Format: ARD 40B (Alternate Read Unit 40)

PON CARD READER INTERRUPT ENABLE (06013040_8)

Description: Command Code 026_8

The Card Reader Interrupt Enable instruction sets the card reader's interrupt control, allowing the CRC to interrupt the CPU at the termination of the card reader's read operation.

Assembler Format: PON 40B (Priority Interrupt ON)

POFF CARD READER INTERRUPT DISABLE (06011040_8)

Description: Command Code 022_8

The Card Reader Interrupt Disable instruction resets the card reader's interrupt control, preventing the CRC from interrupting the CPU.

Assembler Format: POFF 40B (Priority Interrupt OFF)

PCOMP CARD READER INTERRUPT COMPLETE (06001040_8)

Description: Command Code 002_8

The Card Reader Interrupt Complete instruction resets the interrupt in process control in the CRC. Whenever the interrupt system is in use, this instruction must be executed at the termination of an interrupt service routine.

Assembler Format: PCOMP 40B (Priority Interrupt Complete)

ETST ERROR TEST (06010040_8)

Description: Command Code 020_8

The ERROR TEST instruction will read the CRC error status into the CPU GT, EQ, LT, and BE indicators, with the following meaning:

Indicator State				Meaning
GT	EQ	LT	BE	
0	0	0	0	No Error.
X	X	X	1	Data Control Block Error - Word Count too large/ Memory Address out of Range.
X	X	1	X	Data Overflow - CRC cannot obtain access to Memory.
X	1	X	X	Card Reader Device Error - Validity Error, Card Jam, etc.
1	X	X	X	End of file.

"X" indicates that the other conditions may or may not be present. The ERROR TEST instruction neither interrupts nor initiates any CRC operation.

Assembler Format: ETST 40B (Error Test Unit 40)

RDS READ STATUS (06611440_8)

Description: A=1, R=1, Command Code 023_8

The Read Status instruction will transfer the Card Reader Status to the CPU accumulator. The error status indicators in the CRS are reset unless the CRS is Busy (LT indicator set after SPU).

Table 6-1 shows the meaning of the accumulator bits when they are set to the "1" state by the Read Status Command.

Assembler Format: RDS 40B (Read Status for Unit 40)

TABLE 6-1 CARD READER CONTROLLER STATUS REGISTER

Bit Position	Logic Name	Logic Meaning When Set
0	RDY	Card Reader is "READY"
1	INP	Interrupt is in process
2	INA	Interrupt is enabled
3	BINALPHF	CRCU is an Alpha mode (Hollerith)
4	COMINT	CRCU is attempting to initiate an interrupt
5	MOVFLO2	CPI Memory Address Register contains an illegal address
6	DCBERR	Illegal memory address has occurred while processing an SPU or initial word count was zero (data control block error)
7	DATOV	CRCU could not get access to memory in time to service reader, therefore, a data overflow occurred
8	RDCK	Card Reader has detected a read error
9	SPARE	
10	INVCM	CRCU has detected an invalid SPU command
11	VALERR	CRCU has detected an invalid Hollerith punch code. (Validation error)
12	CDJAM	Card Reader has detected a card jam
13	FDCK	Card Reader has detected a feed error

TABLE 6-1 CARD READER CONTROLLER STATUS REGISTER (Continued)

Bit Location	Logic Name	Logic Meaning When Set
14	MPROF	CPI Memory Protect switch is disabled
15	WCZ	Word Counter equals zero
16	WCIL	Word Counter equals one or less
23	VALDAT	This register contains valid data when this bit is set

TABLE 6-2 CARD READER CONTROLLER CODE CONVERSIONS

Octal TRASCII	7-Bit ASCII	ALPHA	HOLLERITH	System Character	029 Equiv.
00	040	20	NO PUNCH	SPACE	
01	041	00	11-8-2	!	
02	042	17	8-7	"	
03	043	13	8-3	#	
04	044	53	11-8-3	$	
05	045	34	0-8-4	%	
06	046	60	12	&	
07	047	15	8-5	'	
10	050	75	12-8-5	(	
11	051	55	11-8-5	)	
12	052	54	11-8-4	*	
13	053	76	12-8-6	+	
14	054	33	0-8-3	,	
15	055	40	11	- (Minus)	
16	056	73	12-8-3	.	
17	057	21	0-1	/	
20	060	12	0	0	
21	061	01	1	1	
22	062	02	2	2	
23	063	03	3	3	
24	064	04	4	4	
25	065	05	5	5	
26	066	06	6	6	
27	067	07	7	7	
30	070	10	8	8	
31	071	11	9	9	
32	072	32	0-8-2	:	0-8-2
33	073	56	11-8-6	;	
34	074	72	12-0	<	12-0
35	075	16	8-6	=	
36	076	52	11-0	>	11-0
37	077	37	0-8-7	?	
40	100	14	8-4	@	
41	101	61	12-1	A	
42	102	62	12-2	B	
43	103	63	12-3	C	
44	104	64	12-4	D	
45	105	65	12-5	E	
46	106	66	12-6	F	
47	107	67	12-7	G	
50	110	70	12-8	H	
51	111	71	12-9	I	
52	112	41	11-1	J	
53	113	42	11-2	K	
54	114	43	11-3	L	
55	115	44	11-4	M	

TABLE 6-2 CARD READER CONTROLLER CODE CONVERSIONS (Cont'd.)

Octal TRASCII	7-Bit ASCII	ALPHA	HOLLERITH	System Character	029 Equiv.
56	116	45	11-5	N	
57	117	46	11-6	O	
60	120	47	11-7	P	
61	121	50	11-8	Q	
62	122	51	11-9	R	
63	123	22	0-2	S	
64	124	23	0-3	T	
65	125	24	0-4	U	
66	126	25	0-5	V	
67	127	26	0-6	W	
70	130	27	0-7	X	
71	131	30	0-8	Y	
72	132	31	0-9	Z	
73	133	74	12-8-4	[	<
74	134	57	11-8-7	\	¬
75	135	36	0-8-6	]	>
76	136	77	12-8-7	↑	|
77	137	35	0-8-5	←	—

All other hole punch combinations are illegal in the Read Alpha mode and will be flagged by "Validity Error, S11".

TABLE 6-3 CARD READER CONTROLLER CARD TO WORD COUNT CONVERSION TABLE

Number of Cards	Octal Word Count	
	Binary	Alpha
1	50	24
2	120	50
3	170	74
4	240	120
5	310	144
6	360	170
7	430	214
8	500	240
9	550	264
10	620	310
11	670	334
12	740	360
13	1010	404
14	1060	430
15	1130	454
16	1200	500
17	1250	524
18	1320	550
19	1370	574
20	1440	620
30	2260	1130
40	3100	1440
50	3720	1750
100	7640	3720
200	17500	7640
300	27340	13560
400	37200	17500

COMPUTATION EXAMPLE: NUMBER OF CARDS TO BE READ IN BINARY IS 225_{10}.

OCTAL WORD COUNT IS

	CARDS	WC	
	200	17500	
	20	1440	
	5	310	
DECIMAL SUM	225	21450	OCTAL SUM

CARD WORDS

21450_8 IS WORD COUNT

This table is provided as an aid for use of the multiple card read capability of the CRC.

6.3 LINE PRINTER

The line printer is interfaced to FST-2 on the Accumulator Bus.

The line printer accepts one character from 'A' register for each WRIT 60B instruction and stores it in an internal 20 character buffer. (Characters must be in standard ASCII code.) If accumulator contains an "operation code" (coded to represent TOF, LINE FEED, or CARRIAGE RETURN) the appropriate operation will be performed.

6.3.1 To Perform WRIT 60B Operation

```
LDA octal
WRIT 60B
```

where octal is one of the following values:

Octal Value	Description
12	LINE FEED (with format)
14	TOP OF FORM
15	CARRIAGE RETURN (no line feed) prints contents of 20 character buffer
40 . . . 137	standard ASCII characters

6.3.2 Summary of Line Printer Instructions

Octal Code	Assembler Mnemonic	
06000060	STST	60B
06013060	PON	60B
06011060	POFF	60B
06001060	PCOMP	60B
06421460	WRIT	60B

6.4 DISC FILE

The Disc File subsystem, Model 2060, uses a high-speed, head-per-track, disc unit. The organization of this system provides the FST Central Processor with a data base of 737,280 words at a transfer rate of 113,000 words per second.

The Disc File format provides 192 useable tracks*, 80 segments per track with 48 words per segment. The Disc Controller provides one six bit parity check per segment.

Data access of the Disc/Disc Controller requires a BCD track and segment address word. Track addresses are BCD '000' to '191'. Segment addresses are BCD '00' to '79'.

6.4.1 System Configuration

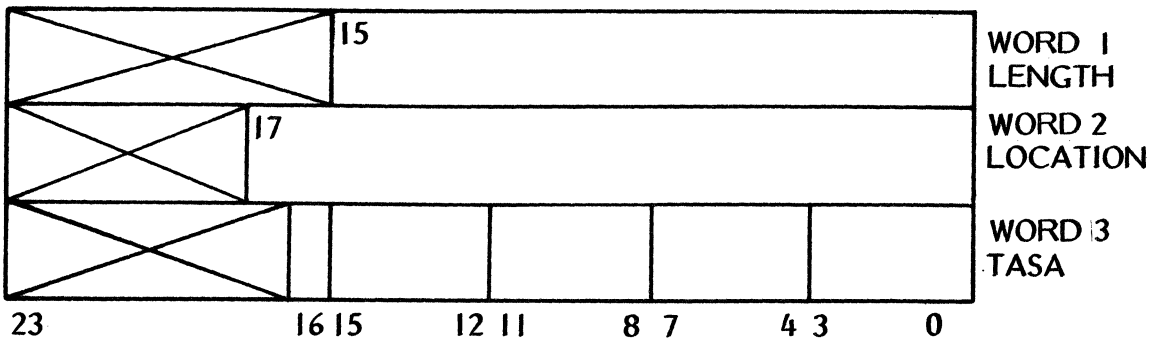
The Disc Control Unit (DCU) is interfaced to the FST-2 Central Processor (CPU) through the Common Peripheral Interface (CPI). The CPI provides the DCU with direct access to memory, access to the accumulator bus and required interrupt features.

