Reference Manual
SEL 840MP General Purpose Computer System

September, 1968

This publication supersedes the Reference Manual, SEL 840MP Multiprocessor Computer System, dated June, 1968, numbered SEL 95098A.

301-095098-001
Price: $5.00

6901 West Sunrise Blvd., Fort Lauderdale, Florida 33310
Area Code 305 587-2900

LIST OF EFFECTIVE PAGES

The total number of pages is 140, as follows:

Page Number	Issue	Page Number	Issue
Title	Original		
A	Original		
i thru viii	Original		
1-1 thru 1-16	Original		
2-1 thru 2-34	Original		
3-1 thru 3-12	Original		
4-1 thru 4-12	Original		
5-1 thru 5-6	Original		
6-1 thru 6-24	Original		
7-1 thru 7-2	Original		
A-1	Original		
B-1 thru B-4	Original		
C-1 thru C-2	Original		
D-1 thru D-2	Original		
E-1 thru E-2	Original		
F-1 thru F-10	Original		
G-1 thru G-2	Original		

TABLE OF CONTENTS

Section	Title	Page
I	GENERAL DESCRIPTION	
	Introduction	1-1
	System Configurations	1-1
	System Characteristics	1-1
	840MP Computer	1-1
	Shared Memory	1-1
	Shared Memory Input/Output Processor	1-1
	Characteristics Summary	1-3
	840MP Computer Characteristics	1-3
	Computer Options	1-3
	Shared Memory System	1-3
	Shared Memory Input/Output Processor	1-3
	Peripheral Devices	1-3
	Standard Software	1-4
	Multiprocessor Organization	1-4
	840MP Computer	1-4
	Private Memory	1-4
	Central Processor	1-6
	Central Processor Input/Output Processor	1-9
	Extended Arithmetic Unit	1-10
	Shared Memory	1-10
	Shared Memory Input/Output Processor	1-12
	SEL 840MP Software System	1-12
	General	1-12
	SEL 840MP Assembly Program	1-14
	SEL 840MP Loader Program	1-15
	SEL 840MP FORTRAN IV Compiler	1-15
	SEL 840MP Debug Program	1-15
	SEL 840MP Update Program	1-15
	SEL 840MP Library Package	1-15
	SEL 840MP Maintenance Routines	1-16
II	MACHINE LANGUAGE PROGRAMMING	
	Introduction	2-1
	Memory Reference Instructions	2-1
	Augmented Instructions	2-4
	Input/Output Instructions	2-5
	Machine Language Instruction Set	2-5
	Load/Store Instructions	2-5
	Arithmetic Instructions	2-9
	Branch/Skip Instructions	2-12
	Logical Instructions	2-15
	Register Change Instructions	2-16
	Shift Instructions	2-17
	Control Instructions	2-20
	Input/Output Instructions	2-22
	Wait Mode	2-22
	Skip Mode	2-22
	EAU Instructions	2-27
	Program Protect Instructions	2-32

TABLE OF CONTENTS (Cont'd)

Section	Title	Page
III	ASSEMBLY LANGUAGE PROGRAMMING	
	General Description	3-1
	Location Field	3-1
	Operation Field	3-1
	Address Field (Variable Field)	3-1
	Operand Address Field Formats	3-1
	No Address	3-1
	Symbolic Address	3-1
	External Symbolic Address	3-1
	Absolute Address	3-1
	Current Location	3-1
	Address Arithmetic	3-3
	Literal Address	3-3
	Location to be Filled	3-3
	Comments Field	3-3
	Identification Field	3-3
	Mnemonic Computer Instructions	3-3
	Pseudo-Operation Instructions	3-8
IV	INPUT/OUTPUT	
	Introduction	4-1
	Central Processor-Input/Output Processor (CP-I/OP)	4-2
	Shared Memory-Input/Output Processor (SM-I/OP)	4-5
	General	4-5
	Shared Memory Input/Output Processor Initialization	4-6
	SM-I/OP Data Transfer	4-8
	Block Transfer Control Unit	4-8
	General	4-8
	BTC Operation	4-9
	BTC Initialization and Data Flow	4-10
	BTC Priority and Timing	4-11
V	PRIORITY INTERRUPT SYSTEM	
	General Description	5-1
	Interrupt Routine Processing	5-1
	Interrupt Connection	5-3
	Interrupt Enabling/Disabling	5-3
	Interrupt Level Logic	5-4
	Interrupt Routine Programming	5-4
VI	PERIPHERAL DEVICES	
	General	6-1
	Console Input/Output Station (Device No. 1)	6-1
	Operation and Programming	6-1
	Paper Tape Reader-Model 84-510A (Device No. 2)	6-3
	Paper Tape Punch-Model 84-520A (Device No. 2)	6-4
	Paper Tape Spooler-Model 80-530A	6-4
	High-Speed Paper Tape System-Model 84-525A (Device No. 2)	6-5
	Operation and Programming	6-5
	Card Reader-Model 84-450A-400 CPM (Device No. 4)	6-5
	Operation and Programming	6-5
	High-Speed Line Printer-Model 80-730 Series (Device No. 5)	6-6
	Operation and Programming	6-6

TABLE OF CONTENTS (Cont'd)

Section	Title	Page
	Magnetic Tape System-Model No. 84-600 Series (Device Nos. 6 and 7)	6-9
	Operation and Programming	6-11
	Magnetic Tape System Commands	6-11
	Magnetic Tape System Tests	6-12
	Use of the Block Transfer Control	6-12
	Use of Word Transfer (No BTC)	6-15
	Use of Interrupts	6-15
	Programming Example	6-16
	X-Y Incremental Plotters-Models 84-810 and 84-812 (Device No. 11)	6-16
	Operation and Programming	6-16
	Movable Head Disc Storage-Model 84-653A (Device No. 13)	6-19
	Operation and Programming	6-20
	Fixed Head Disc Storage-Model No. 84-654A (Device No. 13)	6-21
	Operation and Programming	6-22
VII	OPTIONS	
	General	7-1
	Program Protect and Instruction Trap (84-080MP)	7-1
	Operation and Programming	7-1
	Stall Alarm (84-042MP)	7-2
	Auto Start (84-041MP)	7-2
	20 Sense Switches (84-021MP)	7-2
	Real Time Clock (84-031MP)	7-2
	Input/Output Parity (84-210MP)	7-2
	APPENDIX A - SEL 840MP Computer Word Format	A-1
	APPENDIX B - SEL Peripheral Device Octal Character Codes	B-1
	APPENDIX C - 840MP Computer CEU Word Format	C-1
	APPENDIX D - 840MP Computer TAC Instruction Word Format	D-1
	APPENDIX E - SEL 840 Paper Tape Formats	E-1
	APPENDIX F - Numerical Information	F-1
	APPENDIX G - SEL 840MP Computer Instruction List Summary	G-1

LIST OF ILLUSTRATIONS

Figure	Title	Page
1-1	840 Multiprocessor Computer - System Block Diagram	1-2
1-2	SEL 810MP Computer Block Diagram	1-5
1-3	840MP Program Counter and Bank Address Register Bit Configurations	1-6
1-4	840MP Computer Basic Data Formats	1-8
1-5	EAU Data Formats	1-11
1-6	Block Diagram - Extended Arithmetic Unit	1-12
1-7	Block Diagram - Typical 840MP Shared Memory System	1-13
1-8	Block Diagram - Shared Memory Input/Output Processor	1-14
2-1	Word Format - 840MP Computer Memory Reference Instructions	2-2
2-2	Word Format - 840MP Computer Indirect Address Word	2-2
2-3	Sample Memory Reference Instruction (LAA)	2-3
2-4	Word Format - 840MP Augmented Instruction	2-4
2-5	Sample 008 and 218 Augmented Instructions	2-6
2-6	Sample Input/Output Instruction	2-7

LIST OF ILLUSTRATIONS (Cont'd)

Figure	Title	Page
2-7	AIP/AOP Flow Chart	2-23
2-8	I/O Instruction Word Format	2-24
3-1	Assembly Coding Sample	3-2
4-1	Connection of Peripheral Devices to the 840MP Computer	4-2
4-2	Connections of Peripheral Devices to Shared Memory Input/Output Processor	4-3
4-3	Input/Output Configuration and Computer Interface	4-4
4-4	Peripheral Connections - Simplified	4-5
4-5	Shared Memory Input/Output Processor Functional Block Diagram	4-7
4-6	Shared Memory I/OP, CEU, and TEU Word Formats	4-8
5-1	Word Format - Interrupt Routine Entry Location	5-3
5-2	Word Formats - PIE and PID Instructions	5-4
5-3	Sample Assembly Language Interrupt Processing Routine	5-5
5-4	Sample Assembly Language Interrupt Entry and Exit Routine	5-5
6-1	Paper Tape Data Flow Diagram	6-3
6-2	Sample Program for Console Input/Output Station	6-4
6-3	Sample Program for Copying Paper Tape Using High-Speed Paper Tape System	6-6
6-4	Sample Program for Card Reader	6-7
6-5	Sample Program for Line Printer	6-10
6-6	Magnetic Tape Format 0 Data Word	6-12
6-7	Sample Program for Magnetic Tape System	6-16
6-8	Sample Program for X-Y Plotter	6-18
6-9	Track and Sector Layout	6-19
6-10	Movable Head Arrangement - Recording Surface	6-19
6-11	Head Position	6-19
6-12	Typical Head Positioning Time	6-20
6-13	Fixed Head Track and Sector Layout	6-21
6-14	Fixed Head Arrangement - Recording Surface	6-21
7-1	Program Protect Sample Program	7-2

LIST OF TABLES

Table	Title	Page
2-1	Sample Iterative Subprogram	2-4
2-2	I/O Instruction Bit Definitions	2-24
3-1	Sample Address Field Entries	3-3
3-2	SEL 840MP Mnemonic Instructions	3-4
3-3	SEL 840MP Absolute Notation Formats	3-7
3-4	Summary of SEL 840MP Pseudo-Operations	3-10
4-1	I/O Control Signals	4-5
4-2	Input/Output Processor Dedicated Memory Locations (Shared Memory)	4-6
4-3	Central Processor BTC Dedicated Memory Locations (Private Memory)	4-9
5-1	Optional Interrupt Levels	5-1
5-2	Interrupt Level Memory Location Assignments	5-2
5-3	Standard Interrupt CEU Bit Functions	5-3
6-1	Console I/O Station Specification	6-1
6-2	Console Input/Output Station CEU Function Bid Coding	6-2
6-3	Paper Tape Reader Specifications	6-3
6-4	Paper Tape Punch Specifications	6-4
6-5	Paper Tape Spooler Specifications	6-4
6-6	Card Reader Specifications	6-5

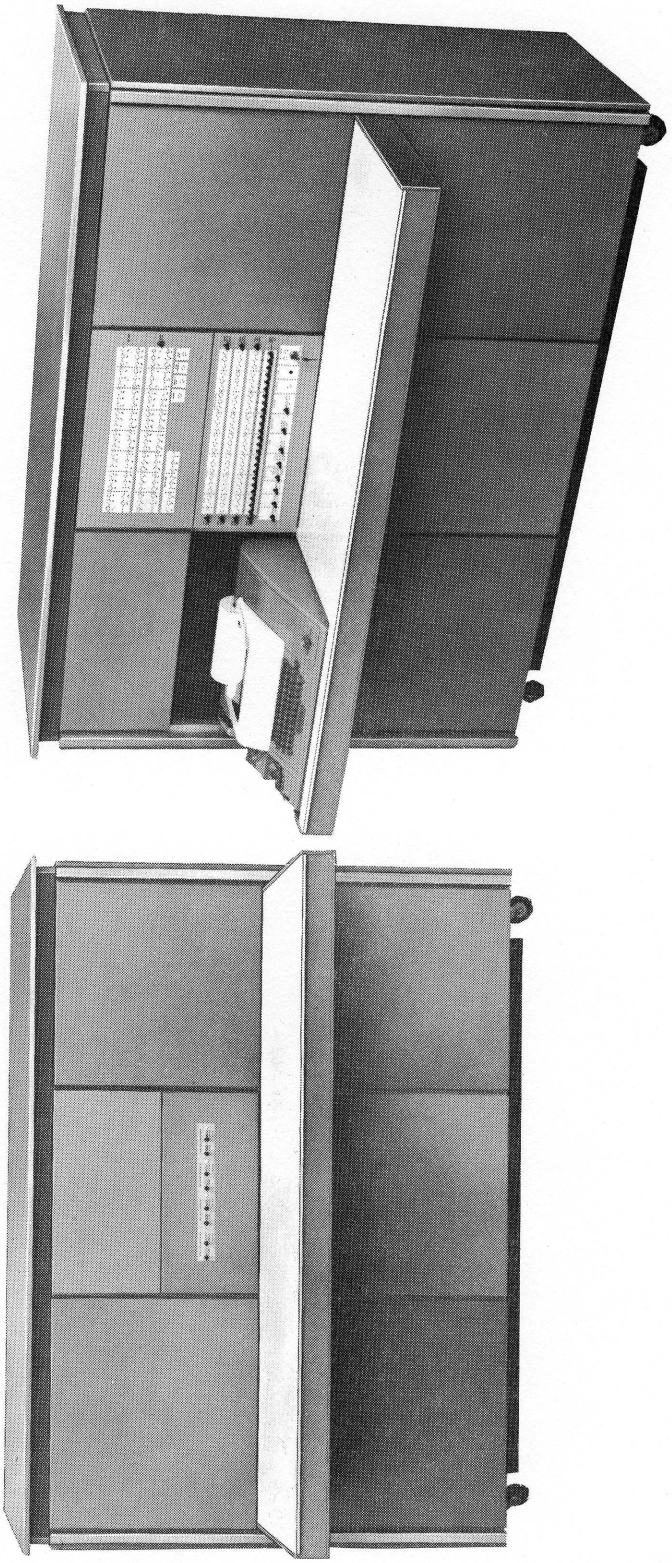
LIST OF TABLES (Cont'd)

Table	Title	Page
6-7	Line Printer Specifications	6-6
6-8	Line Printer Command and Test Functions	6-8
6-9	Magnetic Tape Transports Specifications	6-9
6-10	Magnetic Tape Commands, Format 0	6-11
6-11	Magnetic Tape Commands, Format 1	6-13
6-12	Magnetic Tape Test Functions	6-14
6-13	X-Y Plotter Specifications	6-17
6-14	X-Y Incremental Plotter Commands	6-17
6-15	Fixed Head Disc Storage System Specifications	6-22

LIST OF RELATED PUBLICATIONS

The following publications contain information not included in this manual but necessary for a complete understanding of the SEL 840MP General Purpose Computer System.

Publication Title	Publication No.
SEL 840MP Computer Technical Manual	95017
SEL 840MP I/O Interface Design Manual	95097
SEL 840A/840MP Operators Manual	95051
SEL 800 Computer Series Fortran IV Reference Manual	95131
SEL 800 Computer Series Fortran Compiler Maintenance Manual	95062
SEL 840MP Computer Maintenance Drawings Manual	95095
SEL 840A/840MP Assembler Reference Manual	95053
SEL 840A/840MP Assembler Maintenance Manual	95063
SEL 840A/840MP Loader Maintenance Manual	95056
SEL 800 Computer Series Library Maintenance Manual	95057
SEL 840A/840MP Program Documentation	



P-2575

SEL 840 Multiprocessor Computer System

SECTION I
GENERAL DESCRIPTION

INTRODUCTION

The SEL 840 Multiprocessor Computer System shown in the frontispiece extends the capability of Systems Engineering Laboratories' medium-scale computers into the domain of large-scale computer systems. The two primary features of the system are a large memory capacity and simultaneous multiprocessing. Configurations are available containing from one to four 840MP Computers, each having up to 32,768 private memory locations and capable of directly addressing an additional 65,536 locations in a Shared Memory System. Shared Memory Input/Output Processors (SM-I/OP's) are available to relieve the high-speed computers of time consuming input/output operations. True simultaneous multiprocessing may be obtained between all processors used in the system. Specifically, up to six processors can access the shared memory during the same 1.75 microsecond machine cycle, provided that each processor is accessing from a different 8 K memory module.

Included among the standard features of the SEL 840MP Computer are: a highly flexible I/O structure capable of handling up to 64 devices; three hardware index registers; and a comprehensive software package.

A large number of standard peripheral devices and computer options are available with SEL 840 Multiprocessor Computer Systems. The expandable capabilities make the 840 Multiprocessor Computer Systems ideally suited to application areas such as the real-time simulation of large systems (that is, aircraft and missiles) and time sharing between on-line system control and off-line processing.

SYSTEM CONFIGURATIONS

SEL 840 Multiprocessor Computer Systems are available in a large variety of configurations. A block diagram of a typical 840 Multiprocessor Computer System is shown in figure 1-1. This typical system consists of one 840MP Computer, one Shared Memory Input/Output Processor (SM-I/OP) and a Shared Memory. The Shared Memory may contain from one to eight modules of 8,192 (8 K) locations each. Each 840MP Computer communicates with its private memory containing from 8 K to 32 K memory locations. Shared Memory is equipped with a maximum of six I/O ports. Communications with all shared memory modules is provided through each port. Each 840MP Computer and SM-I/OP requires a single port. Therefore, the following maximum 840MP Computer and SM-I/OP combinations may be employed in an SEL 840 Multiprocessor Computer System.

a. One SEL 840MP Computer, four SM-I/OP's, and one Shared Memory

b. Two SEL 840MP Computers, four SM-I/OP's, and one Shared Memory

c. Three SEL 840MP Computers, three SM-I/OP's, and one Shared Memory

d. Four SEL 840MP Computers, two SM-I/OP's, and one Shared Memory

SYSTEM CHARACTERISTICS

840MP COMPUTER

The 840MP Computer is a fast, general-purpose, 24-bit binary computer. Included among the 840MP Computer standard features are hardware multiply and divide, three hardware index registers, memory parity check, three independent levels of priority interrupt, a private I/O structure capable of addressing 64 computer peripheral devices or peripheral device controllers, and a comprehensive software package.

SHARED MEMORY

The 840MP Multiprocessor Computer System Shared Memory provides 8,192 to 65,536 locations capable of being independently accessed by either the 840MP Computer or the SM-I/OP. Shared Memory may be expanded from the minimum 8,192 locations by the addition of memory modules containing 8,192 (8 K) locations each. A maximum of eight modules may be supplied with the Shared Memory system.

SHARED MEMORY INPUT/OUTPUT PROCESSOR

The Shared Memory Input/Output Processor provides the interface between the Shared Memory and external peripheral devices. A maximum

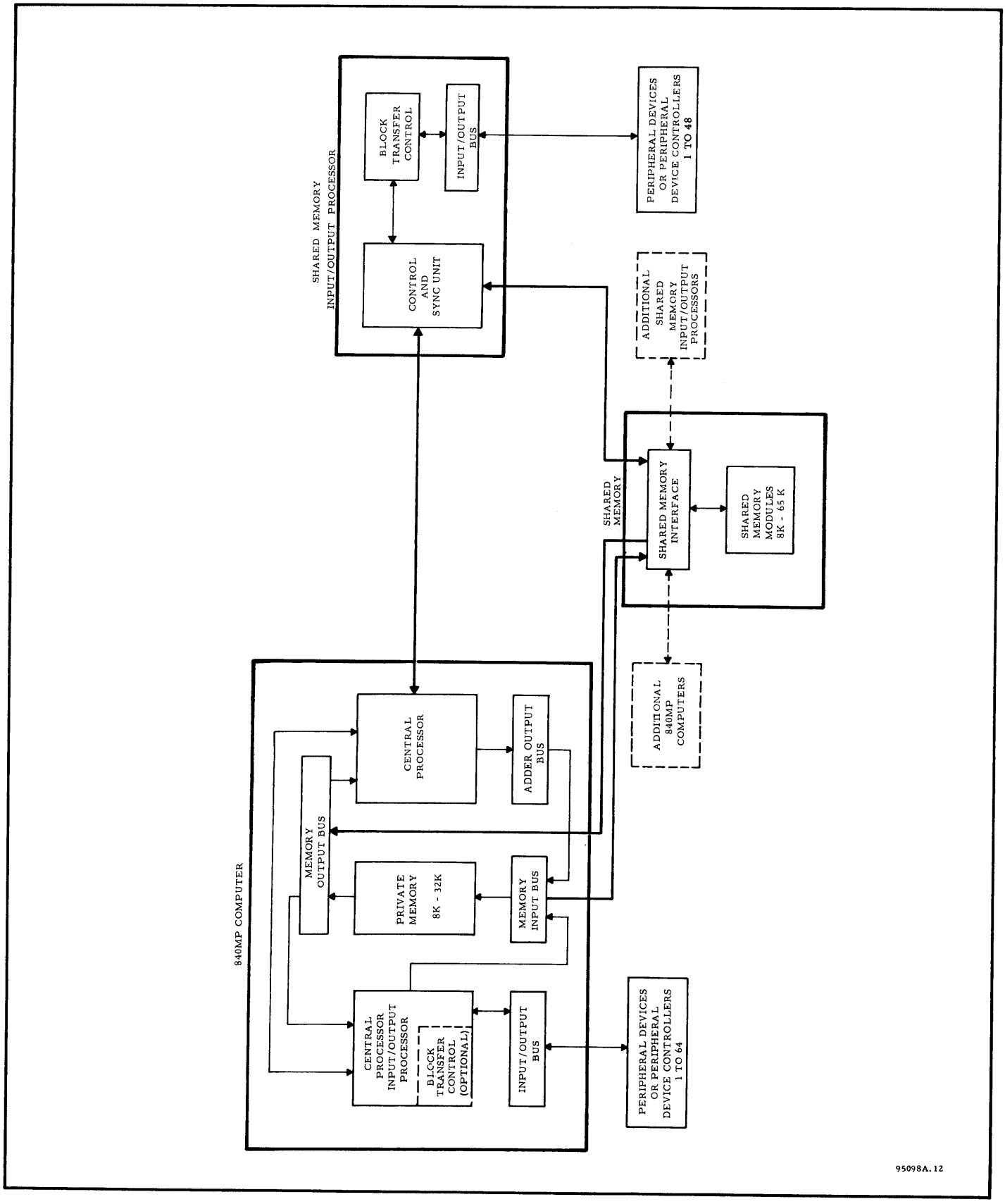


Figure 1-1. 840 Multiprocessor Computer - System Block Diagram

of four SM-I/OP's may be provided with an SEL 840 Multiprocessor Computer System. Up to 48 peripheral devices or device controllers may be connected to each SM-I/OP. Each peripheral device operating with an SM-I/OP utilizes a BTC unit. One BTC unit is standard with each SM-I/OP; two additional BTC ACGP units may be provided to expand the capabilities of each SM-I/OP.

CHARACTERISTICS SUMMARY

The following listing is a summary of the 840 Multiprocessor Computer System's characteristics:

840MP COMPUTER CHARACTERISTICS

All silicon monolithic integrated logic circuits

24-bit word length; plus parity and program protect bits

8,192 word private memory

Address capability - 98,304 locations (32,768 in private memory and 65,536 in Shared Memory)

1.75 microsecond, full-cycle time

Computation time (including accessing and indexing):

Add, subtract	3.5 microseconds
Multiply	10.5 microseconds
Divide	26.25 microseconds

Three hardware index registers

I/O structure for handling 16 peripheral devices (may be expanded to handle 64 devices)

Three independent levels of priority interrupts

Power fail-safe

Four sense switches

Switch-addressable program halt

Integrally mounted ASR-33 typewriter with paper tape reader and punch

Size - 72 inches wide, 51 inches high, 27 inches deep (43 inches deep with desk top)

Temperature environment - 50° to 95°F

COMPUTER OPTIONS

Private memory expandable to 32,768 locations by 8,192 word modules.

Program protect system for guarding individual locations (includes instruction trap which prevents unauthorized execution of privileged instructions)

Up to 58 additional levels of priority interrupts

High speed extended arithmetic unit (EAU) - providing hardware floating-point and double-precision, fixed-point computational capability

Stall alarm

I/O parity

20 additional sense switches

Real-time clock

Auto start

Up to six Block Transfer Control (BTC) units

Up to two Computer Graphics Processor (CGP) units

SHARED MEMORY SYSTEM

8,192 locations - expandable to 65,536 locations with the use of 8 K modules

Capable of being accessed by six processors (available in one, four, and six port configurations)

Size - 72 inches wide, 51 inches high, 27 inches deep (43 inches with desk top)

Temperature environment - 50° to 95°F

SHARED MEMORY INPUT/OUTPUT PROCESSOR

Mounted in shared memory cabinet; a maximum of four I/OP's may be included in any one configuration.

One BTC unit standard - two additional BTC units optionally available

Capable of communicating with up to 48 peripheral devices

PERIPHERAL DEVICES

Card reader - 400 cards/minute

Card punch - 100 cards/minute

Paper tape reader (photoelectric) - 300 characters/second

Paper tape punch - 110 characters/second

Magnetic tape control unit - handles up to eight tape units

Magnetic tape units - 45, 120, 150 inches/second; 200, 556, 800 characters/inch; 7 track and 800 characters/inch; 9 track

Moveable head disc file - 1,024,000 24-bit word storage, 150 millisecond maximum access time

Fixed head disc file - 75,000 to 606,000, 24-bit word storage; 8.3 millisecond average access time

Typewriters - ASR-33, KSR-33, ASR-35, KSR-35, RO-33 and RO-35, 10 characters/second

Line printer - 300, 600, 1000 lines/minute, 120 columns/line

Incremental plotters - 12-inch chart width (300 steps/second) and 31-inch chart width (200 steps/second)

CRT display - 10x10 inch display area includes vector generator

Options: Character generator
Light pen

Interval timer

Interface subsystem components

Multiplexers - low-level and high-level, solid state and relay switching

Analog/digital and digital/analog converters

Decimal displays - in-line data readout

Sample-and-hold units

Custom interface units

STANDARD SOFTWARE

Full ASA FORTRAN IV compiler

FORTRAN library

Mnemonic assembly program (ASSEMBLER) - one or two pass, relocatable object format

Compiler/assembler loader

Real-time monitor

Utility routines - debugging aids, I/O handlers, tape editor

Maintenance routines - complete set for SEL 840MP Computer and peripheral devices

MULTIPROCESSOR ORGANIZATION

840MP COMPUTER

The 840MP Computer (figure 1-2) is formed by three major units - private memory (PM), central processor (CP) and central processor input/output processor (CP-I/OP). The PM unit stores the instruction words which define the operation of the CP and the data words on which the CP operates. The CP's control unit calls up and decodes the instruction words and issues commands to operate the CP. The CP's standard arithmetic unit performs computation with data words supplied by memory and the CP-I/OP under direction of the CP's control unit. The CP-I/OP controls the transfer of data words, commands and status reports between the CP and its private peripheral devices. The CP operates on, and from, 24-bit binary words which are transferred in parallel between the CP units. Arithmetic operations are performed using two's complement binary arithmetic with negative quantities stored in two's complement form.

PRIVATE MEMORY

The 840MP Computer private memory is composed of one to four separate modules. Each module contains 8,192 (8K) addressable locations. The total number of locations may range from 8,192 up to 32,768 (four 8K modules). Each location can store one 24-bit data or instruction word plus a parity bit, and a program protect bit (if the program protect option is included in the system).

Individual modules are composed of four elements:

a. 8K magnetic core memory (25 or 26 bits)

b. 13 bit (8K) memory address register

c. 25 or 26 bit data register

d. Self-contained timing and control

Instruction words and data words are loaded into specific addresses prior to the program execution. Loading may be performed manually through the

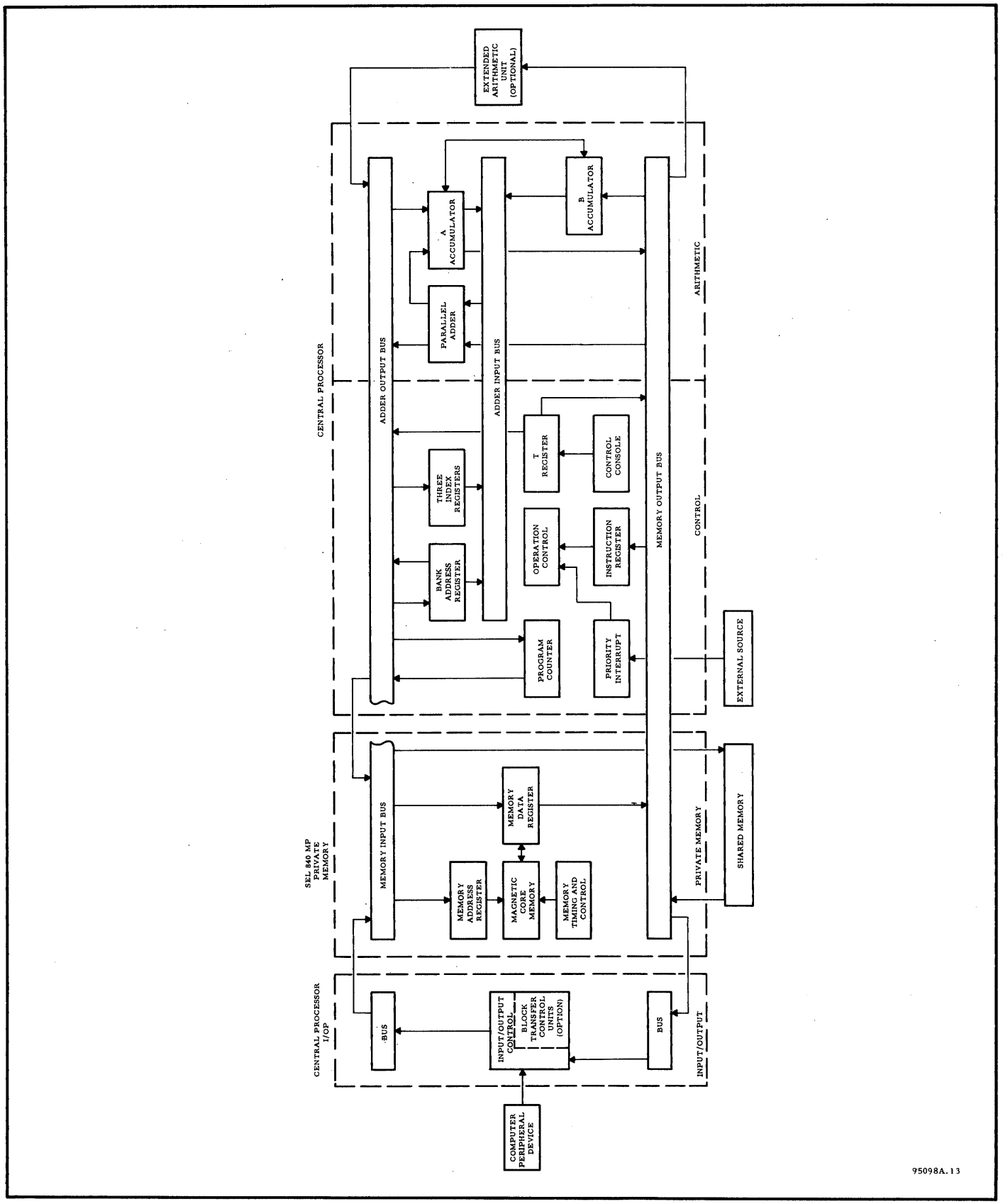


Figure 1-2. SEL 840MP Computer Block Diagram

computer's panel controls or automatically from peripheral devices through the use of the supplied loader program. Each input word is transferred to the memory data register and the accompanying storage address is transferred to the memory address register. When both registers have been loaded, a write command is issued by the program control unit and the 25 bits in the memory data register are written into the 25 magnetic cores addressed by the memory address register.