* Some older systems may use a disc with 200 tracks.

The DCU provides a six bit character format to the Disc, decomposing and reconstructing 24 bit words. Variable length word files up to the memory size of the FST-2 Central Processor can be handled by the disc subsystem. The DCU is capable of interrupting the Central Processor upon normal completion of commands or upon error terminations.

6.4.2 Disc Subsystem Instructions

6.4.2.1 INITIALIZATION. The programmer must specify three data control words for use by the DCU:



Word 1, File Length: specifies the number of words in the file. This number may be any value up to 65,535. A value of n=0 will terminate before operation. Values greater than 65,535 will be truncated at 16 bits and used as a valid file length.

Word 2, Location: specifies the memory address of the first word for which the file is a source or destination.

Word 3, TASA: contains the TRACK and SEGMENT address. The segment address is two BCD characters right justified in WORD 3. The range of the segment address is 00 to 79. Values greater than 79 will cause a "DCB ERROR" and will terminate the DCU operation.

The track address is two BCD characters plus one bit. The track address is in bit positions 8 through 16 of WORD 3. The range of the track address is 000 through 191.

Each TASA character is checked and if it exceeds a BCD "9", a "DCB Error" will be noted.

The address of the first data control word must be in the CPU accumulator when a command is issued.

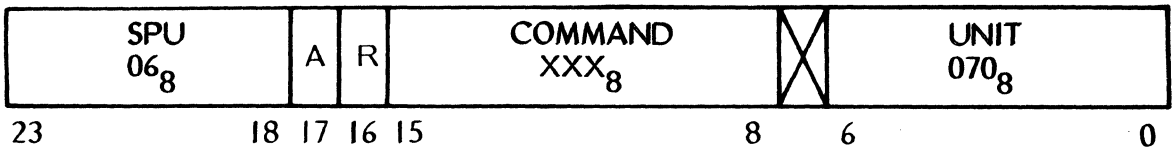
6.4.2.2 GENERAL STATUS. The Disc Control Unit, DCU, accepts ten commands which provide control for Read, Write, Parity Check, and various status and control functions. (See the command summary in Table 6-4.) The DCU controls the Disc Unit in such a way that the programmer may consider it as a continuous word-string file.

The DCU returns general status through the CPU indicators for all commands with the exception of the ERROR-TEST instruction. The meaning of these conditions is as follows:

GT	Idle
EQ	Idle Error
LT	DCU Busy
BE	Disc Not Ready

Program tests may be performed on the CPU indicators to interpret device general status.

6.4.2.3 INSTRUCTION FORMAT. Instructions for the DCU are Select Peripheral Unit type with a unit code of 70_8. The general format is:



STST STATUS TEST (06000070_8)

Description: Command Code 000_8

The Status Test command is a null command used to obtain the DCU status without changing the DCU controller condition.

Assembler Format: STST 70B

ETST ERROR TEST (06010070_8)

Description: Command Code 020_8

The Error Test command provides broad error status for the DCU. The ETST command uses the CPU indicators to store error information. This is the only command which does not return general status. The meaning of the indicators is as follows:

GT	Track Address Overflow
EQ	Parity Error
LT	Data Overflow
BE	Data Control Word Error

RDS READ STATUS (06611470_8)

Description: Command Code 023_8

Execution of the RDS instruction transfers the contents of the DCU status register to the CPU accumulator, clears the DCU error flip-flops and returns the DCU to

the Idle state. General status is stored in the CPU indicators. The meaning of the CPU accumulator bits is summarized in the following table:

Bit		
	0	Disc Not Ready
	1	DCU Parity Error
	2	DCU Interrupt Control active/enabled
	3	DCU Interrupt not manually inhibited
	4	DCU/CPI Memory Interface not manually inhibited
	5	DCU NORMAL/80 segments/track are addressable
	6	Data Overflow/memory not available when required
	7	Track Address Overflow
	8	Data Control Block Error (TASA or Memory Address out of bounds)
	9	Interrupt operation active
	10	Segment Not Found, Address Search not successful
	11	Disc Write not manually inhibited
	12	Disc Write not inhibited
	13	DCU has one or more error states

Status bit 5 reflects the condition of the manual switch which allows access to the "maintenance" segment (segment 80) on all tracks. (If the disc is good, only bits 3,4,5, 11, and 12 are set).

RD READ DISC (06401470_8)

Description: Command Code 003_8

The number of words specified by the first data control word is transferred to memory beginning at the location specified by the second data control word. A parity check is made on all segments read. The source of data read is specified by the track and segment address.

Assembler Format: RD 70B

ARD ALTERNATE READ (06403470_8)

Description: Command Code 007_8

This command performs the same function as RD except that data transfers to memory are inhibited. Parity is checked on all segments read.

Assembler Format: ARD 70B

PON DCU INTERRUPT ENABLE (06013070_8)

Description: Command Code 026_8

This instruction enables the Interrupt Control of the DCU.

Assembler Format: PON 70B

POFF DCU INTERRUPT DISABLE (06011070_8)

Description: Command Code 022_8

This instruction resets the DCU Interrupt Control, preventing an interrupt to the CPU.

Assembler Format: POFF 70B

PCOMP DCU INTERRUPT COMPLETE (06001070_8)

Description: Command Code 002_8

This instruction resets the Interrupt In Process Control in the DCU/CPI.

Whenever the interrupt system is used, this instruction must be executed at the termination of an interupt service routine.

Assembler Format: PCOMP 70B

WRIT WRITE DISC (06421470_8)

Description: Command Code 043_8

The number of words specified by the first data control word is transferred from memory starting from the address in the second data control word to the track/sector in the third data control word.

Assembler Format: WRIT 70B

TABLE 6-4 DISC COMMANDS

06 00 00 70	TEST NO-OP, STATUS TEST
06 01 00 70	ERROR TEST
06 00 10 70	INTERRUPT ROUTINE COMPLETE
06 01 10 70	INTERRUPT DISABLE
06 01 30 70	INTERRUPT ENABLE
06 61 14 70	READ STATUS
06 61 34 70	READ ALTERNATE STATUS, READ TRACK ADDRESS/SEGMENT ADDRESS
06 40 14 70	READ DISC BINARY
06 40 34 70	READ DISC ALTERNATE PARITY CHECK
06 42 14 70	WRITE DISC BINARY

6.5 MAGNETIC TAPE

The magnetic tape subsystem consists of a tape control unit (TCU) and up to two tape transports. Each transport is connected to the TCU by its own interface cable.

Recording format is 9-track, 800 BPI, NRZI, IBM compatible. Data rate is 6,400, words/sec. Tape speed is 24 IPS (refer to USAS 3.22 for a format description).

6.5.1 System Configuration

The Magnetic Tape Subsystem uses the Common Peripheral Interface (CPI) for interfacing with the CPU. It receives SPU instructions via the accumulator bus. Data is transferred by direct-memory-access (DMA).

The tape subsystem is assigned the following addresses and interrupt levels:

Device code:	$10X_8$ (X=0, 1 for one of up to two tape transports per TCU)
Interrupt address:	10_8
Interrupt priority:	7
Memory priority:	8

6.5.2 Tape Command Codes

The Tape Control Unit accepts eighteen forms of the SPU instruction. Table 6-5 summarizes all commands (except ARM) executed by the TCU.

All commands requiring data transfer operations use a Data Control Block (DCB). Refer to Table 6-6 for DCB and Data Format.