When the entire group of instruction words forming a program is loaded and execution is started, addresses selected by the CP's control unit are sent to the memory address register and a read command is issued. The state of each core at that address is sensed and transferred to the memory output register. The sensing of the cores sets them all to the same state, so the memory word now in the memory data register is immediately rewritten into its original memory location so as to be available for later use. The word is also transferred to the CP's control unit to be decoded or to the CP's arithmetic unit for computation. The memory read and write cycles are completely automatic; only the memory address and source or destination must be supplied by the program.

CENTRAL PROCESSOR

The Central Processor, often referred to as the mainframe, comprises a control unit and an arithmetic unit which are described in the following paragraphs.

Control Unit

The control unit contains a 15-bit binary program counter which, in conjunction with the bank address register (BAR), allows the maximum of 98,304 memory locations to be directly addressed by the CP. Figure 1-3 illustrates the bit configurations of the program counter and the BAR.

The two most significant bits of the program counter (memory bits 9 and 10) are used to designate one of four previously defined memory banks. The four-bit address of each defined memory bank is stored in one of the four BAR's (BAR 0-3). Each time an instruction is accessed from memory, the two most significant bits of the instruction address FF9-FF10 of the program counter) are used to select one of the four BAR's. The four bits in the selected BAR are appended to the most significant end of the remaining 13 bits of the operand address (FF11-FF23 of the program counter) to create a 17-bit effective address. This 17-bit address is capable of referencing any of the maximum of 98,304 locations in private or shared memory.

Initially, the program counter is set to the address of the first instruction in the program. The program counter is automatically incremented by one when each instruction is executed except when a halt, branch or conditional skip instruction is unloaded from memory. The halt instruction stops the execution of the program while a branch instruction changes the contents of the program counter to the operand address specified by the instruction. A conditional skip instruction causes

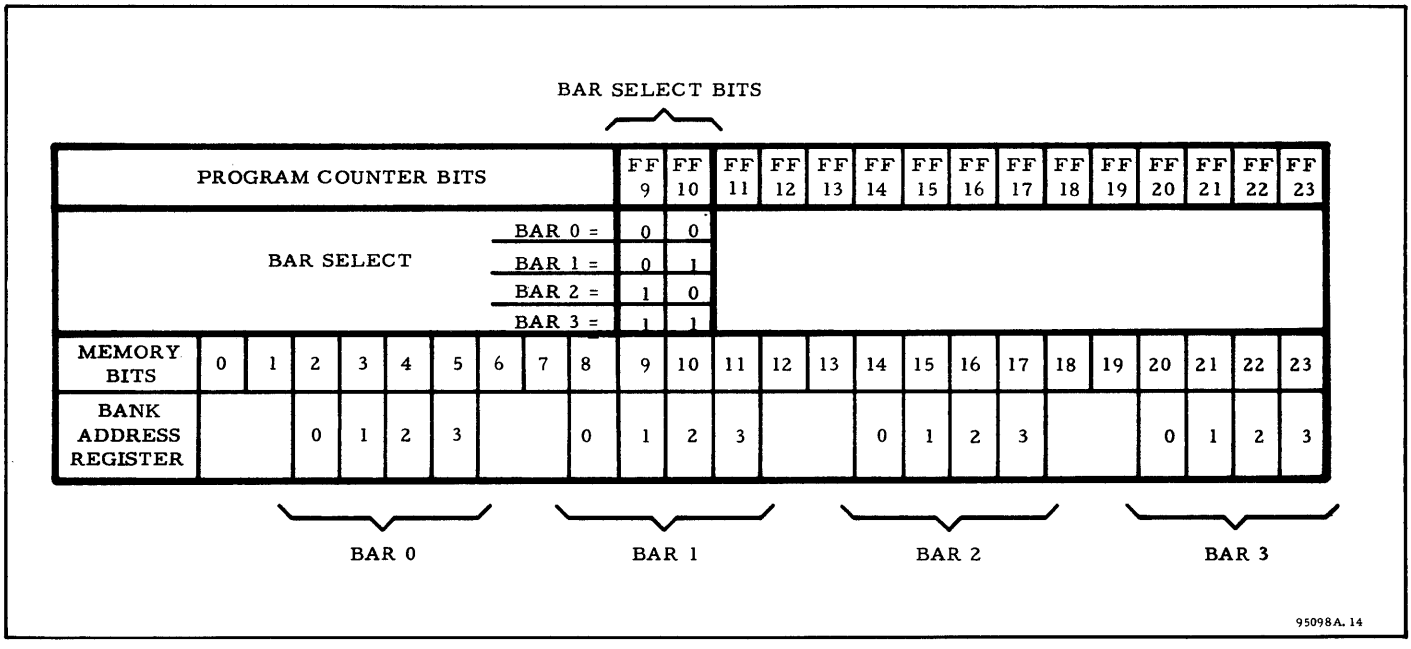


Figure 1-3. 840MP Program Counter and Bank Address Register Bit Configurations

the program counter to be advanced by one or more locations, depending on the value of the condition specified by the instruction.

The instruction words are read from memory into the instruction register and automatically restored in memory. The binary digits forming the instruction word are then applied to the operation control circuits. The unique codes assigned to each instruction in the SEL 840MP instruction repertoire are then decoded and used to provide timing and gating signals to the other units in the CP. Signals generated by control panel operations are also fed to the operation control circuits. External priority interrupts will cause the operation control circuits to alter the program execution sequence in order to process the external demand, and to return to the point at which the normal sequence was interrupted.

The memory cycle during which instruction words are read and decoded is designated as the instruction cycle. Memory reference instructions specify the location of an operand which is to be operated on by the processor. For this instruction, one or more additional memory cycles - execution cycle- are required.

Most memory reference instructions are accessed and executed in a total of two memory cycles. However, instructions such as multiply and divide require more than one execution cycle. The operand address contained in the instruction word may be modified during the instruction cycle by adding the contents of one of the three index registers to the address. Multilevel indirect addressing may be performed as part of the execution of all memory referencing instructions. Both of these forms of address modification are performed prior to the performance of the address mapping function in which the two most significant bits of the 15-bit effective virtual address (EVA) are used to access the contents of one BAR to form the 17-bit effective address (EA). The EA consists of the contents of the specified BAR in the four most significant bit positions and the 13-low-order effective virtual address bits.

Many instruction words require no operand from memory and are executed completely within the instruction cycle. Others, while requiring no operand from memory, do require one or more execution cycles for completion. Chief among this latter group are the shift instructions. For these instructions, a group of bits within the instruction word defines the number of shifts to be performed while the operation code of the word defines the type of shifting to be done. Other instructions, notably the I/O control instructions, are composed of two instruction words; one defining the type of operation and specifying the device and the other defining the actual operand or the operand memory location. The words defining these I/O instructions are automatically accessed from memory in the proper sequence.

Arithmetic Unit

The arithmetic unit consists of a 24-bit parallel adder and two accessory storage registers (A- and B-Accumulators). Both the A- and B-Accumulators may be loaded and unloaded under program control. The A-Accumulator is the primary arithmetic register and derives its name from its function of accumulating results of arithmetic operations. Since only one word may be taken from memory and/or I/O units by each instruction, the second operand in addition and subtraction operations must be loaded in a register prior to the execution of the add and subtract instructions. The A-Accumulator provides the required storage for the operand and temporary storage for the result of the arithmetic operation. The B-Accumulator holds the multiplier during multiply operations, and stores the least significant bits of the product. In addition to these arithmetic functions, the two accumulators provide a convenient storage area for rearranging data words through shifting and logical operations. A third register connected to the adder is the T-Register which holds the operand accessed from memory. This 24-bit register plus the 24-bit A- and B-Accumulators supply inputs to the 24-bit binary adder. When an add instruction is performed, the data words are simply added according to the rules of two's complement binary arithmetic.

The basic data format of the 840MP Computer is a 24-bit binary, single-precision, fixed-point word (see figure 1-4). This format contains the sign bit in bit position 0 with bit positions 1 and 23 holding the most significant and least significant data bits, respectively. This format is defined as an integer, with the assumed binary point located to the right of bit position 23. The SEL 840MP library subroutines assume this representation. The programmer may scale single-precision words in any desired manner and utilize the test and shift instructions to maintain the binary point location.

The 840MP Computer also accommodates double-precision data words (figure 1-4B) of 46 bits, plus sign, through the use of the B-Accumulator. Each double-precision data word is normally stored in two adjacent memory locations with the most significant half stored in the lower address. The product of a single-precision multiply operation is located in the A- and B- Accumulators in this format. Prior to the execution of a divide

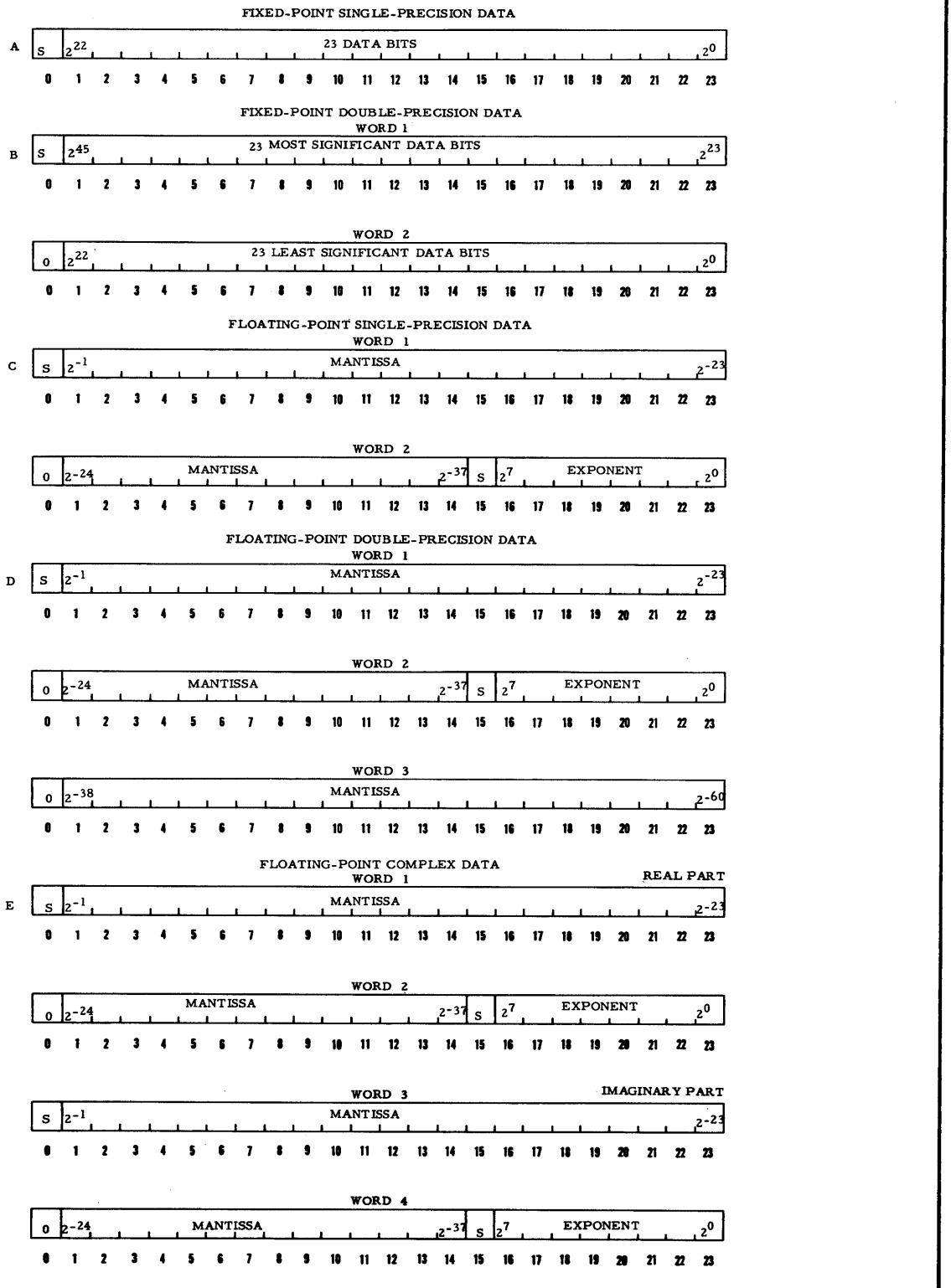


Figure 1-4. 840MP Computer Basic Data Formats

instruction, the dividend must be in the double-precision format.

Three floating point data formats are utilized by the 840MP Computer library. The single-precision floating point format (figure 1-4C) consists of two words. The first word contains the sign and 23 most significant bits of the fractional mantissa; the second word contains the 14 least significant mantissa bits and the signed 8-bit exponent. The words are stored in adjacent memory locations with the first word located in the lower memory address. Both the mantissa and the exponents carry separate signs so that the mantissa may be positive or negative independent of the sign of the exponent. A double-precision floating point format (figure 1-4D) consisting of three memory words is provided for use with the set of double-precision floating point library subroutines. The third floating point data format (figure 1-4E) is provided for the set of FORTRAN IV subroutines dealing with complex numbers.

The arithmetic unit includes two control latches which are addressable by the program. The first of these is the overflow latch which can be set during addition, subtraction, and division operations. The overflow for an add or subtract occurs when the result exceeds the accumulator capacity of ±8,388,607. A divide overflow occurs if the divisor is zero or if the divisor is smaller than the dividend. This latter overflow is due to the fact that the machine treats all divide arguments as double-precision numbers by scaling the single-precision divisor by 2^{23}. If the dividend is larger than the scaled divisor, the quotient will necessarily be a number greater than 2^{23}. Since a number exceeds the capacity of the 24-bit A-Accumulator in which the quotient is to be stored and thus produces a false result. The set overflow latch lights the OVERFLOW indicator on the control console and remains set until tested, and reset by a BOF (branch on overflow) instruction. Because the latch remains set until tested, such a test should be made immediately following an arithmetic process when an overflow condition could result. This prevents the possibility of a second overflow being undetected by the already set latch.

The second addressable arithmetic latch is the carry latch which is connected to the least significant bit of the parallel adder. This latch is set during the regular arithmetic operations to produce a two's complement number (one's complement of the number plus one). The latch is also used under program control in the addition and subtraction of double-precision numbers formed in the A- and B-Accumulators. The least significant words of the double-precision numbers are processed and stored in the B-Accumulator. If a carry or borrow is generated, it will cause the sign of the B-Accumulator to change. A CSB (copy sign of B) instruction is used to set the carry latch to the state of the B-Accumulator sign bit and then reset to the B sign bit to zero (as required in the double-precision format). If the operation is addition, the true output of the carry latch is added to the most significant words. If a subtract operation is in process, the false output of the carry latch is added to the most significant words (effectively subtracting the borrow). Because the carry latch is set and reset as a part of various arithmetic and logical instructions (subtract, negate, divide, etc.), care must be taken that the use of the CSB instruction with this latch does not interfere with subsequent instructions.

CENTRAL PROCESSOR INPUT/OUTPUT PROCESSOR

The Central Processor Input/Output Processor (CP-I/OP) is capable of communicating with up to 64 peripheral devices or peripheral device controllers. These device controllers, such as the magnetic tape controller, may each communicate with several peripheral devices.

Data transfer instructions are provided which enable word transfers directly between computer memory and peripheral devices as well as between the A-Accumulator and peripheral devices. In addition, external unit command and test instructions are provided. The I/O instruction set is particularly powerful because each instruction causes several functions to be performed. First, execution of each I/O instruction causes a device to be connected to the computer. The device (unit) number is contained in each I/O instruction. Second, an automatic test is made of the unit which determines if the device can execute the instruction. Third, the data or command transfer is made if the device is ready. Fourth, the device is disconnected. If the device is not ready when tested, the computer will either wait until the device is ready and then transfer or it will disconnect the device and advance the program counter to a reject location. A wait flag is provided in each I/O instruction except the test instruction (TEU) to enable the programmer to specify the wait or skip mode of execution. The normal time required to perform the complete connect, test, transfer, and disconnect operation is only three machine cycles.

In addition to the basic I/O structure, up to six BTC units can be added to the 840MP Computer. These units permit the transfer of blocks of data between either private or shared memory and peripheral units at rates up to 572,000 words per second. One machine cycle is stolen from the

840MP Computer for each word transferred. Transfers are made under hardware control with block length varying from a single word to 98,304 words. An automatic re-initialization feature allows chaining of blocks. Also up to two computer Graphics Processors (CGP) may be added to the 840MP Computer. The CGP is similar to the BTC with the exception of its specialized operating characteristics and added control functions. Unlike the BTC, the CGP examines each word from memory and either interprets the word as data or as an instruction.

A priority interrupt system is provided, enabling the computer to have up to 61 individual levels of priority interrupts. The 59 programmable interrupts can be selectively enabled and disabled under program control. A unique memory location is assigned to each interrupt level.

An ASR-33 typewriter, with paper tape reader and punch, is supplied with the 840MP Computer. This complete unit is referred to as the Computer Console I/O Station or Console I/O Station. The console I/O station can be operated either off-line or on-line, the input (reader and keyboard) and output (printer and punch) operate independently. This allows, for example, a paper tape to be read and a separate set of characters to be printed at the same time. The paper tape reader is under complete control of the program when operating on-line; the reader is started and stopped by a command from the computer. Operating speed is 20 characters per second for the reader and 10 characters per second for the keyboard, printer, and punch.

EXTENDED ARITHMETIC UNIT

The optional extended arithmetic unit (EAU) offers a means of rapidly and easily performing the double-precision and floating-point arithmetic associated with scientific computation. A distinct group of 23 instructions is provided for programming the EAU. The results of EAU operations may be tested for sign, zero, and overflow conditions under program control.

Any EAU operation initiated by the CP control unit proceeds independently of the 840MP Computer until that operation is completed. During this time, no further EAU operations may be initiated. This active condition may be tested by a special instruction, extended skip if ready (ESR). The EAU instruction repertoire includes several load/store instructions which transfer words between memory and the EAU.

The basic EAU data format is shown in figure 1-5. As illustrated in the figure, the EAU manipulates 24-bit words from memory to construct fixed-point double-precision and floating-point quantities. Memory words 1, 2, 3 and 4 are called from, or loaded into, sequential memory locations. Only the initial address need be specified in the program; the second memory word will be accessed automatically. When transferring double-precision data between memory and the EAU, two load/store operations must be performed.

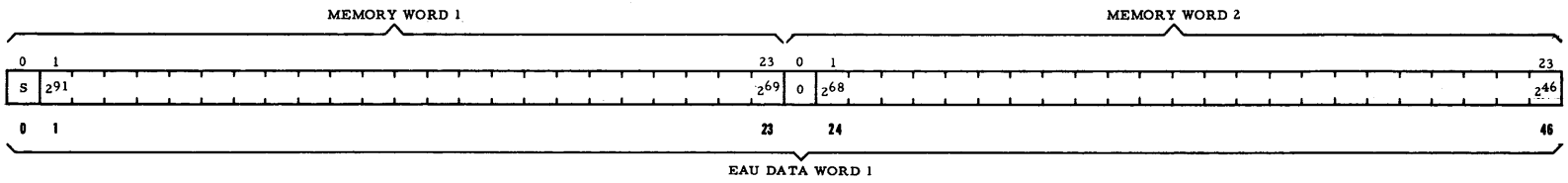
A simplified block diagram of the EAU is shown in figure 1-6. The EAU contains two 47-bit accumulators (EA and EB) and a 47-bit parallel adder to accommodate double-precision, fixed-point data and the mantissa of floating-point data. The floating-point exponents are processed in a special nine-bit register and a nine-bit parallel adder. EAU arithmetic operations utilize the same two's complement binary arithmetic used in the 840MP Computer. The EAU operates in parallel with the arithmetic unit. This capability finds its greatest value in high-speed, real-time applications of the 840 Multiprocessor Computer System. In these applications, the 840MP Computer's arithmetic register and the I/OP's can be used to manipulate I/O data words while the real-time calculations are performed by the EAU.

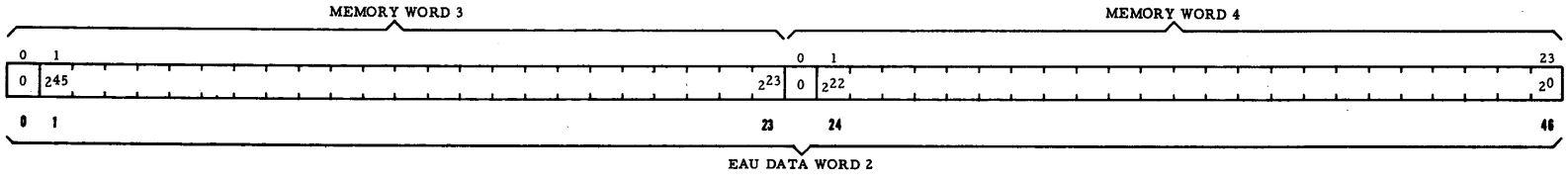
The EAU is equipped with an indicator panel which permits the display of the contents of the major EAU register, and indicates the status of various operating conditions.

SHARED MEMORY

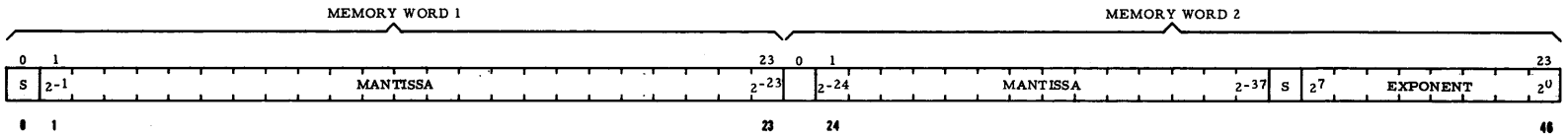
A block diagram of a typical 840MP Shared Memory system is shown in figure 1-7. The Shared Memory system is composed of from one to eight memory modules and control logic. Each memory module contains 8,192 (8K) addressable locations. Up to 65,536 addressable locations may be provided in the shared memory system. Each location can store one 24-bit data or instruction word plus a parity bit. This 25-bit capacity is increased to 26 bits with the program protect option. Each memory module consists of the four elements listed in the private memory discussion.

The organization and the operation of the Shared Memory system are similar to the Private Memory unit in the 840MP Computer. Instruction and data words are loaded into specific Shared Memory addresses prior to execution of the program. Loading may be performed manually through the 840MP Computer Control Panel or automatically from the computer or peripheral devices through the use of the loader program supplied with the system. Instruction and data words are stored (written) or unloaded (read) in the same manner as in the computer's private memory.





A. FIXED-POINT DOUBLE-PRECISION DATA



B. FLOATING-POINT DATA

95098A-17

Figure 1-5. EAU Data Formats

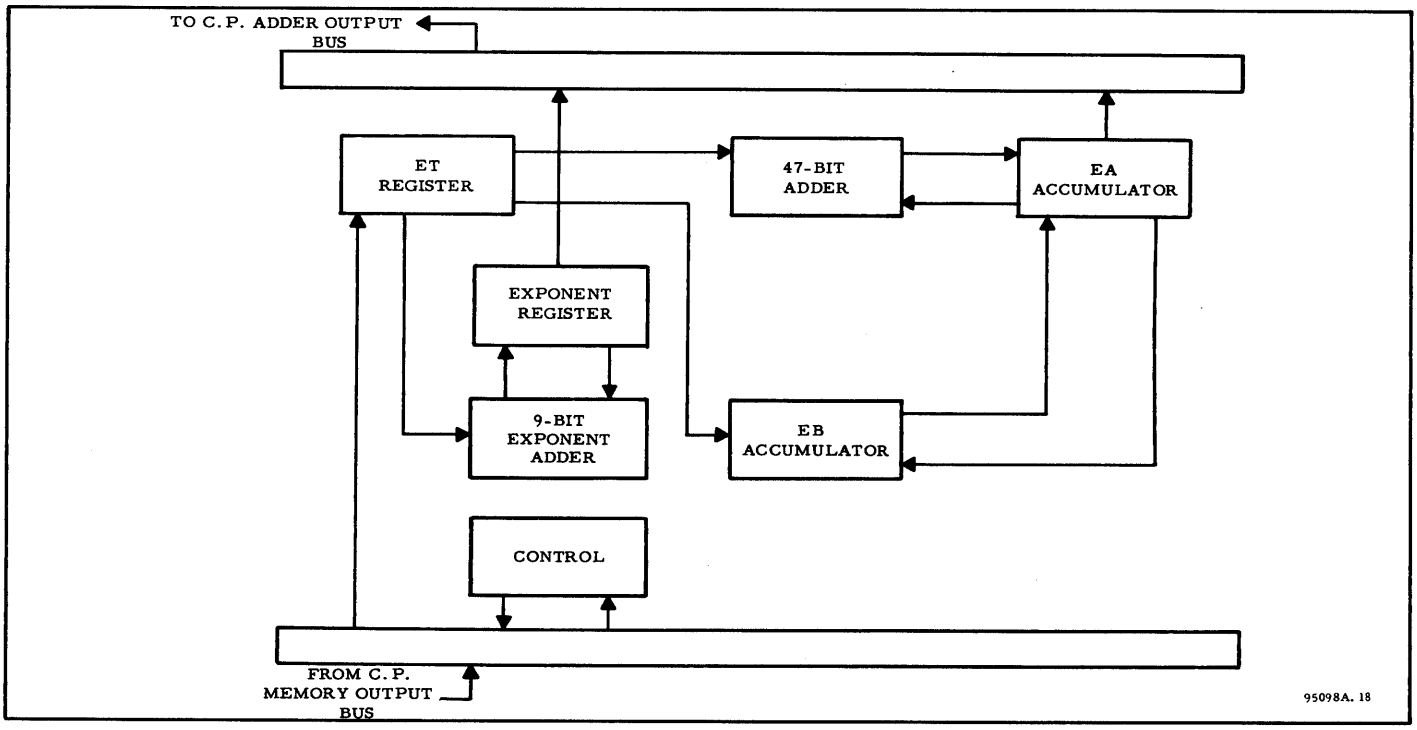


Figure 1-6. Block Diagram - Extended Arithmetic Unit

The Shared Memory may be used by up to six processors (CP's or SM-I/OP's). Since every CP and SM-I/OP can access any module in shared memory, it is necessary to have a priority system that will be activated whenever two or more processors (CP or SM-I/OP) attempt simultaneous access of the same memory module. Under such a system, each 840MP Computer and SM-I/OP has a priority number. When two processors attempt to enter a memory module at the same time, the priority number determines which entry takes precedence. The Shared Memory control logic includes priority logic as well as input and output control logic for the selection of processors and memory modules when transferring information between SM-I/OP's and Shared Memory.

SHARED MEMORY INPUT/OUTPUT PROCESSOR

A block diagram of the Shared Memory Input/Output Processor (SM-I/OP) is shown in figure 1-8. The SM-I/OP provides communication between Shared Memory and external peripheral devices while freeing the 840MP Computer(s) from time-consuming I/O operations. Once initialized, the SM-I/OP is completely independent of the SEL 840MP Computer. The SM-I/OP is initialized by the execution of a Test And Activate (TAC) instruction in the SEL 840MP Computer. This instruction causes the SM-I/OP to unload a word from a fixed location in shared memory and to use this word as a command or test for one of the peripheral units connected to the SM-I/OP. All peripheral devices connected to the SM-I/OP use a BTC unit. Up to three BTC units can be provided with each SM-I/OP. The BTC units can be set to handle data block lengths from one word to 64K words. The operation of the BTC and block priority control (BPC) is discussed in Section IV of this manual.

The I/OP contains anti-coincidence and priority logic to deal with simultaneous commands from two or more 840MP Computers. The lower priority 840MP Computer must wait until the higher priority operation is completed. Another 840MP Computer's interrupt will not be serviced until six cycles after completing the servicing of another 840MP Computer's previous interrupt.

The six cycle margin prevents the interruption of lower priority successive operations (once initiated) by a higher priority 840MP Computer's command until the lower priority operations are completed.

SEL 840 MP SOFTWARE SYSTEM

GENERAL

A comprehensive, fully-integrated, well-documented and completely checked-out program preparation, library, debugging, and utility system is supplied with the 840 Multiprocessor Computer System. Two basic types of program preparation