Instruction Format:

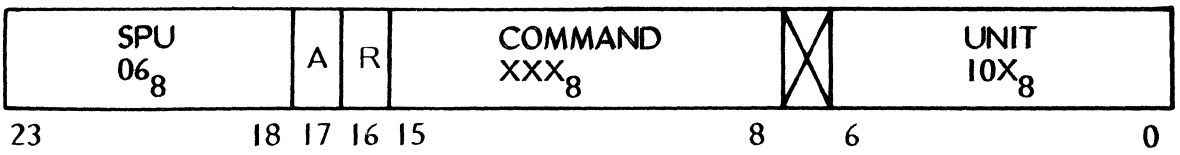


TABLE 6-5 TCU COMMAND CODES

COMMAND	Octal Code Generated by Assembler	17	16	15	14	13	12	11	10	9	8	7	6
Write Commands													
Write record WRIT	0642150X	1	0	0	0	1	0	X	X	1	1	X	1
Skip and write record SKWR	0646150X	1	0	0	1	1	0	X	X	1	1	X	1
Write tape mark WRITM	0606150X	0	0	0	1	1	0	X	X	1	1	X	1
Read Commands													
Read record RDT	0650150X	1	0	1	0	0	0	X	X	1	1	X	1
Alternate read record ART	0640150X	1	0	0	0	0	0	X	X	1	1	X	1
Advance one record RSKIPF	0600150X	0	0	0	0	0	0	X	X	1	1	X	1
Go back one record RSKIPB	0601150X	0	0	0	0	0	1	X	X	1	1	X	1
Advance to tape mark FSKIPF	0604150X	0	0	0	1	0	0	X	X	1	1	X	1
Go back to tape mark FSKIPB	0605150X	0	0	0	1	0	1	X	X	1	1	X	1
Logic Commands													
Rewind REWIND	0600050X	X	X	X	X	X	X	X	X	0	1	0	1
Read status RDS	0661150X	1	1	0	0	0	1	0	0	1	1	0	1
Read excess word count REWC	0661170X	1	1	0	0	0	1	0	0	1	1	1	1
Status test STST	0600010X	0	0	0	0	0	0	0	0	0	0	0	1
Error test ETST	0601010X	0	0	0	0	0	1	0	0	0	0	0	1
Enable interrupt PON	0601310X	0	0	0	0	0	1	0	1	1	0	0	1
Disable interrupt POFF	0601110X	0	0	0	0	0	1	0	0	1	0	0	1
Interrupt completed PCOMP	0600110X	0	0	0	0	0	0	0	0	1	0	0	1

TABLE 6-6 DCB AND DATA FORMATS

1. DCB address Word

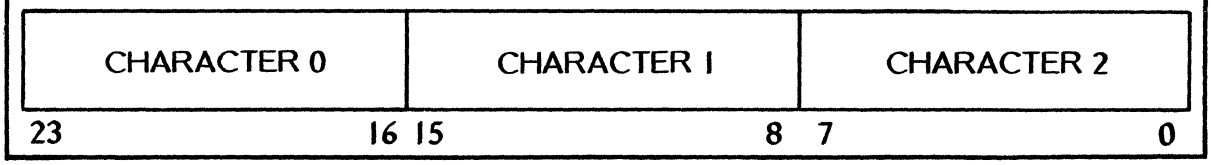
2. Word Count

The word count is stored at "DCB address". If the word count on "Read" is greater than the record length, the complete record will be read and stored in memory. "DCB error" and "Excess word count" (bit 13) will be flagged.

3. Start Address

The start address is stored at "DCB address" = 1. If the start address and word count are such that memory overflow occurs, enough data will be transferred between tape and memory to fill or read memory locations "start address" through "highest memory address". "DCB error" (bit 11) will be set.

4. Data, 9 track



The TCU, SPU has a device address field of three bits of which only three codes are used:

0 = TRANSPORT-0
1 = TRANSPORT-1
2 = TRANSPORT-2

Note that the TCU decodes bit 7 as a part of the command field.

STST STATUS TEST

Description: Assembler Octal - 0600010X

The Status Test command returns general status of the TCU into the indicators of the CPU.

General Status

BE	Device Not Available
LT	Device Busy
EQ	Device Idle with Error
GT	Device Idle with no Error

All SPU commands return general status except ETST. SPU commands will return status even when the TCU is Busy.

RDS READ STATUS

Description: Assembler Octal - 0661150X

The RDS instruction reads the TCU status register into the accumulator.

Status Register

Bit	Definition
0	Device Ready
1	Interrupt In Process
2	Interrupt Enabled
3	Interrupt Pending
4	Rewinding
5	No Write Enable Ring
6	Memory Protect Switch on
7	BOT
8	Low Density
9	Tape Mark has passed
10	Data Overflow
11	DCB Error
12	Rewind ended
13	Word Count > Record Length
14	Word Count < Record Length
15	Longitudinal Parity Error
16	Vertical Parity Error
17	EOT has passed
18	
19	
20	
21	
22	
23	Error Status set

REWC READ EXCESS WORD COUNT

Definition: Assembler Octal 0661170X

The TCU transfers the contents of the memory address register into the accumulator. This contains the last memory location accessed by the TCU.

ETST ERROR TEST

Definition: Assembler Octal 0601010X

The TCU returns error status to the CPU indicators.

Error Test	
BE	DCB Error
LT	Data Overflow
EQ	Parity Error
GT	Write: Write Enable Ring Not Present Read: Word Count Not Equal to Record Length

PON INTERRUPT ENABLE

Definition: Assembler Octal 0601310X

This enables the TCU to request an interrupt of the CPU. An interrupt will then occur after all instructions which require some time to be processed by the TCU.

POFF INTERRUPT DISABLE

Definition: Assembler Octal 0601110X

Disables TCU interrupt request feature.

PCOMP PRIORITY ROUTINE COMPLETE

Definition: Assembler Octal 0600110X

This instruction resets the TCU Interrupt In Process flag in preparation for future interrupt requests.

WRIT WRITE RECORD

Definition: Assembler Octal 0642150X

The TCU loads the word count register and memory-address register from the DCB. It starts the tape transport, and, if the tape is at the BOT marker, writes a 4-inch gap. It reads from consecutive memory locations and writes on tape. The tape is read while being written and the characters checked for parity.

Minimum record length is 12 characters (4 words). When the count reaches zero, the TCU writes end-of-record characters (CRC and LRC), puts in a gap and stops the transport.

If enabled, an interrupt will be issued to the CPU on completion of this command.

If the write enable ring is not in the tape reel, the transport is not started and status register bit 23 is set.

If a data overflow occurs, the TCU ends the record, sets bit 10 of the status register, writes a gap and halts.

If memory overflow occurs, the TCU ends the record and sets bit 11 of the status register.

If read-after-write check has a vertical parity (column) error, the TCU sets bit 16 of the status register. The record is completed.

SKWR SKIP AND WRITE RECORD

Definition: Assembler Octal 0646150X

Identical to "Write Record" with the exception that the TCU writes 4 inches of blank tape before it starts writing the record. If enabled, an interrupt will be issued to the CPU on completion of this command.

WRITM WRITE TAPE MARK

Definition: Assembler Octal 0606150X

The TCU starts the transport, writes a 4 inch gap followed by a tape mark (character 310_8) and a normal record gap. The TCU sets status bit 9, stops the transport, and, if enabled, generates an interrupt.

RDT READ RECORD TAPE

Definition: Assembler Octal 0650150X

The TCU loads the word-count register and MAR from the DCB. It starts the tape transport and enables reading after start-up distance. If the tape was at BOT, after two inches of tape travel the TCU reads tape, checks parity, assembles characters into words and writes them into consecutive memory locations. At the end of the record, it checks longitudinal parity, stops the transport, and, if enabled, generates an interrupt.

Minimum record length is four words.

If the record consists of a tape mark, status register bit 9 is set.

If data overflow occurs, the TCU stops transmission to memory and sets bit 10 of the status register. The TCU moves tape to the record gap and halts.

If memory overflow occurs, the TCU stops transmission to memory, sets bit 11 of the status register, moves tape to the record gap and halts.

If a parity error occurs, the TCU replaces the character in error with all ones, sets bit 16 of the status register and continues.

If the word count is less than the record length, the TCU stops transmission to memory, sets bit 14 of the status register and continues for parity check.

If the record is less than the word count, the TCU sets bit 13 of the status register, halts tape, and, if the character count MODULO 3 was non-zero, the last word is padded with all ones.

ART ALTERNATE READ RECORD TAPE

Definition: Assembler Octal 0640150X

Identical to "Read Record" except that data is not transmitted to memory. The word count is not checked. If enabled, an interrupt will be issued to the CPU on completion of this command.

RSKIPF READ, SKIP FORWARD, ONE RECORD

Definition: Assembler Octal 0600150X

The TCU moves tape in the forward direction, advances to the end of the record and stops in the record gap. If enabled, an interrupt will be issued to the CPU on completion of this command.

RSKIPB READ, SKIP BACKWARD, ONE RECORD

Definition: Assembler Octal 0601150X

The TCU moves tape in the reverse direction to the beginning of record and halts in the record gap. If enabled, an interrupt will be issued to the CPU on completion of this command.

FSKIPF FILE SKIP FORWARD

Definition: Assembler Octal 0604150X

The TCU moves tape forward until a tape mark record or End of Tape (EOT) is detected. The tape is stopped after the tape mark or EOT.

If a tape mark is detected, bit 9 of the status register is set.

If EOT is detected, bit 17 of the status register is set. If enabled, an interrupt will be issued to the CPU on completion of this command.

FSKIPB FILE SKIP BACKWARD

Definition: Assembler Octal 0605150X

The TCU moves tape backwards until a tape mark record or Beginning of Tape (BOT) is detected.

The tape is stopped before the tape mark. If a tape mark is detected, bit 9 of the status register is detected.

If BOT is detected, bit 7 of the status register will be true.

If enabled, an interrupt will be issued to the CPU on completion of this command.

REWIND REWIND TAPE

Definition: Assembler Octal 0600050X

The TCU initiates a rewind operation on the selected tape drive. During a rewind operation, the TCU is busy for six memory cycles.

When the selected drive detects a BOT marker the tape will stop. The TCU will set bit 12 of the status register, REWIND END.

If enabled, an interrupt will be issued to the CPU on completion of this command.

SECTION 7

ASSEMBLER

7.1 OPERATING INSTRUCTIONS

This section is concerned primarily with the mechanical motions required in using the assembler; that is, in getting a program assembled. Subsequent paragraphs give the information necessary to write programs that are acceptable to the assembler.

7.1.1 Hardware Configuration

The minimum hardware required by the assembler is:

(1) 16K Memory
(2) VKT (Video Keyboard Terminal)
(3) Disc File

The other I/O devices supported by the assembler are the card reader, line printer, and the magnetic tape unit. The card code required is that of the 029 keypunch (EBCDIC).

7.1.2 Loading the Program

The program source may be loaded from the console typewriter keyboard, cards, magnetic tape or a disc file.

Command input format:

// ASM [input/'file name'] (NLIST) (NOBJ) (NSYM)
(NXREF) (INSEQ) (FST1) (OUTPUT) (REL)

NLIST	Request for no source listing, no symbol table listing, no cross reference listing (external symbols and errors are still output)
NOBJ	Request for no object generation (OBJ directive in the program overrides this command)
NSYM	Request for no symbol table listing (external symbols and errors are still output)
NXREF	Request for no cross reference listing
INSEQ	Request for line sequence check
FST1	Request for FST-1 CPU object generation (default is FST-2 CPU object)
REL	Request hardware relocatable object

input	Standard DOPSY input device mnemonic (TTK, CR, MTR, TTR)
file name	name of the source program file on disc
output	Standard DOPSY output device mnemonic for listing (TTP, LP, MTW) (default is POD)

Object output is always placed in disc working storage.

Restrictions:

There are several types of object files produced by the assembler. It is the users responsibility to select proper type. It is discussed in detail in section 7.5.3.

7.1.3 Error Messages and Recovery

There are two types of errors; source language error and processing error.

Source language errors up to four combinations may be indicated at the line. (see output format). It is indicated by an alphabetic code as follows:

A	Address overflow (address exceeds the maximum allowed program size)
D	duplicate label
L	label error (start with a number or a special character)
O	non-existing opcode
U	undefined symbol usage
S	Syntax error (opcode-operand combination is illegal)
R	Relocation error (EQU operand is a forward reference)
↑	sequence error (INSEQ parameter in the command)
N	number too large
V	data field overflow (bit 23 is used as a data in relocatable memory reference instruction)

Processing errors:

Error Message	Description
MONITOR RECORD IGNORED	A source line starting with // has appeared. It is output during the 1st pass. The line is skipped.
XREF TABLE OVERFLOW-NO XREF LISTING	While building symbol list for XREF the memory is exhausted. The cross reference listing will not be produced. It is output during the 1st pass.

Error Message	Description
PARTIAL XREF LISTING TO XXXXX	While picking up references, the memory is exhausted at address XXXXX. The references including and beyond this address will not be included in the cross reference listing. It is output after the last reference line output.
MAG TAPE WRITE ERROR	While outputing listings to a mag tape write error occurred.
EOT FOILMARK. ASSEMBLY ABORTED	While outputing listings to a mag tape EOT FOIL mark is sensed. Assembler returns to monitor.
ERROR--DISC OVERFLOW	While generating object to working storage, disc space is exhausted. Assembler quits immediately.
EOF-END ASSUMED	End of file is reached without END statement.
ERROR--SYMBOL TABLE OVERFLOW	While building the symbol table, the memory is exhausted. Assembler quits immediately.
INVALID FILE NAME	Input mnemonic is not valid or specified file cannot be found on disc.
MAG TAPE I/O CONFLICT	Mag tape is specified as both input and output.
LINE PRINTER NOT READY	The line-printer specified as output is not on-line.
INVALID FILE TYPE	Specified file is not a source file.
SYSTEM-5	$ASM1 cannot be found on disc.

7.2 SYNTAX

This section gives the information necessary to produce programs in the format required by the assembler.

7.2.1 Character Set

Letters	A, B, C, ...Z, and $
Digits	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Special Characters	! " # % & ' () * + , - . / : ; <=>? @[\ ] ↑ SPACE

Table 6-2 shows the internal code (TRASCII) for the character set. These are the printing ASCII characters and are produced directly by the teletype. The 029 keypunch character set differs from the teletype character set in a few instances. These are indicated in this table under the '029' heading.

7 2.1.1 SYMBOLIC ADDRESSING. Symbols refer to assembly language instruction, data, constants and certain assembler directives. A symbol represents the address for a computer word in memory. It is defined when it is used as a label (i.e., in the label field) for a location in the program, a label of a data storage area, constant, or the label of a relocatable value.

Examples:

```
MINUS1    DATA   -1
SAVEIT    BSS    2
          .
          .
START     LDA    MINUS1
          STA    SAVEIT
BUFEND    EQU    *-1
```

The symbol MINUS1 defines the address of a word which contains the constant negative 1.

The symbol SAVEIT refers to the first word of a two word data storage area.

The symbol START and BUFEND define relative location in the program.

7.2.1.2 SYMBOLS. A symbol is either a single letter or a letter followed by one or more letters and/or digits. Symbols may be of any length, but only the first six characters are retained by the assembler. They must, therefore, be unique through the first six characters. Good practice dictates that symbols should be limited to six characters because of the risk of unintentional duplication.

Examples:

```
TEST10              BSS    20
A15                 DATA   158
AVERYLONGSYMBOL     EQU    A15     Only the first six
                                   characters are used.
$15C                LDA    A15
```

7.2.2 Record Format

The assembler obtains its input from either cards, a keyboard, magnetic tape, or disc files. Records obtained from the card reader are fixed length and occupy the first 72 characters of the record. Keyboard records are variable in length and are terminated by a carriage return; all other control characters are ignored. Disc files are normally variable length string files.

Assembler records are of two types: comment records and statement records.

7.2.2.1 COMMENT RECORD FORMAT. Comment records are characterized by having an '*' as the first character of the record. These records have no effect on the assembly process. They do occur in the assembly listing, however, and serve to document or explain program segments.

7.2.2.2 STATEMENT RECORD FORMAT. The fields of a statement record appear in the following order:

Label field
Opcode field
Operand field
Comment field

These fields are seperated by one or more spaces. Except for the restrictions on the label field, the input format does not require the fields to begin in any particular column. This format is referred to as free-field.

Example:

```
COL  1 2 3 4 . . . . . 11        16
     S T A R T           L D A   M I N U S 1
     S T A R T  L D A    M I N U S 1
```

These statements are equivalent to the Assembler.

7.2.2.3 LABEL FIELD. The assembler requires that the first character of a record be blank, or an alphabetic character.

If the first character is blank, then no label exists. If the character is alpha, then it is assumed to be the first character of a label. The label field identifies the statement and is used as a reference point by other statements in the program.

The label has the same syntax as a symbol and like a symbol may be any length, but only the first six characters are retained. The label must begin with the first column of the record and terminate with a SPACE.

The label must be unique within the program. Duplicate labels are flagged as errors.

7.2.2.4 OPCODE FIELD. The opcode field must be present and begins with the first non-blank character after the label terminator. It must contain an assembly language or an assembler opcode mnemonic. Appendix B contains a list of the allowed opcode mnemonics.

The syntax of the opcode field is the same as for symbols. The first special character terminates the field and if it is an '*' the instruction will be generated with indirect addressing, otherwise this character must be a space.

Examples:

```
FRST      LDA    STARS
          DSR    10
          ATX    X5, 1
TEST1     BSS    40B
  |        |
LABELS   OPCODES
```

7.2.2.5 OPERAND FIELD. The operand field begins with the first non-blank character after the opcode field terminator. If the last (or only) expression of the operand field is blank it is assumed to be zero. Appendix C contains the FST-2 opcodes sorted on the number of operand expressions required.

The operand field consists of one or more expressions. These expressions are used to represent storage locations, index registers and other constant values for instructions, and to provide information required by the assembler to process directives correctly.

The elements that make up expressions are:

Symbols
Numbers (octal or decimal)
The current location symbol (i.e., asterisk)
Combinations of the above joined together with the operators +, - and ↑ (up-arrow).

A single-term expression such as A5, 55B, or * has the value of the term itself while multi-term expressions, such as TABLE+3, are reduced by the assembler to a single value by arithmetically combining the terms.

Operand Symbols

The values assigned to operand symbols are either absolute or relocatable. An absolute value is one that is unaffected by the relocation of the program that contains it. A symbol can be assigned an absolute value only by using the EQU directive to equate it to an absolute numeric quantity or another absolute symbol. This implies that the other absolute symbol has already been equated to an absolute numeric quantity.

Examples:

```
A15   EQU   15B      The Symbol A15 will be assigned the value of OCTAL 15.
*
C15   EQU   A15      C15 is assigned the value of OCTAL 15.
```

Decimal/Octal Integers

A decimal number is a sequence of 6 or fewer digits. The range is ±799,999. An octal number is a sequence of 8 or fewer digits terminated by a 'B'. If more than 8 digits are present, only the low-order digits are retained.

Examples:

```
DEC15     DATA        15
OCT15     DATA        15B
MINUS1    DATA        -1
TOOBIG    DATA        332222222B
LABELS    OPCODES     OPERANDS
```

The constant TOOBIG will be reduced to 32222222B.

Current Location Reference

The value of an '*' in an operand expression is that of the current location counter. That is, the value is the location of the current instruction or data item.

Example:

```
DELTE    EQU    *
```

The label DELTE is assigned the value of the location counter.

Strings

A string is a sequence of four (or fewer) characters enclosed in single quotes. Any of the characters in Table 6-2 except the single quote may be part of a string. Each character is in 6-bit TRASCII code.

Strings are used in expressions as 24-bit binary numbers. If more than four characters are present, only the first four are retained and if there are less than four characters they are left justified in the word.

Examples:

```
ALFA     DATA   ' A'   will produce the constant 00000041B
ALFB     DATA   'B'    will produce the constant 42000000B
NUMB     DATA   '1234' will produce the constant 21222324B
```

NOTE

The TEXT directive allows a character string in excess of 4 characters but not to exceed column 72 of the input record, including the single quote which terminates the string.