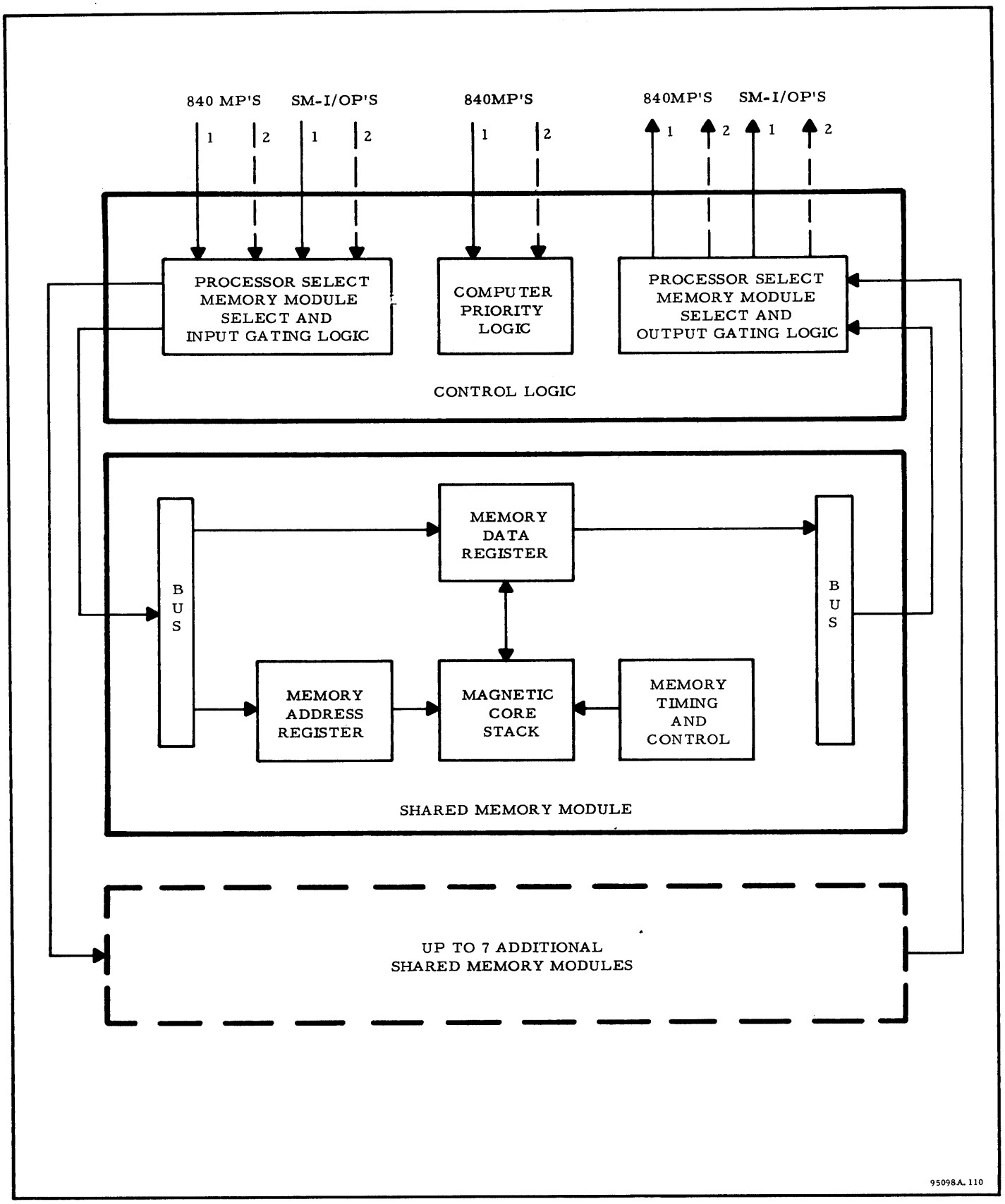


Figure 1-7 Block Diagram - Typical 840MP Shared Memory System

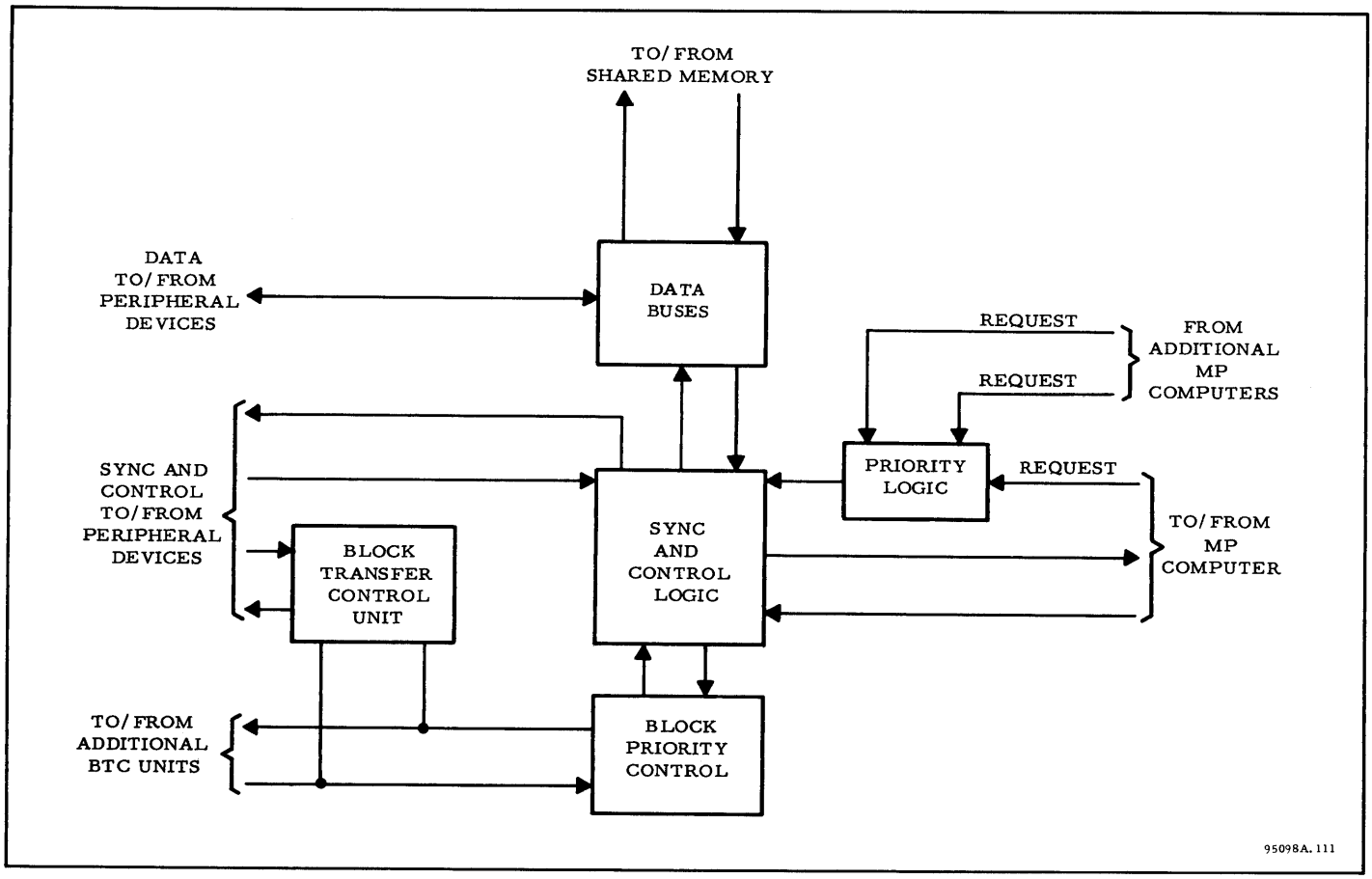


Figure 1-8. Block Diagram - Shared Memory Input/Output Processor

systems are provided; a symbolic assembler and a full FORTRAN IV compiling system. The FORTRAN language specified for this system is the standard ASA FORTRAN IV language, providing compatability with FORTRAN IV systems supplied by other manufacturers. All of the supplied software packages are written in a modular form with a standard program interface specification. This will allow any combination of I/O devices without major revisions of the programs.

SEL 840MP ASSEMBLY PROGRAM

All symbolic instructions in the 840MP Instruction set (with the exception of the I/OP-CEU/TEU, see figure 4-6) are accepted by the assembler. Addresses may be expressed in symbolic, decimal, or octal formats with address arithmetic using any combination of these.

The following special pseudo-operations are also processed by the assembler:

BSS	Reserve block of storage; name at start
BES	Reserve block of storage; name at end
EQU	Define symbolic name
ORG	Set next storage address
ZZZ	Set instruction bits to zero
REL	Set assembly mode to relative
ABS	Set assembly mode to absolute
CALL	Call library subroutine
NAME	Define subroutine name
DATA	Define octal, decimal (fixed or floating), or alphanumeric data
REM	Line printer; top of form
MOR	Pause in assembly process
END	End of program
DAC	Direct address constant
BAR	Bank address register; selects memory bank for program loading

A symbolic side-by-side listing, complete with error messages, is printed (at the operator's option) when the object output tape is generated. Two passes are required to complete the assembly.

SEL 840MP LOADER PROGRAM

The SEL 840MP loader program will load any object program generated by the assembly program or the FORTRAN IV compiler. The loader program provides for relocatable and absolute instructions. The capability of using precompiled subroutine libraries is included in a manner that loads a given routine only once, regardless of how many times it is referenced in the program.

SEL 840MP FORTRAN IV COMPILER

Ease of use is a prime feature of this compiler. Convenience features include:

a. No Reserved Identifiers - All names are available for use as identifiers. For example, this FORTRAN readily distinguishes that:

FORMAT (5H) = A(2); is a FORMAT statement
FORMAT (H5) = A(2); is an arithmetic statement

b. Optional In-line Assembly - This feature allows assembly language coding to be intermixed with FORTRAN statements.

c. Optional Tracing - This feature allows selective object code tracing for diagnostic purposes

d. Optional Mapping - This feature provides a listing of the subprograms required for execution and the names or values and relative location assignments of all variable-array names and constant values used by the program.

e. Optional Chaining - This feature provides for sequential execution of segmented programs.

SEL 840MP DEBUG PROGRAM

The debug program is a utility program designed to help a programmer correct errors in a program while it is in memory. The debug program will:

a. Type the contents of specified memory in octal or command format.

b. Modify the specified memory, input being in octal format.

c. Dump specified memory areas onto paper tape in a format (nonrelocatable) that can be loaded using the loader resident in debug.

d. Enter breakpoints in order to "leap-frog" trace a program.

e. Clear specified areas of memory to zero.

f. Search memory for reference to specified areas.

g. Initiate branches (or halt and branch) to any part of memory.

h. Load a binary tape that was dumped using debug.

Each of these functions is initiated by typing a keyword through the console typewriter keyboard.

SEL 840MP UPDATE PROGRAM

Correction of errors in card decks is a relatively easy procedure, consisting of pulling out the bad cards and inserting new cards. However, symbolic source programs on paper tapes are not so easily corrected or modified. The update program is designed to allow the computer operator to easily correct or modify a symbolic source program tape by providing the following functions:

a. Delete a specified line or group of lines.

b. Insert a new replacement line or lines.

c. List the source programs (or portions of it), complete with line reference numbers.

All references to the symbolic source tape are made by referring to a sequence number. This number is present on all assembly listings and on all listings generated by the update program.

SEL 840MP LIBRARY PACKAGE

The SEL 840MP library package includes the complete set of ASA FORTRAN IV subroutines in the following categories:

a. Single-Precision Floating Point Functions

b. Double-Precision Floating Point Functions

c. Complex Floating Point Functions

d. Integer Functions

e. Input/Output Functions

f. Control Functions

SEL 840MP MAINTENANCE ROUTINES

The SEL 840MP Checkout Program is a complete package designed to give the operator the ability to exercise the memory, the mainframe logic, the I/OP's and associated peripheral equipment.

The memory exerciser routine generates various types of worst case bit patterns and exercises the memory with these patterns while monitoring for errors. Provisions are made for automatic relocating of the exerciser program to allow the entire memory to be included in all tests. Also, included are certain branch/skip instructions which are sequenced and executed through each location in the memory.

The mainframe exerciser routine executes the entire instruction repertoire individually in a large variety of sequences while monitoring the results for errors. Errors are indicated by halts. Pertinent information concerning the instruction that failed and the nature of the failure can be obtained from the displays, the program counter, and certain selected memory locations.

The programs for checking the I/O structures and associated peripheral equipment test the ability of the various I/O units to generate or receive all acceptable characters. A selected input is used and visual monitoring of the control panel or output unit is required by the operator for verification of proper operation. Equipment tested includes punch, and reader as well as optional card punch, card reader, line printer, high-speed paper tape equipment, magnetic tape units, movable head disc, input/output standard teletypewriting, and other units as needed for a particular application.

SECTION II
MACHINE LANGUAGE PROGRAMMING

INTRODUCTION

The 840MP Multiprocessor Computer System is operated by a series of instruction words stored in private or shared memory. The instruction words are successively read from memory locations addressed by the program counter in conjunction with the Bank Address Register (BAR). Each word specifies one operation; that is, transferring a data word from a peripheral device to memory location, adding a memory word to a word in the A-Accumulator, shifting the contents of the A-Accumulator, etc. Normally, the program counter is advanced one count after each instruction to access the instruction word located in the next sequential memory address. The program counter may be preset to any address by branch/skip instructions which detect certain conditions such as A-Accumulator sign positive, arithmetic overflow, input (peripheral) unit ready, etc. The program counter then continues its sequential advance, but starting from the new address. The branch may be to either a higher or lower address.

The 840MP Computer System instruction repertoire includes: load/store instructions which transfer words between memory and the accumulators and index registers; arithmetic instructions; shift instructions which allow the movement of bits within a word; logical instructions (AND, OR, etc.); control instructions (halt, etc.); branch/skip instructions to provide program segment repetition and modification; and input/output instructions to test and command peripheral units and transfer data to and from the computer.

Each instruction word is formed by 24 bits, each of which performs a particular function - defining the operation to be performed, addressing a memory location, defining the number of shifts, etc. The function of a particular bit will vary in different types of instructions. For example, in some words, bit 14 forms part of a memory address; in others, bit 14 forms part of the operation code. The function of the bits depends on the instruction word type defined by the six-bit operation code located in bits 3-8 of the first word of each instruction. There are three major types of instruction words used by the SEL 840MP Computer.

a. Memory reference instructions - instructions containing memory addresses

b. Augmented instructions - instructions containing additional code bits in lieu of memory addresses

c. Input/output instructions - instructions consisting of one or two separate words forming one instruction. Input/output instructions contain augmenting code bits; and, depending on the manner in which they are executed, may or may not contain memory addresses.

Other instructions that do not fall into any of the three major categories are used for numerical conversion during arithmetic operations.

EFFECTIVE VIRTUAL ADDRESS (EVA) refers to the 15-bit address which results after all indexing and indirect addressing operations have been performed.

EFFECTIVE ADDRESS (EA) refers to the 17-bit concatenated address consisting of the 13 least significant bits of the EVA and the 4 bits obtained from the bank address register (BAR) designated by the two most significant bits of the EVA.

FULL SCALE NEGATIVE QUANTITY implies a configuration of a ONE in the sign bit and eight ZEROs (full scale mantissa), when used in reference to the exponent register.

MEMORY REFERENCE INSTRUCTIONS

The 840MP Computer memory reference instructions access an operand from a location in either private or shared memory. These instructions contain a six-bit binary operation code, a 13-bit memory address, a two-bit BAR address, and three address modifier bits as illustrated in figure 2-1.

The first address modifier contained in all 840MP Computer memory reference instructions is the indirect address flag (bit 0). When bit 0 is a ONE, the address specified in the instruction is interpreted by the central processor (CP) as the location containing the effective virtual address (EVA) rather than the operand itself. When the indirect bit is ZERO, the address specified in the instruction contains the operand itself. Indirect addressing (I=ONE) can be performed on as many levels as desired. An indirect address word is

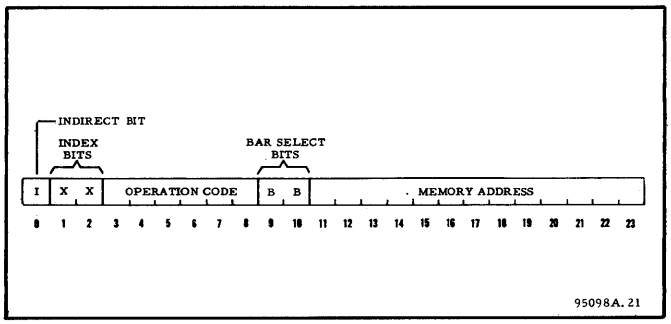


Figure 2-1. Word Format - 840MP Computer Memory Reference Instructions

stored in memory in the format shown in figure 2-2. Each time indirect addressing is employed, an additional 1.75 microseconds (1 machine cycle) is required to call the next word from memory.

The second address modifier consists of the contents of the index register specified by the index bits of the instruction word (bits 1 and 2). The index bits (XX in figures 2-1 and 2-2 are decoded by the 840MP Computer to select index register 1, 2, or 3 for use in address modification. Modifying an address by indexing involves adding the contents of the specified 15-bit index register to the 15-bit memory address field contained in the instruction to create the EVA. If the code in bit positions 1 and 2 is 00_2, no indexing is performed. If the code is 01_2, index register 1 is selected. 10_2 code selects index register 2 and 11_2 selects index register 3. The index registers may be selectively loaded with any 15-bit number ranging from 00000_8 to 77777_8.

The contents of the index registers may be incremented (increased by one) by instruction execution. The incrementing instruction also tests the index register for zero and generates a branch to a specified memory location when the contents of the register is not zero. This feature allows the programmer to create a minor iterative subprogram

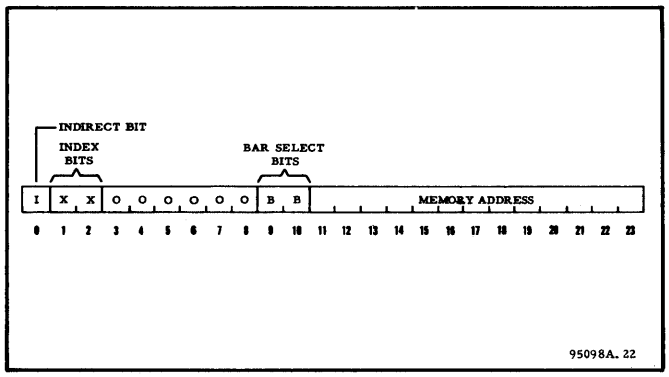


Figure 2-2. Word Format - 840MP Computer Indirect Address Word

that will access a series of sequential memory locations. Such a subprogram or loop, written in assembly language, is shown in table 2-1. A complete description of the SEL 840MP assembly language is presented in Section III.

The series of indexed instructions in table 2-1 beginning with location LOOP serves to add 20 sets of numbers and to store the resulting sums in 20 memory locations. The first pair of numbers is taken from locations 200 (220 - 20) and 300 (320 - 20), added and stored in location 400 (420 - 20). The plus 1 is then added to the index count by the IIB instruction. The resulting -19 count does not equal zero, so the program branches to location LOOP. The next two arguments are taken from locations 201 and 301 and stored in location 401, and the cycle repeated. After adding 20 sets of numbers, the final IIB instruction execution reduces the index count to 00. This causes the next instruction to be executed. The program counter now calls a new set of instructions from memory beginning with location PROG.

The index count may be added to the address in the instruction word or to the indirect address or to both at the programmer's options.

The third address modifier consists of the contents of the BAR specified by the BAR selection bits of the instruction word (bits 9 and 10). Bits 9 and 10 (figures 2-1 and 2-2) are interpreted by the SEL 840MP Computer to select one of four BAR's for use in modification of the effective virtual memory address. If the code in bits 9 and 10 is 00_2, BAR 0 is selected. If the code is 01_2, BAR 1 is selected. A 10_2 code selects BAR 2, while 11_2 specifies BAR 3. Each BAR contains a four-bit address specifying the memory bank that is to be accessed. This four-bit address is appended to the 13-bit memory address in the instruction to create a 17-bit address capable of accessing any of the 98K locations available in the 840 Multiprocessor Computer System.

Two instructions are provided for loading and storing the bank addresses in the BAR's. The load bank address register (LBR) instruction causes the contents of the address (in the LBR instruction) to be transferred to the BAR's. The store bank address register (SBR) instruction causes the contents of the BAR's to be transferred to the specified address. The net effect of the BAR addressing scheme is to designate which four (of the maximum of 12) 8K banks available to a single SEL 840MP Computer are to be used in executing the current program. There are no restrictions on the combinations or order of banks that may be used at one time. All may be in private or shared memory or any combination of the two. Banks may be

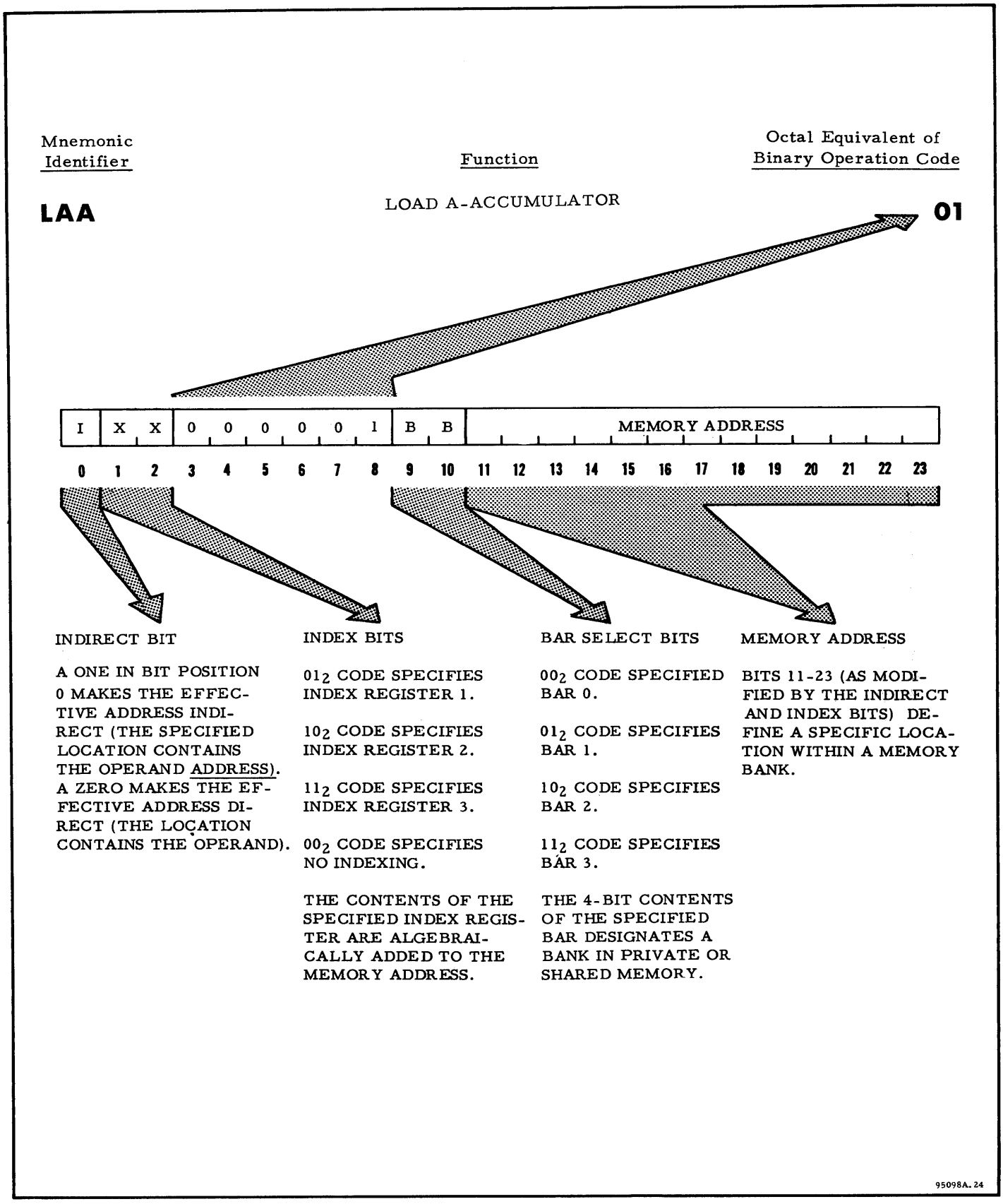


Figure 2-3. Sample Memory Reference Instruction (LAA)

Table 2-1. Sample Iterative Subprogram

Location	Operation	Address	Comments
INDX	LIX	= - 20, 1	Load an index count of -20 into index register 1.
LOOP	LAA	220, 1	Load the A-Accumulator with data word from location (220 + index count).
	AMA	320, 1	Add the contents of the A-Accumulator to the contents of location (320 + index count).
	STA	420, 1	Store the sum (of the previous operation) from the A-Accumulator in location (420 + index count).
	IIB	LOOP, 1	Increment index register; test for index count of zero; branch to location LOOP if not zero; execute next instruction if index equal to zero.
PROG	LAA	IPUT	Next instruction after index loop

addressed in either descending or ascending order (sequentially or nonsequentially).

Another important aspect of BAR addressing is that the selection of a memory bank in mode after all other forms of address modification (indirect addressing and indexing) is completed.

Many of the SEL 840MP Computer branch/skip instructions are memory reference instructions that designate the memory location to which the program counter will be set if a certain condition is found to exist. An example in the branch group is the branch if a negative (BAN) instruction, which specifies a location to which the program is to branch if the sign of the A-Accumulator is negative. Each of the skip instructions tests for specific conditions within the computer; primarily in the arithmetic unit. An example of a memory reference skip instruction is the compare memory and A-Accumulator (CMA) instruction, which compares the value of the contents of a specified memory location with the contents of the A-Accumulator. Depending on the result of the comparison, the computer will either execute the next sequential instruction or skip one or two instructions. Thus, a skip instruction, followed by an unconditional branch instruction, can serve as a conditional branch and offer the programmer a convenient method for branching to specific locations in the program.

Descriptions in this manual of the SEL 840MP Computer memory reference instruction consist of three-letter mnemonic identifier, a description of the instructions function, and a two-character octal operation code. The descriptions also illustrate which address modifiers are used. An example of a memory reference instruction (LAA) is shown in figure 2-3.

AUGMENTED INSTRUCTIONS

Augmented instructions contain no memory address bits in the first instruction word, but do contain additional (augmenting) operation code bits. The augmented instructions have operation codes prefixed by 00_8, 13_8, 17_8 or 21_8. The 21_8 operation codes pertain exclusively to instructions for programming the optional extended arithmetic unit (EAU). The word format for 00_8 and 21_8 augmented instructions is shown in figure 2-4. Instructions with 13_8 and 17_8 pertain primarily to I/O instructions, and their word formats differ from the 00_8 and 21_8 instructions.

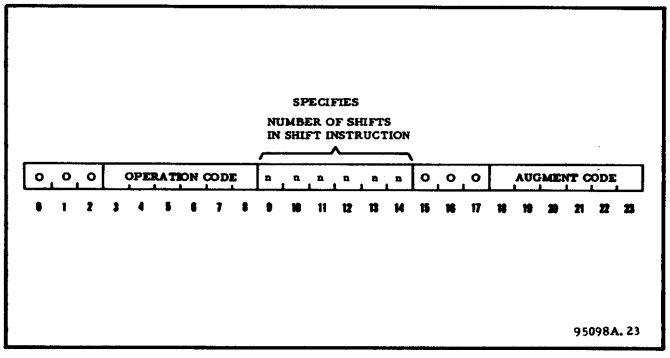


Figure 2-4. Word Format - 840MP Augmented Instruction

Upon detection of a 00_8 or a 21_8 code in bit positions 3-8, the augmenting code bits (18-23) are gated to a special decoding matrix in the

840MP Computer or the EAU. The output of the decoding matrix defines the operation to be performed. If a shift operation is to be executed, the number of shifts will be coded in bit positions 9-14. Other augmented instructions having (00_8 and 21_8) operation codes allow for changing the contents of major registers, skipping sequential instructions, halting the 840MP Computer, etc., without addressing memory locations. These groups generally do not require coding in bits 9-17. The complete operation code for 00_8 and 21_8 augmented instructions consists of the 00_8 or 21_8 operation code prefix followed by a hyphen and the two-digit octal equivalent of the augmenting code bits. Figure 2-5 illustrates examples of both the 00_8 and 21_8 operation codes.

INPUT / OUTPUT INSTRUCTIONS

Input/output instructions contain 13_8 or 17_8 operation codes. Several of the input/output instructions are two word instructions. The two instruction words are stored in sequential memory locations with the second word called automatically by the 840MP Computer. An example of a two-word input/output instruction is shown in figure 2-6. The six input/output instructions are:

Command External Unit (CEU)

Test External Unit (TEU)

Accumulator Word Output to Peripheral (AOP)

Memory Word Output to Peripheral (MOP)

Accumulator Word Input from Peripheral (AIP)

Memory Word Input from Peripheral (MIP)

The three-digit operation code (130_8 or 131_8 in figure 2-6) specifies not only the operation to be performed, but also the mode of execution. The augmenting code bits are in bit positions 9-11, with bit 11 having special significance as the wait bit. The state of the wait bit is indicated in the operation code as shown in figure 2-6. If the instruction is programmed in the wait mode (bit 11= ONE), the 840MP Computer will halt at the current location until the addressed peripheral unit signifies that it is ready to execute the command (or transfer/accept data). If the instruction is programmed in the skip mode (bit 11=ZERO), the 840MP Computer will not wait on the peripheral device, but will execute the next sequential instruction if the peripheral device is not ready. If the peripheral device is ready, the next sequential instruction will be skipped. The wait bit feature is contained in all I/O instructions except the test external unit (TEU) instruction.

All I/O instructions have their augmenting code bits in bit positions 9-11 of the first instruction word. One instruction - A-Accumulator In From Peripheral (AIP) - has an additional augmenting code bit in bit position 12. This is the character merge provision that allows for character assembly in the A-Accumulator. (This provision is described in detail later in this section.) Thus, the AIP instruction has a four-digit operation code (that is, 1734) which describes the state of the merge and wait bits.

As shown in figure 2-6, a two-word instruction may have two different formats for the second word depending on the state of the indirect bit (bit position 0) in the first word. If the indirect bit is a ONE, the second word format is the same as that of all other indirect address words. Address modification is performed in an identical manner. If the indirect bit is ZERO in the first instruction word, the second word contains a 24-bit command or data word. These two modes of designating the second word format are referred to as the address and immediate modes.

MACHINE LANGUAGE INSTRUCTION SET

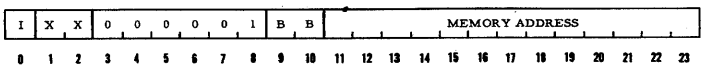
The instruction words which initiate the various 840MP Computer System machine operations are described in detail on the following pages. The descriptions include the mnemonic identifier (used in assembly language) and the operation code in both types. The binary word format shows bit assignments for operation code, augment code, operand address and address modifiers (indirect, index, and BAR select bits). A brief explanation of the function(s), register(s) affected, memory cycles required, control panel indicators (if any), and special notes complete the description.

LOAD/STORE INSTRUCTIONS

A total of 10 instructions comprise this group. Five of these instructions provide the means of accessing memory both to load data into the major register for computation and/or data manipulation, and to return the results to memory. Two other instructions are provided to transfer instructions between the index registers and memory. An additional two instructions are provided for loading and storing the contents of the BAR's. The last instruction in the group is used to transfer the settings on the 24 console SENSE/HALT/LOAD T switches to the A-Accumulator. This instruction permits the use of the switches for manual modification of the program at predetermined points.