Operators

Strings, numbers, symbols, and '*' can be combined by the operators, + (plus), – (minus), ↑(up arrow). ↑is always followed by only one number, symbol, or asterisk. + and - have the usual arithmetic meaning of add and subtract. ↑*has the value of the current address if the address is even, the current address +1 if it is odd. Its primary use is for aligning the location counter on an even boundary.

Examples:

```
LNGTH    DATA   LSTWRD-FRSTWD
BRTBL    BRU    *+ DVC + 1
LPDCB    EQU    LP+00500000B+20
```

Labels — Opcode — Operands

Example:

```
         ORG    ↑*
CLABEL   BSS    2
```

This example shows how to align to an even boundary.

7.2.2.6 COMMENT FIELD. The comment field begins with the first non-blank character after the operand field terminator (i.e., a space after the last operand expression). The comment field is not processed by the assembler, but does occur on the assembly listing. It can contain any of the characters in Table 6-2. Column 72 is the end of the comment field.

Note that the operand field cannot be omitted if the comment field is present because the assembler would assume the comment field is the operand field. This rule does not apply to those instructions which do not have operands, e.g., TCA and EXC.

Example:

```
                NOP         *               CALL+1 RETURN NOT USED
GETENT          LDA         TABLE+1, XP     GET TABLE ENTRY
                STA         TBLHLD          AND SAVE FOR PROCESSING
```

Label field — Opcode field — Operand field — Comment field

7.3 INSTRUCTIONS

The operand of every instruction must have a particular format. Appendix B contains a list of all of the legal mnemonics. Appendix C lists the opcodes by the type of operand required.

Operands are expression lists where the expressions reference memory locations or hardware features such as the comparison indicator, index registers, or state flip-flops. Some general comments concerning these expressions are:

(1) Expressions referencing hardware elements must be absolute (not relocatable). The magnitude of these expressions must not exceed 7 for index registers or 15 for states and indicators.

(2) An address expression is automatically truncated to 14 bits for regular instructions and to 10 for augmented ones. In the latter case the address expression cannot be relocatable.

In subsequent subsections the following notation is used.

(1) (address) is an expression for the instruction operand address,
(2) (state) is an expression referencing a state flip-flop,
(3) (indicator) is an expression referencing the comparison indicator,
(4) (index) is an expression referencing an index register,
(5) Items enclosed in square brackets '[' , ']' denote optional items,
(6) '. . .' is read as 'zero' or 'more of the following elements'.

7.3.1 Indexable Instructions

If the instruction (opcode) can be indexed its operand must be in the following form.

(address) [,(index)]

Examples:

```
LDA   TABLE-1, X5
STA   TEMP1
BRU   L2
BAH   * + 1
LDA*  0, X1
```

7.3.2 Non-Indexable Instructions

If the instruction cannot be indexed, its operand must be in one of the following forms:

(index) [,(address)]
(state), (address)
(indicator), (address)

Examples:

```
LAX   X3
LXA   X5
STX   X1, TEMP
BOS   10,L1  TEST SS 1
BOI   3,LEQ
LDX   X6, -2
```

The values of the states (switches) that can be tested by BOS are 0-15. These have the value shown below:

0-7	Defined by programmer (See SST, RST)
8	Interrupt Enable
9	Overflow
10-15	Console switches 1-6 respectively

7.3.3 SST, RST

RST and SST can be used to turn the eight programmable switches, interrupt enable, the overflow flag, and parity disable switch on or off. Their operands have the following form:

(state) . . . , (state)

```
Example   PD      EQU    14
          OV      EQU     9
          PASS1   EQU     2
                  SST    PASS1
                  RST    0,4,6, OV
                  SST    PD
```

Switches which can be set or reset are defined as follows:

0-7	State switch defined by program
8	Interrupt
9	Overflow
14	Parity Disable Switch

7.3.4 BAT, BOI Augments

These instructions require operands of the following form:

(address)

Example:

```
BNE   L1
BP    L2
BGE   * + 2
```

7.3.5 SPU Augments

These instructions require an operand expression that is the device number of the peripheral to be affected. This expression must be absolute and less than 200_8 in magnitude. A complete list of the instruction mnemonics may be found in Section 6 of this manual. See also Appendix D for a listing of all SPU commands.

Examples:

```
ARD   40B
STST  60B
```

7.3.6 Memory Reference

These instructions may reference absolute or relative addresses. The program with ABS directive provides absolute addresses for all labels. In addition, labels defined by EQU to absolute operand are also assigned absolute addresses. All other labels or the current address defined by '*' are considered relative.

Memory Reference instructions with relative addresses may contain one in relocation bit (bit 23). Refer to REL directive.

Examples:

```
$IOCS EQU    350B
      BSM*   $IOCS          absolute memory reference
LAB   EQU    *
      BRU    LAB           relative memory reference
```

Addresses in memory reference instructions may be 13 bits long (8K).

7.4 DIRECTIVES

A directive is a command to the assembler that allows the user to describe or select assembly options. It also allows the user to specify groups of data, character strings or storage areas to be defined and/or reserved within the boundries of the program.

The syntax for the directives is the same as the Command Syntax in the User Manual.

7.4.1 BSS

(label) BSS expression

This statement saves a block of storage N words in size; N is the value of the operand expression.

The label, if present, references the first word of the block.

The operand expression must be reducible to an absolute value and cannot contain any forward references. (See Section 7.2.2.5 on Operand Symbols).

Example:

```
HOLD    BSS    1
A15     EQU    15B
HLD15   BSS    A15
```

The following is an example of a forward reference. It would produce errors in the assembly.

```
HLD15   BSS    A15   This is a Forward Reference.
A15     EQU    15B   **** ILLEGAL ****
```

7.4.2 DATA

(label) DATA expression1 (, . . . ,expression n)

The DATA statement provides a means of entering constants and data into the program. Each expression in the operand generates a 24-bit binary value. Each expression, except for the last, must be followed by a comma. If the expression contains a relative label, bit 23 may not be used as part of its data.

The label, if present, references the first operand expression.

Example:

```
D10   DATA    12B
MSG3  DATA    5, *+1, 'ERRO', 'R ON', 'TEST', 'ER  ', '1'
ADR   DATA    LABEL + 40000000B  **** ILLEGAL ****
```

7.4.3 TEXT

(label) TEXT 'character string. . . .'

The TEXT directive allows the programmer to define a literal character string which may be output during program execution by a call to the desired output routine.

Typically the character string could be informative comments, actual operating instructions, or error messages for the programmer/operator.

The TEXT statement generates a 24-bit binary value for each four (4) characters in the string. The string must begin and end with single quote marks. Any characters in Table 6-2 except the single quote mark, are allowed. Each character is in 6-bit TRASCII code.

TEXT statement may have from 1 to 64 characters. The character string, including the terminating single quote, must not exceed column 72 of the input record, when entered from the VKT.

Examples:

```
LABEL1  TEXT  'BOTH ALPHA AND NUMERIC CHARACTERS ALLOWED'
LABEL2  TEXT  '1.234 MIXED 9999 ALPHA 24B NUMERICS'
MSG1    TEXT  'ERROR ON STATION 4'
```

7.4.4 ORG

(label) ORG expression

ORG sets the value of the current location counter to the value of the operand which must be completely defined, i.e., it can contain no foreward references.

The label, if present, is assigned the value of the location counter before the counter is assigned its new value. The special expression, ↑* is used to force an even boundary for operands. An 'ORG 0' is assumed if none is given. A relocatable assembly is assumed to be assembled relative to '0'.

Examples:

```
ORG     ↑*
```

This forces an even word boundary.

7.4.5 PZE

```
(label)     PZE     0
```

The PZE "instruction" forces a word of zeroes. It is used wherever a zero word is needed. One major use is as the entry point for subroutines. (See Section 3 for subroutine entry with BSM instruction).

Example:

```
BINASC  PZE  0
```

7.4.6 EQU

```
LABEL     EQU (expression) [,(expression)]
```

EQU is used to assign values to symbols. It does not generate object program code.

The label is assigned the value of the first operand expression; the expression cannot contain any forward references. If second expression is present its value is entered into the symbol table to further define the symbol as an opcode mnemonic. The second expression must be absolute and cannot contain forward references.

EQU directives assigning absolute values to symbols must occur before the symbol is referenced. The EQU directive will produce an 'R' error if this restriction is violated.

The second expression defines the opcode and operand formats according to the following table:

Bits		Meaning
0-2	Operand Type	
	0	User - operand expressions are ORed with opcode value
	1	(address)
	2	(address) [,(index)]
	3	(index)
	4	(index), (address)
	5	(indicator/state),(address)
	6	(state) [. . . . , (state)]
3	0	Not Augmented
	1	Augmented
4-5	Opcode Type	
	1	No Operand Required
	2	Operand Required
6	0	No Indirection allowed
	1	Indirection allowed
8	0	No Memory Reference
	1	Memory Reference

Examples:

```
A     EQU   3
STO   EQU   14000000B, 442B
DO    EQU   ZERO
```

7.4.7 PROC

```
LABEL        PROC        [(expression)]
```

The label is assigned the value of the location counter. In addition, a record is placed in the object program that allows the coreimage create procedure to link CALL directives to the PROC; the CALL statements may be in the current assembly or other ones.

If the operand expression is non-zero, the coreimage create procedure assumes that the PROC statement is an entry point to an interrupt service routine. The expression value is the location of the interrupt entry address and the loader will establish a linkage at the location to the PROC statement. This expression must be absolute and cannot exceed 64 in magnitude. The PROC statement can also be linked to CALL directives when the operand is non zero.

This directive is also used to specify the entry point to the main program in DOPSY. This is done by using the label 'MAINPR'.

Note that the label of a PROC directive is treated like any other label and must be unique or a duplicate label message is issued.

Examples:

```
SIN      PROC
INTI     PROC 1
MAINPR   PROC 0
```

7.4.8 CALL

```
[LABEL]      CALL        (symbol)
```

The label, if present, is assigned the value of the current location counter. A record is placed in the object program that will allow the coreimage create procedure to link the generated BSM to the PROC directive whose label symbol matches the operand symbol of the CALL statement. CALL can be used to link to PROC statements in the same assembly or an independent assembly.

Example:

```
             CALL      SIN
CALLT2       CALL      TEST2
```

7.4.9 END

This statement is a signal to the assembler that the end of the source program has been reached. Source programs must have this statement in order to terminate the assembly if the input is paper tape or keyboard.

Example:

```
QUIT         BRU*   ØVBGEP          Exit from the program.
             END                    End of program to be assembled
```

7.4.10 LIST/NOLIST

These two directives may be used to control which portions of the program will produce an assembly listing.

The mnemonic is placed in the opcode field. There are no label, operand or comment fields.

Statements containing errors will be listed independently of a LIST or NOLIST statement.

7.4.11 OBJ

```
OBJ          expression
```

OBJ allows an operand expression that is used to specify the maximum number of instructions that will be placed in a single object record. This number must be in the range of 1-14 and if unspecified is assumed to be seven (7).

Example:

```
OBJ          12          PUT 12 instructions in each OBJ record
```

7.4.12 PAGE

The assembly and symbol table listings produced by the assembler are formatted on 8 1/2 x 11" pages. The PAGE directive will force a top-of-form; i.e., the PAGE record will be the first line listed on the next page.

The PAGE mnemonic is placed in the opcode field. There are no label, operand, or comment fields.

7.4.13 ABS

This directive indicates that the object program produced is not to be relocated when it is loaded. That is, assembly addresses and execution addresses are the same.

This statement must occur before any object code is produced or the assembly will be produced subject to relocation at load time.

Example: ABS

7.4.14 REL

This statement indicates that the hardware relocatable object program is to be produced. All relocatable memory reference instructions will have bit 23 set to 1 unless ABS is specified. This statement must appear before any instruction.

7.5 ASSEMBLER OUTPUT

The assembler produces the following types of hard copy output.

(1) Assembly listing
(2) Symbol table listing
(3) Object program listing
(4) Cross reference table listing

In this section the formats of these outputs will be described and explained.

7.5.1 Assembly Listing:

The assembly listing has 2 formats:

(1) Lines for comments, PAGE, LIST, NOLIST.
(2) Source statements and some assembler directives.

The source statements have the following format:

EEEE LLLLL VVVVVVVVV		(source statement image)
Where EEEE		is the error code(s)
LLLLL		is the Value of the current location counter
VVVVVVVVV		is the value stored at the location or the value of the operand expression for some directives, (BSS and EQU, for example).

The error can be up to four characters. A description and explanation of these characters is shown in the following table.

Character	Description
D	Duplicate label
L	Label error - label is not an identifier or not terminated by a space.
O	Opcode error - opcode is not an identifier; opcode is not in symbol table.
U	Undefined operand identifier - symbolic index register, state or switch is not defined yet; forward reference is used in multi-term expressions.
S	Syntax error - operand contains illegal operator; magnitude of expression is too large.
R	Relocation error - two relocatable expressions are being added; a relocatable expression is being subtracted from an absolute one, a relocatable expression is being used to reference a hardware element.
N	A number with more than 8 characters
X	Monitor control record
↑	Out of sequence record
V	Data overflow - in the Relocatable Memory Reference Instruction in FST-2 mode uses Bit 23 as data.
A	Address overflow - address exceeds the maximum allowed program size (16K for FST-1, 8K for FST-2)

7.5.2 Symbol Table Listing

The symbol table contains three entries per line. An individual entry has the following format:

```
SYMBOL    C       VVVVVVVVV
```

Where C is one of the following characters:

Character	Meaning
SPACE	
U	The symbol is undefined
N	The symbol is not used
E	The symbol is a label of PROC or the operand of CALL

and VVVVVVVVV is:

(1) The line number in the listing where the symbol is defined/used.
or
(2) The value of the symbol if the label is an EQU directive.

7.5.3 Object Output/File Types

Assembler is capable of generating four types of object files depending on the command and directives. It is the user's responsibility to select proper type according to the usage.

(1) FST-1 object files

Entering the FST-1 parameter in the command specifies this type. Object files produced are identical to FST-1 defined format including instruction set.

It may be an absolute or relocatable program. Instructions LRA and LAR may not be used in this program. The program size can be up to 16K.

(2) FST-2 mode absolute files

ABS directive must precede executable instructions. A special absolute indicator is placed in the object file which directs the CREATE program not to relocate during coreimage create procedure. Since there is no relocation, the user is completely responsible with the memory map during execution.

(3) FST-2 mode relocatable until coreimage create

This is the Assembler's default condition. Object files of this type are similar to FST-1 relocatable files and must go through coreimage create prior to execution. These files are absolute during execution and must stay within the base page. DOPSY subroutines belong to this type.

(4) FST-2 mode hardware relocatable object files

REL directive must precede executable instructions or REL parameter must be provided in the command. Programs must be written relative to zero (do not use ORG directive except ORG 0 in the beginning of the program). ASSEMBLER sets Bit 23 to 1 for all instructions except the following:

- Non-memory reference instructions (e.g., LXA, SST, etc.)
- Instructions with a number in the operand field (e.g., BRU, 100B, ATX X1, 1, etc.)
- Instructions with a variable operand which is EQU-ed to a number (e.g. LAB EQU 245B)
  BRU* LAB

All instructions with Bit 23 'on' will be relocated by the hardware during execution. Programs of this type are the only ones which can be executed anywhere within memory, in the base page or above 16K.

APPENDIX A

FST-2 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 (18 bits)
R-Register (R)	Interrupt Address Director Register (6 bits)
RA	Relocation Adder (18 bits)

RR-Register (RR)	Relocation Register (18 bits)
SN	Nth Output of the AU
SOR	Sum of Relocation Register or 0 and Operand
TOL	Tape Object Loader
TSB-Register (TSB)	Tri-State Buffer Register (24 bits)
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 (18 bits)

APPENDIX B

INSTRUCTION MNEMONICS

B.1 OPCODES SORTED BY ASCENDING ALPHA OPCODES

	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
	064034XX	ARD	ALTERNATE READ	1
	066134XX	ARDS	ALTERNATE READ STATUS	1
	06422400	ASPAC	ALTERNATE SPACE	1
	06401500	ART	ALTERNATE READ RECORD TAPE	1
*	11000000	ATX	ADD TO INDEX	2
	07000000	AUG	AUGMENT	
	06423400	AWRIT	ALTERNATE WRITE	1
*	00000000	BAH	BRANCH AFTER 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	
*	02200000	BZ	BRANCH IF ZERO	1
	12000000	CALL	SUBROUTINE CALL	
*	23000000	CAM	COMPARE A WITH MEMORY	2

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

	OPCODE	MNEMONIC	CODE DESCRIPTION	CYCLES
	07000604	CLA	CLEAR ACCUMULATOR	1
*	30000000	DADD	DOUBLE ADD	3
*		DATA	DATA DEFINITION	
*	35000000	DIV	DIVIDE	26
*	31000000	DLD	DOUBLE LOAD	3
	07034000	DSA	DOUBLE SHIFT AROUND	
	07036000	DSL	DOUBLE SHIFT LEFT	
	07016000	DSN	DOUBLE SHIFT NORMALIZED	
	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	
	060100XX	ETST	ERROR TEST	1
	07010000	EXC	EXCHANGE A AND E	1
	06051500	FSKIPB	SKIP FILE BACKWARD (GO BACK TO TAPE MARK)	1
	06041500	FSKIPF	SKIP FILE FORWARD (ADVANCE TO TAPE MARK)	1
	07012400	IDA	INTERRUPT DISABLE	1
	07004400	IEN	INTERRUPT ENABLE	1
	07000600	LAR	LOAD A FROM RELOCATION REGISTER	1
	07000200	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	1
		LIST	PRODUCE ASSEMBLY LISTING	
	07000400	LRA	LOAD RELOCATION REGISTER FROM A	1
	07022000	LS	LOGICAL SHIFT A	
	07000000	LXA	LOAD INDEX FROM A	1
*	34000000	MUL	MULTIPLY	25
		NOLIST	NO ASSEMBLY LISTING	
*	10000000	NOP	NO OPERATION	1
		OBJ	PRODUCE OBJECT PROGRAM	
*	27000000	OR	OR (INCLUSIVE)	2
		ORG	ORIGINATION CONTROL	
		PAGE	PAGINATION CONTROL	
	060010XX	PCOMP	PRIORITY COMPLETE	1
	060110XX	POFF	PRIORITY OFF (INTERRUPT DISABLE)	1
	060130XX	PON	PRIORITY ON (INTERRUPT ENABLE)	1
	00000000	PROC	SUBROUTINE ENTRY POINT	
	00000000	PZE	POSITIVE ZERO (ENTRY PT)	
	064014XX	RD	READ	1
	066114XX	RDS	READ STATUS	1