LAA 01

LOAD A-ACCUMULATOR



The contents of the effective memory address replace the previous contents of the

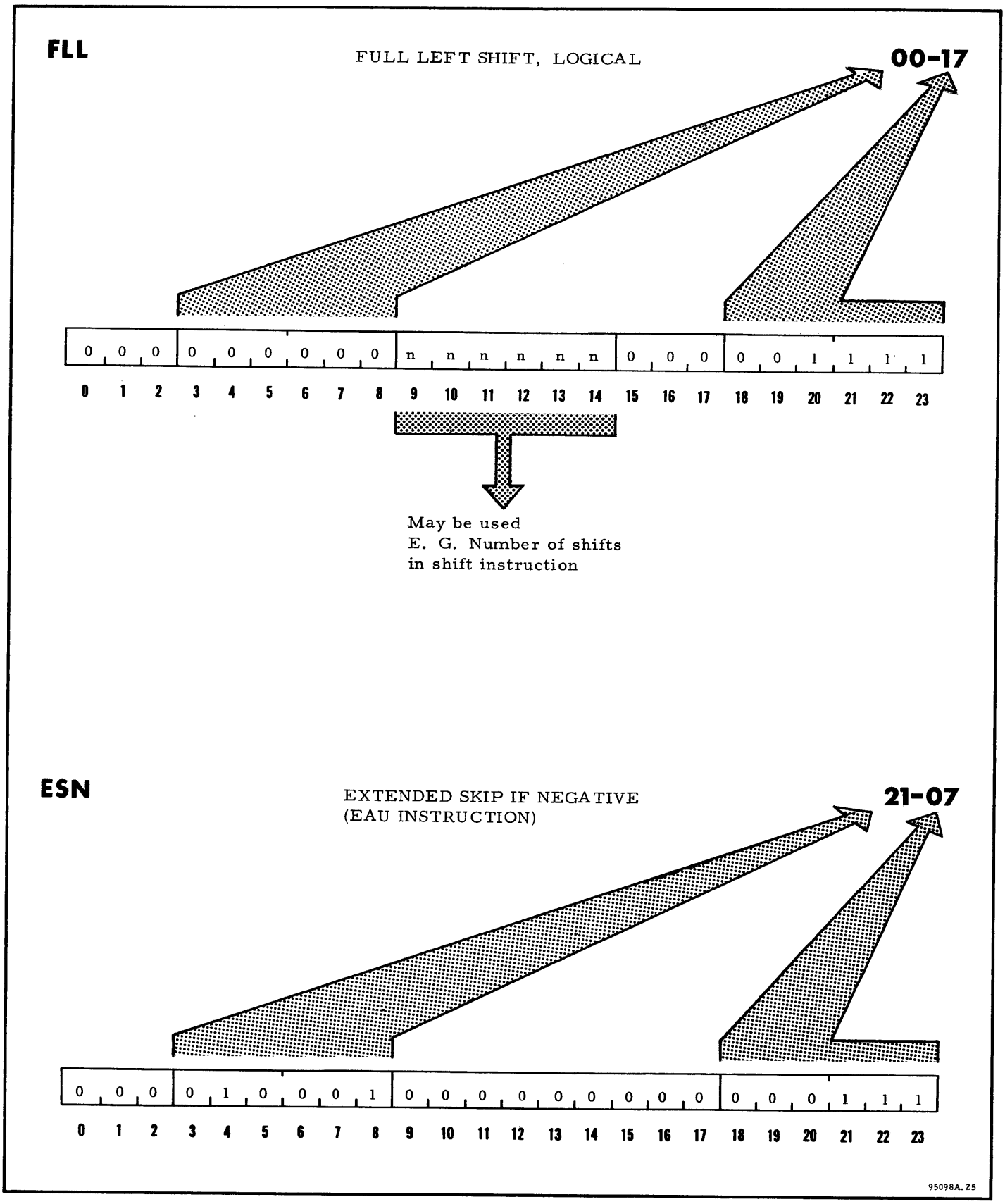


Figure 2-5. Sample 00_8 and 21_8 Augmented Instructions

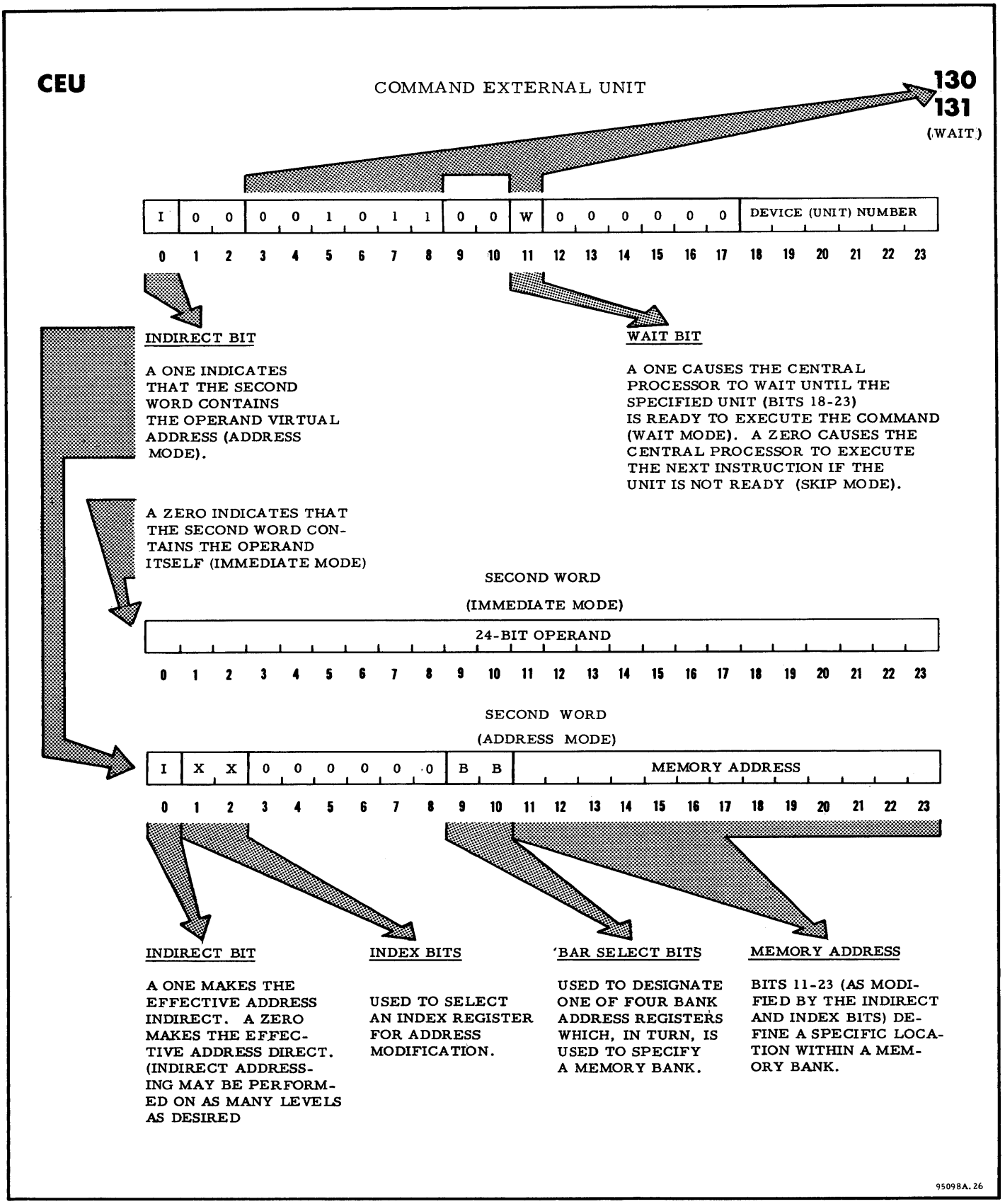


Figure 2-6. Sample Input/Output Instruction

A-Accumulator. The contents of memory are unchanged.

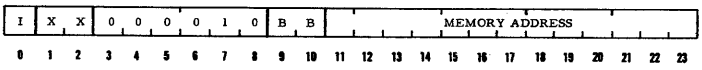
NOTE

The A-Accumulator must be loaded with the augend, minuend, and most significant bits of the dividend prior to add, subtract, and divide instructions.

Timing:	2 cycles
Indicators:	None
Registers Affected:	A-Accumulator

LBA 02

LOAD B-ACCUMULATOR



The contents of the effective memory address replace the previous contents of the B-Accumulator. The contents of the memory are unchanged.

NOTE

The B-Accumulator must be loaded with the multiplier prior to a multiply instruction and with the least significant bits of the dividend prior to divide instructions.

Timing:	2 cycles
Indicators:	None
Registers Affected:	B-Accumulator

STA 03

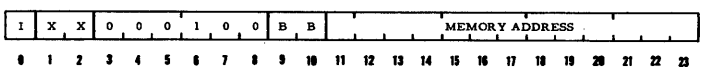
STORE A-ACCUMULATOR

The contents of the A-Accumulator replace the previous contents of the effective memory address. The contents of the A-Accumulator are unchanged.

Timing:	2 cycles
Indicators:	None
Registers Affected:	None

STB 04

STORE B-ACCUMULATOR

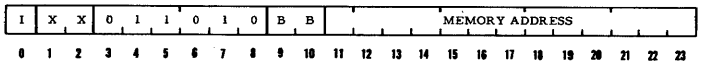


The contents of the B-Accumulator replace the previous contents of the effective memory address. The contents of the B-Accumulator are unchanged.

Timing:	2 cycles
Indicators:	None
Registers Affected:	None

LIX 32

LOAD INDEX REGISTER



The least significant 15 bits of the contents of the effective memory address replace the previous contents of the designated 15-bit index register. The contents of memory are unchanged.

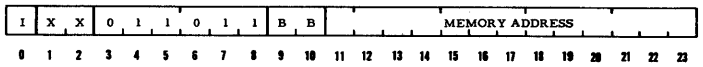
NOTE

Bits 1 and 2 specify the index register to be loaded. If bit zero equals one, indirect addressing will be performed and no index register need be specified. However, bits 1 and 2 may be used to specify an index register as an address modifier when the indirect address mode is used (bit 0 = 1). The last indirect address word in an indirect chain must specify which index register is to be loaded. Also, bit position 0 must contain a ZERO in the last word of an indirect chain.

Timing:	2 cycles
Indicators:	None
Registers Affected:	Index register 1, 2, or 3

STI 33

STORE INDEX REGISTER



The 15-bit contents of the designated index register replace the 15 least significant bits of the effective memory address. The sign bit and 7 most significant bits of the effective memory address are replaced by ZERO's at the time of transfer. The contents of the index registers are unchanged.

NOTES

Note 1: The index bits (1 and 2) function as address modifiers only if bit 0 is a ONE (indirect address mode). Otherwise, bits 1 and 2 designate the index

NOTES (Cont'd)

register that is to be the source register. If indirect chaining is employed, the last instruction must specify the index register whose contents is to be stored, while all other instructions in the chain may specify any index register as an address modifier.

Note 2: If no index register is specified, ZERO's will be loaded into the effective memory address.

Timing:	2 cycles
Indicators:	None
Registers Affected:	None

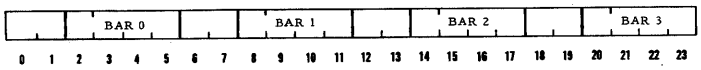
LBR 37

LOAD BANK ADDRESS REGISTER

The contents of the effective memory address as defined below replace the previous contents of the four bank address registers. The contents of memory are unchanged.

NOTES

Note 1: Prior to loading the bank address registers, the desired bit configuration must be loaded into the effective memory address in the format shown below.

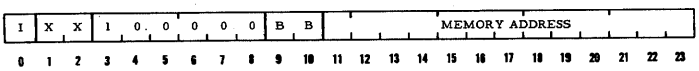


Note 2: As with any other memory reference instruction, the LBR instruction must specify (in bits 9 and 10) which BAR is to be used to create the effective memory address. The BAR select bits (BB) specify the BAR register. All four BAR's are loaded simultaneously.

Timing:	2 cycles
Indicators:	None
Registers Affected:	Bank address register

SBR 40

STORE BANK ADDRESS REGISTER



The contents of the four bank address registers replace the previous contents of effective memory address. The contents of the bank address register are unchanged.

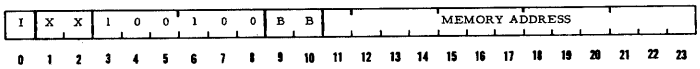
NOTE

The BAR select bits (BB) are used as address modifiers and not to designate one of the BAR's as a source register; the contents of all four BAR's are stored simultaneously.

Timing:	2 cycles
Indicators:	None
Registers Affected:	None

IAM 44

INTERCHANGE A-ACCUMULATOR AND MEMORY

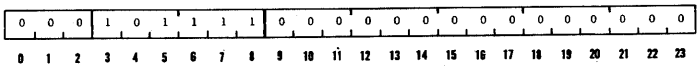


The contents of the effective memory address replace the contents of the A-Accumulator, and the contents of the A-Accumulator replace the contents of the effective memory address.

Timing:	3 cycles
Indicators:	None
Registers Affected:	A-Accumulator

LCS 57

LOAD CONTROL SWITCHES



The status of SENSE/HALT/LOAD T switches 0 through 23 on the control console replace the previous contents of the A-Accumulator.

Timing:	1 cycle
Indicators:	None
Registers Affected:	A-Accumulator

ARITHMETIC INSTRUCTIONS

All arithmetic functions of the SEL 840MP Computer are performed by this group of 10 instructions. The add memory to A-Accumulator (AMA) instruction adds the contents of a memory location to that of the A-Accumulator and stores the sum in the A-Accumulator. The add A-Accumulator to memory (AAM) instruction performs a similar operation, but the result is stored in the memory location which previously contained one of the operands. A third add instruction; add memory to index register (AMX), adds the contents of the addressed memory location to that of the index

register selected by the instruction, and stores the 15 least significant bits of the sum in the index register.

The subtract memory from A-Accumulator (SMA) instruction subtracts the contents of a memory location from the contents of the A-Accumulator, and places the result in the A-Accumulator.

The multiply (MPY) instruction multiplies the single-precision contents of a memory location by the single-precision contents of the B-Accumulator, and places the double-precision product in the A- and B-Accumulators.

The divide (DIV) instruction divides the double-precision dividend (in the A- and B-Accumulators) by the single-precision contents of a memory location and places the single-precision quotient in the A-Accumulator and the remainder in the B-Accumulator. Round A-Accumulator (RNA) is the instruction used to round the contents of the A-Accumulator by the magnitude of the B-Accumulator contents.

The convert number system (CNS) and negate (NEG) instructions are used primarily in the arrangement of data formats to satisfy I/O requirements, but also find use in creating special indicators and constant words. The CNS instruction allows the contents of the A-Accumulator, excluding sign, to be changed from sign magnitude form to two's complement form, or vice versa. The NEG instruction causes the contents of the A-Accumulator, including sign, to be two's complemented. The copy sign of B-Accumulator (CSB) instruction is used with double-precision arithmetic to handle carries and borrows between accumulators.

AMA 05

ADD MEMORY TO A-ACCUMULATOR

The contents of the effective memory address (the addend) are algebraically added to the contents of the A-Accumulator (the augend). The sum is stored in the A-Accumulator, and the sign of the A-Accumulator is set to the algebraic sign of the sum. The contents of memory are unchanged.

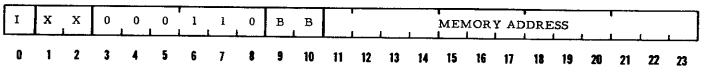
NOTE

The A-Accumulator must be loaded with the augend prior to execution of the AMA instruction, either by execution of an LAA instruction, or as the result of a previous operation involving the A-Accumulator.

Timing: 2 cycles
Indicators: OVERFLOW, if the SUM exceeds 23 bits plus sign
Registers Affected: A-Accumulator

SMA 06

SUBTRACT MEMORY FROM A-ACCUMULATOR



The contents of the effective memory address (the subtrahend) are algebraically subtracted from the contents of the A-Accumulator (the minuend). The difference replaces the minuend in the A-Accumulator and the sign of the A-Accumulator is set to the sign of the algebraic sum. The contents of memory are unchanged.

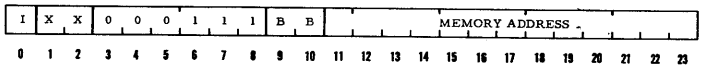
NOTE

The A-Accumulator must be loaded prior to the execution of the SMA instruction, either through execution of an LAA instruction, or as the result of a previous operation involving the A-Accumulator.

Timing: 2 cycles
Indicators: OVERFLOW, if difference exceeds 23 bits plus sign
Registers Affected: A-Accumulator

MPY 07

MULTIPLY



The contents of the effective memory address (multiplicand) are multiplied by the contents of the B-Accumulator (multiplier). The most significant half of the product replaces the previous contents of the A-Accumulator. The least significant half of the product replaces the previous contents of the B-Accumulator. The sign of the A-Accumulator is determined by the algebraic sign of the product; the sign of the B-Accumulator is set to plus. The contents of memory are unchanged.