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

	OPCODE	MNEMONIC	CODE DESCRIPTION	CYCLES
	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 (DETAILED SPU COMMANDS ARE LISTED IN APPENDIX D)	1
	07020000	SR	SHIFT A RIGHT	
	07004000	SST	SET STATE	1
*	14000000	STA	STORE A-REGISTER	2
*	15000000	STE	STORE E-REGISTER	2
	07000611	STM1	SET FST-1 MODE	1
	07000612	STM2	SET FST-2 MODE	1
	060000XX	STST	STATUS TEST	1
*	16000000	STX	STORE INDEX	2
*	22000000	SUB	SUBTRACT	2
	07002000	TCA	TWO's COMPLEMENT A	1
	06000400	TOF	TOP-OF-FORM	1
	064214XX	WRIT	WRITE	1
	06061500	WRITM	WRITE TAPE MARK	1

* B23=0 for Absolute Memory Reference of Non-REL Program
=1 for Relocatable Memory Reference with REL Program

SORTED BY ASCENDING OCTAL OPCODE

	OPCODE	MNEMONIC	CODE DESCRIPTION	CYCLES
		ABS	ABSOLUTE PROGRAM LOCATOR	
	00000000	BSS	BLOCK STORAGE SIZE	
*	00000000	DATA	DATA DEFINITION	
	00000000	END	PROGRAM TERMINATOR	
		EQU	EQUIVALENCE	

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

	OPCODE	MNEMONIC	CODE DESCRIPTION	CYCLES
		LIST	PRODUCE ASSEMBLY LISTING	
		NOLIST	NO ASSEMBLY LISTING	
		OBJ	SPECIFY OBJECT PROGRAM SIZE	
		ORG	ORIGINATION CONTROL	
		PAGE	PAGINATION CONTROL	
*	00000000	BAH	BRANCH AFTER 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 (DETAILED SPU COMMANDS ARE LISTED IN APPENDIX D)	1
	060000XX	STST	STATUS TEST	1
	06000400	TOF	TOP-OF-FORM	1
	06000500	REWIND	REWIND TAPE	1
	060010XX	PCOMP	PRIORITY COMPLETE	1
	06001500	RSKIPF	SKIP RECORD FORWARD	1
	060100XX	ETST	ERROR TEST	1
	060110XX	POFF	PRIORITY OFF (INTERRUPT DISABLE)	1
	06011500	RSKIPB	SKIP RECORD BACKWARD	1
	060130XX	PON	PRIORITY ON (INTERRUPT ENABLE)	1
	06020400	FEED	CHARACTER (PAPER TAPE) FEED	1
	06041500	FSKIPF	SKIP FILE FORWARD (ADVANCE TO TAPE MARK)	1

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

	OPCODE	MNEMONIC	CODE DESCRIPTION	CYCLES
	06051500	FSKIPB	SKIP FILE BACKWARD (GO BACK TO TAPE MARK)	1
	06061500	WRITM	WRITE TAPE MARK	1
	064014XX	RD	READ	1
	06401500	ART	ALTERNATE READ RECORD TAPE	1
	064034XX	ARD	ALTERNATE READ	1
	06420400	SPAC	SPACE	1
	064214XX	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
	066114XX	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
	07000200	LAX	LOAD A FROM INDEX	1
	07000400	LRA	LOAD RELOCATION REGISTER FROM A	1
	07000600	LAR	LOAD A FROM RELOCATION REGISTER	1
	07000604	CLA	CLEAR ACCUMULATOR	1
	07000611	STM1	SET FST-1 MODE	1
	07000612	STM2	SET FST-2 MODE	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
		CALL	SUBROUTINE CALL	

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

	OPCODE	MNEMONIC	CODE DESCRIPTION	CYCLES
*	14000000	STA	STORE A-REGISTER	2
*	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	SUBRACT ONE FROM MEMORY	4

* Bit 23=0 for Absolute Reference of non-REL Program
=1 for Relocation Memory Reference with REL Directive

APPENDIX C

FST-2 OPCODES SORTED ON NUMBER

OF OPERANDS REQUIRED

Operand type

0 — No operand required

DTC	IDA	RSR	LAR	CLA
EXC	IEN	TCA	LRA	

1 — One address operand expression required

BAT Typ		BOI Typ	
BN	BO	BBC	BL
BNEZ	BP	BE	BLE
BNZ	BPZ	BG	BNE
BZ		BGE	

2 — Optional double operand expressions. The first expression must be an address. The second expression, if present, must refer to an index register.

ADD	DLD	LDS	STA
AND	DSA	LS	STE
AOM	DSL	MUL	SUB
AUG	DSN	NOP	
BAH	DSR	OR	
BRU	DST	RUM	
BSM	DSUB	SA	
CAM	EOR	SL	
DADD	LDA	SOM	
DIV	LDE	SR	

3 — Operand is a single expression which is an index register.

LAX
LXA

FST-2 OPCODES SORTED ON NUMBER OF OPERANDS REQUIRED (Continued)

4 — Operand is a double expression which consists of an index register and a value.

ATX - The value can be a constant or an address.
LDX - The value can be a constant or an address.
STX - The value must be an address.

5 — Operand is a double expression which consists of an indicator or state switch and an address

BAT
BOI
BOS

6 — Operand may have multiple expressions which are state switches.

RST
SST

** High speed com link has priority of 10 (9746016/9746017)

* → 488 & com link interrupt addresses are switch selectable!

→ Normal com link interrupt address is 31-D (9742016/9742016) " " " 31-B
High speed com link " " " this address (9746016/9746017) is not switch selectable!!

→ 488 interrupts not used by O.S. 4/22/80 normal would be 40-B

APPENDIX D

FST-2 SPU INSTRUCTION SET

		TTY Keybd (TTK)	TTY Printer (TTP)	Card Reader (CR)	Line Printer (LP)	Disc (fixed)	Mag Tape	Tester	Data Set	488 Bus
Device Code (octal)		020	030	040	060	070	10X	120	13X	14X
Interrupt Priority		1	2	5	3	8	7	9	11 **	6
Interrupt Address (octal)		02	03	04	06	07	10	12-21	30-37*	40-43*
Memory Priority		--	--	5	--	12	8	1	--	--
Status bit return (except ETST):										
20 = BE: not available		x	x	x	x	x	x	VCCT		
21 = LT: busy		x	x	x	x	x	x	DMA		
22 = EQ: idle with error		x	x	x		x	x	IREQ		
23 = GT: idle, no error		x	x	x	x	x	x	TBUSY		
Instruction Set	06...									
STST Status test	0000XX	x	x	x	x	x	x	x	x	x
ETST Error test	0100XX			x		x	x			
RDS Read status	6114XX	x	x	x		x	x			x
PON Enable interrupt	0130XX	x	x	x	x	x	x	x	x	x
POFF Disable interrupt	0110XX	x	x	x	x	x	x	x	x	x
PCOMP Interrupt complete	0010XX	x	x	x	x	x	x	x	x	x
Feed paper tape	020420	x								
Read keyboard	601420	x								
Print	421430		x							
Read card binary	401440			x						
Read card BCD	403440			x						
Write	421460				x					
Read disc	401470					x				
Alternate read disc	403470					x				
Read alt. disc status (TASA)	613470					x				
Write disc	421470					x				
Read record	50150X						x			
Alternate read record	40150X						x			
Advance 1 record	00150X						x			
Go back 1 record	01150X						x			
Advance to TM	04150X						x			
Go back to TM	05150X						x			
Write record	42150X						x			
Skip & write record	46150X						x			
Write TM	06150X						x			
Rewind	00050X						x			
Read excess word count	61170X						x			
Read	6XX120							x		
Write	4XX120							x		
Special	2XX120							x		
Noop	0XX120							x		

FST-2 SPU INSTRUCTION SET (Continued)

		TTY Keybd (TTK)	TTY Printer (TTP)	Card Reader (CR)	Line Printer (LP)	Disc (fixed)	Mag Tape	Tester	Data Set	488 Bus
Instruction Set (Continued)	06...									
Write data	50013X								X	
Read data	70013X								X	
Write control fcn.	44013X								X	
Latch UART status	01213X								X	
Read UART status	64013X								X	
Read RS232 status	60413X								X	
Reset interface	01013X								X	
Control fcn. load	02013X								X	
Read sequence reg.	60053X								X	
Spare	02113X								X	
SRE Send remote enable	00214X									X
RRE Reset remote enable	00314X									X
GTS Go to standby	00414X									X
SIC Send interface clear	00514X									X
RPP Request parallel poll	00614X									X
TCON take control	01014X									X
RDD Read data	60154X									X
WRIT Write data	42154X									X
BCOM Bus command	43154X									X
ARM Interrupt	400170	X	X	X		X	X	X		

VKT1 VKT2 Flop not used!

20/30 21/31 22/32 23/33 device address

1/2 1/2 1/2 1/2 interrupt priority

2/3 52/53 32/33 42/43 interrupt address

APPENDIX E

FST-2 HARDWARE MNEMONIC SIGNAL CODE DESCRIPTIONS

Mnemonic Signal	Description
ACC/	Memory accessed by Peripheral, Pause CPU
ACS	Peripheral has memory access
ACOLT/	A column address select
ADA00-23	A memory data bus
AMB00/-23/	Peripheral A memory bus
AMEM/	A memory enable
AMSOC/	A memory start of cycle
AMSWC/	A memory start write cycle
AMW/	A memory write
AOMTEX1/	TEX1 cycle of AOM instruction
ARAS/	A row address select
AREF/	A refresh
AZERO	Accumulator equals zero
A00-23	Accumulator register bits 00-23
A2223	Clock A bit 22 into A bit 23
A2323	Clock A bit 23 into A bit 23
BCOLT	B column address select
BDA00-23	B memory data bus
BE	Bit equal status
BEI	BE lamp drive

FST-2 HARDWARE MNEMONIC SIGNAL CODE DESCRIPTIONS (Continued)

Mnemonic Signal	Description
BEQ/	Bit equal bus
BMB00/-23/	Peripheral B memory bus
BMEM/	B memory enable
BMSOC/	B memory start of cycle
BMSW/	B memory start write cycle
BMW/	B memory write
BN00/-23/	Accumulator bus bits 00-23
BRAS/	B row address select
BREF/	B refresh
B00-23	Buffer register bits 00-23
CAO01-23 odd	Carry out from arithmetic unit
CI	Carry in ALU 1st bit EQ1 to forms two's comp.
COEQZ	Iteration counter = zero
COEQ1	Iteration counter = one
COLEQ5	Iteration counter LT or EQ five
CO00-05	Iteration counter bits 00-05
CPS/	System clock
CPX/-Z/	System clock
CP1/-4/	System clock
CRBOOT	Card reader boot
CRLU	Command register lock up
CS0-5	Console switches 0-5
CY	Carry flip-flop

FST-2 HARDWARE MNEMONIC SIGNAL CODE DESCRIPTIONS (Continued)

Mnemonic Signal	Description
CYE/	Carry to E0 (Divide)
DBSST	Disc busy status
DCO	Decrement iteration counter
DIS00I-23I	Display bits 00-23
DSERR	Disc has some error
DSPLR/	Display relocation register
EB	Enable Branch
EQ	Equal status
EQI	Equal lamp drive
EXCLK	External clock enable
E00-23	Extension register bits 00-23
FST-1 LT/	FST-1 mode lamp drive (not used)
FST1	FST1 mode
F2LAX/	Load A from X in FST-2 mode
GAE	Gate A to E
GAFUA	Gate A register Comp. to "A" input of ALU
GAN/	Gate A to Accumulator bus
GBM/	Gate B to memory
GBUBXX/	Gating to "B" input of arithmetic unit
GCPA/	Gate clock pulse to A
GCPB/	Gate clock pulse to B
GCPC/	Gate clock pulse to C
GCPE/	Gate clock pulse to E
GCPI/	Gate clock pulse to indirect bit

FST-2 HARDWARE MNEMONIC SIGNAL CODE DESCRIPTIONS (Continued)

Mnemonic Signal	Description
GCPXA	Gate clock pulse to index bits
GCP09/	Gate clock pulse to operand 0-9
GCP1013/	Gate clock pulse to operand 10-13
GCRN/	Gate command register to BN bus
GDIS0/-2/	Gate display to front panel
GEFUA	Gate E register comp. to A input of ALU
GMCR/	Gate memory to command register
GMO1417/	Gate memory to ALT operand bits 14-17
GM1UA/	Gate minus 1 to "A" input of ALU
GM1UAA/	Gate minus 1 to "A" input of ALU
GOA1417/	Gate clock to ALT operand
GOCO	Gate operand field (bits 0-5) to iteration counter
GOM/	Gate operand to memory
GOM1/	Gate operand minus 1 to memory
GOP	Gate operand to P counter
GOPIM	Adds one to operand address (double precision)
GO23/	Gate clock pulse to relocation bit (operand bit 23)
GPM	Gate P to memory
GRO	Gate interrupt register (R) to operand (bits 0-5)
GSA	Gate sum to A register
GSB	Gate sum to B register
GSE	Gate sum to E register

FST-2 HARDWARE MNEMONIC SIGNAL CODE DESCRIPTIONS (Continued)

Mnemonic Signal	Description
GT	Greater than status
GTI	GT lamp drive
GUAXX/	Gating to "A" input of arithmetic unit
GWA	Gate switch register (W) to A register
GWCR/	Gate switch register to command register
GWP	Gate switch register to P counter
GXA	Gate index to A register
GXUB/	Gate index to "B" input of arithmetic unit
G1-7	Gating signals 1-7
G10-19	Gating signals 10-19
G23CO	Sets iteration counter to 27B
IAT	Load interrupt address to R register
IC/	Inhibit carry
IDLE	Idle, CPU stopped
IDX/	Index cycle
IE	Interrupt enable
IEI	IE lamp drive
INA	Indirect cycle
INAFIDXF	No indirect or index cycle
INAOIDX	Indirect or index cycle
IP	Interrupt priority time
IPC	Increment P counter
IPC08, 16, 24	Carry from P counter stages
ISTOP	Immediate stop

FST-2 HARDWARE MNEMONIC SIGNAL CODE DESCRIPTIONS (Continued)

Mnemonic Signal	Description
IX0-2	Index register selection for display
KCRLU	Command register lockup switch
KEXAM	Examine switch
KFML	Manual load switch (not used)
KLDA	Load A switch
KLDCR	Load command register switch
KLDP	Load P counter switch
KLGOM/	Disable gating of operand to memory
KRST	Reset switch
KSTRT	Start switch
KSTW	Store switch
LDATX/	Load index or add to index
LT	Less than status
LTI	LT lamp drive
MABZ/	A memory busy (not used)
MBBZ/	B memory busy (not used)
MCLR/	Master clear
ML/	Manual load
MLDA/	Manual load A
MLDCRA/	Manual load command register
MTBOOT	Mag tape boot
MTBUSY	Mag tape busy
MTSERR	Mag tape has some error
M00-23	M bus bits 0-23

FST-2 HARDWARE MNEMONIC SIGNAL CODE DESCRIPTIONS (Continued)

Mnemonic Signal	Description
OA14 - 17	ALT operand bits 14-17
OEQ0/	Operand field (bits 0-5) equals zero
OVF1	Overflow status FST-1 mode
OVF2	Overflow status FST-2 mode
OVI	OV lamp drive
O00-23	Operand register bits 0-23
PAERR	Parity error A
PAI08,16	Parity A carry between stages
PAI24	Parity A into parity generator from memory
PAO24	Parity A out of parity generator to memory
PB08, 16	Parity B carry between stages
PBI24	Parity B into parity generator from memory
PBO24	Parity B out of parity generator to memory
PDSW/	Parity disable switch
PS	Peripheral select time
PW00-23	P counter or switch register bus 0-23
P00, 08, 16	P counter carry between stages
RCPS1/-3/	System clock - inactive for CPU pause or halt except during manual load
RCRY08, 16, 14	Relocation adder carry between stages
REQ0/	Interrupt register equals zero
RESET	Reset
RUN	CPU is not halted
RX15/	Disable clock to index bits 16 & 17

FST-2 HARDWARE MNEMONIC SIGNAL CODE DESCRIPTIONS (Continued)

Mnemonic Signal	Description
R00-05	Interrupt register bits 0-5
SAL	Shift A left
SAUG/	Subaugmented instruction
SEL	Shift E left
SELR/	Select relocation register
SER	Shift E right
SFST2/	Set FST-2 mode
SIC	Single instruction cycle
SMC	Single memory cycle
SM0 1	Select data to M bus
SOR00, 08, 16	Relocation adder outputs 0, 8, 16
SPRHLT	Spare halt
SPRHLT1	Spare halt 1
STOPM/	Stop memory cycles
STOP	Stop CPU
STWB/	Store switch register to buffer
STWM/	Store switch register to memory
ST9	State 9 (overflow)
SW0I-7I	State switch lamp drivers 0-7
S01/-23/ odd	Sum of ALU bits 0-23
TEX1 I	TEX1 lamp drive
TEX1	Time of execution 1
TEX2I	TEX2 lamp drive
TEX2	Time of execution 2

FST-2 HARDWARE MNEMONIC SIGNAL CODE DESCRIPTIONS (Continued)

Mnemonic Signal	Description
TIF	Time of instruction fetch
TIFI	TIF lamp drive
TMB/	Transfer memory to buffer register
TOF	Time of operand fetch
TOFI	TOF lamp drive
TV	Time for variable
TVI	TV lamp drive
T1-5	CPU phase time 1-5
UA23/	"A" input to ALU bit 23
UB23/	"B" input to ALU bit 23
WR	Write relocation register
WRM	Write to memory
WX/	Write index register
W00-23	Switch register bits 0-23
XSUB/	Gate ones to index bits 14-17 (negation FST- 2 only)
X15A/-17A	Effective index address
ZERO	ALU sum equals zero

INDEX

Feedback Form

Please send us your comments. They will help us produce better publications for you. We are interested in comments on accuracy, completeness, organization, clarity, audience suitability, illustrations, and detail level. If you choose to identify yourself (lower left), we will acknowledge your comments. All comments are confidential, FTSG property.

Optional

Name _____

Title _____

Address _____

Phone _____

Telex _____

Publications Department Use Only

P/N 57000002

Rev _____ Date November 1978

Technical Reviewer _____ Date _____ Response Sent _____ Date _____

FAIRCHILD Test Systems Group

634700-1

STAPLE

STAPLE

No Postage
Necessary If
Mailed In The
United States

Business Reply Mail
First Class Permit No. 5699 San Jose, California

Postage Will Be Paid By

Fairchild Test Systems Group
Technical Publications
1725 Technology Drive
San Jose, Ca. 95110