NOTE

If the multiplier and multiplicand are considered to be integers (binary point to the right of bit 23), the product is a double-precision integer (binary point to the right of bit 46 in the B-Accumulator). If the multiplier, multiplicand,

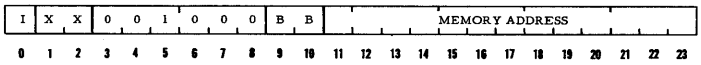
NOTE (Cont'd)

or both are scaled left 2^m and 2^n, then the product is scaled left 2^{m+n}.

Timing:	6 cycles
Indicators:	None
Registers Affected:	A-Accumulator, B-Accumulator

DIV 10

DIVIDE



The contents of the A- and B-Accumulators (double-length dividend) are divided by the contents of the effective memory address (single-length divisor). The quotient is stored in the A-Accumulator and the remainder is stored in the B-Accumulator. The sign of the quotient is set to the algebraic product of the divisor and dividend signs, and the sign of the remainder is set to the sign of the original dividend. The contents of memory are unchanged.

NOTES

Note 1: The dividend is assumed to be a double-precision quantity (46 bits + sign) which is to be divided by a single-precision quantity (23 bits + sign). The result will be two single-precision quantities; that is, quotient and remainder. If the divisor is less than, or equal to, the most significant half of the dividend (A-Accumulator), an overflow will result.

Note 2: If it is desired to divide one single-precision quantity by another, the divide may be implemented by clearing the A-Accumulator to zero and loading the single-precision dividend into the B-Accumulator. The quotient and remainder will be placed in the A- and B-Accumulator, respectively, as when dividing a double-precision quantity. No overflow will result from this division unless the divisor is equal to zero.

Note 3: Fractional parts in the quotient can be obtained by scaling the divisor and/or the dividend.

NOTES (Cont'd)

Considering the binary point to be located between the sign bit and most significant bit, the binary point in the quotient can be obtained by the following formula:

C(M) dividend - B(N) divisor = B(M-N) quotient

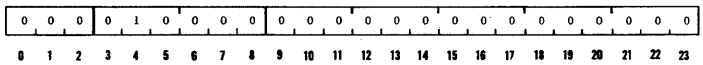
where: $-46 \leq M \leq 46$ and $-23 \leq N \leq 23$

If the scale factor is greater than B_{23} an overflow will result.

Timing:	15 cycles
Indicators:	OVERFLOW, if divisor is less than, or equal to, the most significant half of the dividend if the divisor equals zero, or if the scale factor exceeds B_{23} (when using fractional scaling)
Registers Affected:	A-Accumulator, B-Accumulator

CNS 20

CONVERT NUMBER SYSTEM

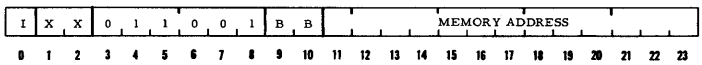


Bits one through 23 of the words in the A-Accumulator are two's complemented, if the negative sign is unchanged. If the word was originally in the sign magnitude form, it is converted to two's complement form. If originally in two's complement form, it is converted to sign magnitude form. Positive-signed words are unchanged since the word structure is the same for both forms.

Timing:	1 cycle
Indicators:	None
Registers Affected:	A-Accumulator

AAM 31

ADD A-ACCUMULATOR TO MEMORY



The contents of the A-Accumulator (addend) are algebraically added to the contents of the effective memory address (augend). The sum replaces the augend in the effective memory address. The contents of the A-Accumulator are unchanged.

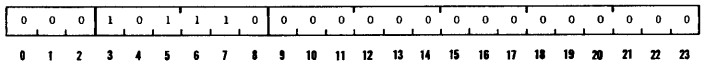
NOTE

This instruction is useful in the adding of a constant, stated or derived, to a series of related data words.

Timing:	3 cycles
Indicators:	OVERFLOW, if the sum exceeds 23 bits plus sign
Registers Affected:	A-Accumulator

NEG 56

NEGATE A-ACCUMULATOR

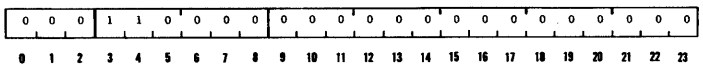


The contents of the A-Accumulator are two's complemented.

Timing:	1 cycle
Indicators:	None
Registers Affected:	A-Accumulator

RNA 60

ROUND A-ACCUMULATOR

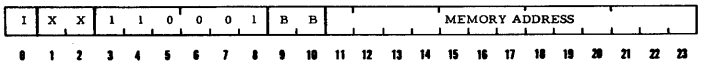


The contents of the A-Accumulator are increased by one if the second most significant bit (B_1) of the B-Accumulator is a ONE.

Timing:	1 cycle
Indicators:	OVERFLOW, if the result in the A-Accumulator exceeds 23 bits.
Registers Affected:	A-Accumulator

AMX 61

ADD MEMORY TO INDEX REGISTER



The contents of the effective memory address (the addend) are algebraically added to the contents of the selected index register (the augend). The 15 least significant bits of the sum are stored in the selected index register. The contents of memory are unchanged.

NOTE

The index bits (1 and 2) function as address modifiers in this instruction, only if indirect addressing is employed (I = ONE). Otherwise,

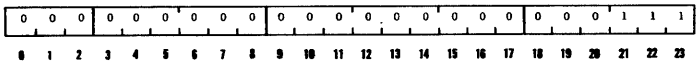
NOTE (Cont'd)

bits 1 and 2 designate which index register is to be selected for the addition. If indirect chaining is employed, the last indirect address word in the chain must specify which index register is to be selected for the addition, and must have the indirect bit (I) set to ZERO. All other indirect address words in the chain may specify any index register for use as an address modifier.

Timing:	2 cycles
Indicators:	None
Registers Affected:	Index register 1, 2, or 3

CSB 00-07

COPY SIGN OF B-ACCUMULATOR



The sign of the B-Accumulator is transferred to the carry latch and replaced with a ZERO (plus).

NOTE

This instruction is used with double-precision arithmetic to obtain the proper double-precision word format.

Timing:	1 cycle
Indicators:	None
Registers Affected:	B-Accumulator

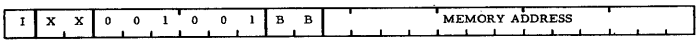
BRANCH/SKIP INSTRUCTIONS

This group of 13 instructions provides the decision-making capability of the SEL 840MP Computer. The majority of these instructions are branch instructions which contain memory addresses and preset the program counter to the effective virtual address contained in the instruction. The skip instructions advance the program counter by one or two additional locations. Three of the branch instructions - unconditional branch (BRU), store place and branch (SPB), and priority interrupt return (PIR) - are unconditional branches. The remainder of the branch instructions will cause a branch in the program only if certain conditions exist; that is, A-Accumulator sign positive or negative, arithmetic overflow, A-Accumulator contents zero or non-zero index count. The skip instructions are all conditional instructions that test the sign of the contents of a specific memory word or the A-Accumulator, the status of a console switch, or the contents of a specified memory

location for zero; or compare the contents of a specified memory location with that of the A-Accumulator.

BRU 11

UNCONDITIONAL BRANCH

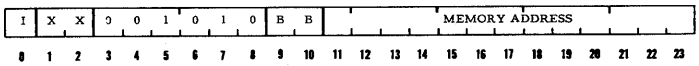


The effective virtual address replaces the previous contents of the program counter. This causes the program to begin with the instruction located at the new memory address and proceed through sequential addresses from that point.

Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter

SPB 12

STORE PLACE AND BRANCH



The 15-bit contents of the program counter (effective virtual address) replace the previous contents of the effective virtual memory address. The address is stored in bit positions 9-23. The effective virtual memory address (plus 1) then replaces the previous contents of the program counter.

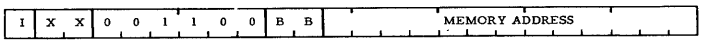
NOTE

When the SPB instruction is executed after being accessed from a dedicated interrupt location, the return effective virtual address (two BAR select bits + 13 address bits) is stored in bit positions 9-23 of the first location in the interrupt routine. The contents of BAR 0 is stored in bit positions 0-5, the state of the overflow latch in bit position 6, and the state of the protect latch (optional) in bit 7. The second word of the subroutine (effective address + 1) must contain the first subroutine instruction. The last instruction in the subroutine must be a priority interrupt return (PIR), addressing the first location to replace the stored program count in the program counter, and to restore the original status of the BAR 0, overflow latch, and protect latch.

Timing:	2 cycles
Indicators:	None
Registers Affected:	Program counter, bank address register

IMS 14

INCREMENT MEMORY AND SKIP



The contents of the effective memory address are increased by one. If the contents then equal zero, the next instruction (NI) is skipped and the second sequential instruction (NI+1) is executed. If the contents of the effective address do not equal zero the next instruction (NI) is executed.

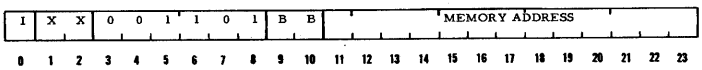
NOTE

This instruction allows any memory location to be used as an auxiliary counter by loading that location with a negative count.

Timing:	3 cycles
Indicators:	None
Registers Affected:	Program counter

CMA 15

COMPARE MEMORY AND A-ACCUMULATOR



The contents of the effective memory address and the A-Accumulator are algebraically compared.

If A < M, the next sequential instruction (NI) is executed.

If A = M, the next sequential instruction (NI) is skipped and the second sequential instruction (NI+1) is executed.

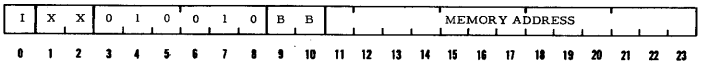
If A > M, the next two sequential instructions (NI) and (NI+1) are skipped and the third sequential instruction (NI+2) is executed.

The contents of memory and the A-Accumulator are unchanged.

Timing:	3 cycles
Indicators:	None
Registers Affected:	Program counter

BAZ 22

BRANCH IF A-ACCUMULATOR ZERO

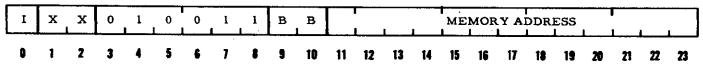


If the contents of the A-Accumulator are zero, the effective virtual address replaces the previous contents of the program counter. If the contents of the A-Accumulator are not zero, the next sequential instruction is executed.

Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter

BAN 23

BRANCH IF A-ACCUMULATOR NEGATIVE

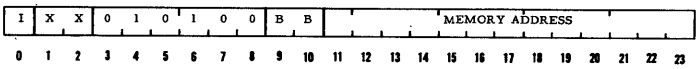


If the contents of the A-Accumulator are negative, the effective virtual address replaces the previous contents of the program counter. If the contents of the A-Accumulator are greater than, or equal to, zero, the next sequential instruction is executed.

Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter

BAP 24

BRANCH IF A-ACCUMULATOR POSITIVE

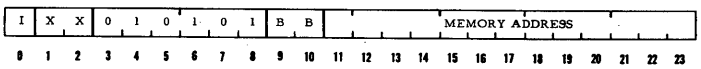


If the contents of the A-Accumulator are greater than, or equal to, zero, the effective virtual address replaces the previous contents of the program counter. If the contents of the A-Accumulator are less than zero, the next sequential instruction is executed.

Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter

BOF 25

BRANCH ON OVERFLOW



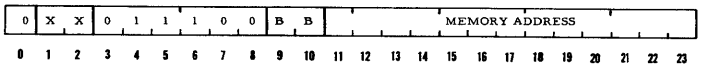
If the overflow latch is set, the effective virtual address replaces the previous contents of the program counter and resets the overflow latch.

If the overflow latch is not set, the next sequential instruction is executed.

Timing:	1 cycle
Indicators:	OVERFLOW is extinguished
Registers Affected:	Program counter

IIB 34

INCREMENT INDEX AND BRANCH



The contents of the designated index register are increased by one. If the contents of that register are then non-zero, the effective virtual address replaces the previous contents of the program counter. If the contents of that register are zero, the next sequential instruction isexecuted.

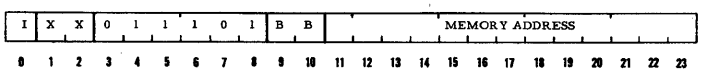
NOTE

Bits 1 and 2 cannot be used as address modifiers with the IIB instruction. These bits are used to specify the index register that is to be incremented.

Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter, index register 1, 2, or 3

SMP 35

SKIP IF MEMORY POSITIVE

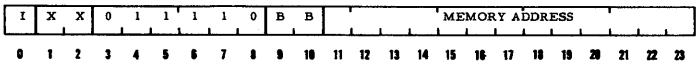


If the contents of the effective memory address are greater than, or equal to, zero, the next sequential instruction (NI) is skipped and the second sequential instruction (NI+1) is executed. If the contents of the effective memory address are less than zero, the next sequential instruction (NI) is executed.

Timing:	2 cycles
Indicators:	None
Registers Affected:	Program counter

PIR 36

PRIORITY INTERRUPT RETURN



The contents of the effective memory address replace the previous contents of the program counter, and the highest set priority interrupt latch is reset.

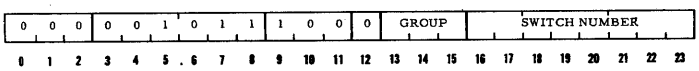
NOTE

The PIR instruction provides a means of returning to the main program after an interrupt subroutine is completed. Upon execution of this instruction the contents of BAR 0 and the status of the overflow latch (in the computer) are restored to the values which existed when the interrupt occurred. The status of the optional protect latch is also restored if the PIR instruction is protected.

Timing: 2 cycles
Indicators: None
Register Affected: Program counter bank address register

SNS 134

SENSE NUMBERED SWITCH



If the designated console control switch is set the next instruction is executed; if that switch is not set or if the SENSE HALT switch is not in the SENSE position, the next instruction is skipped and the second sequential instruction is executed.

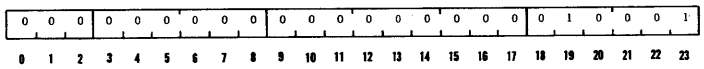
NOTE

Instruction word bits 13 through 23 designate which of the 24 control switches 0-23 is to be tested. Bit 13 designates switches 0-7; bit 14 designates switches 8-15; bit 15 designates switches 16-23. Bits 16 through 23 of the instruction word then designate which of the eight switches in the selected group are to be tested. For example, bits 14 and 23 are set to ONE and bits 13 and 15 through 22 are set to ZERO to designate switch 15 as the SENSE switch to be tested. If more than one switch is selected in the instruction, then the skip will not occur if any selected switch is set.

Timing: 1 cycle
Indicators: None
Registers Affected: Program counter

SAS 00-21

SKIP ON A-ACCUMULATOR SIGN



The contents of the A-Accumulator are tested. If the contents are negative, the next sequential instruction (NI) is executed. If the contents are zero, the second sequential instruction (NI+1) is executed. If the contents are positive, the third sequential instruction (NI+2) is executed.

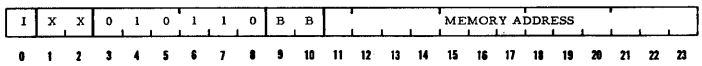
Timing: 1 cycle
Indicators: None
Registers Affected: Program counter

LOGICAL INSTRUCTIONS

These three logical instructions affect only the A-Accumulator. These instructions are provided to allow the logical modification of instruction and data words; that is, masking portions of a single word or merging and comparing two words.

MEA 26

MEMORY EXCLUSIVE OR A-ACCUMULATOR



The contents of the effective memory address and the contents of the A-Accumulator form a bit-by-bit (no carry) arithmetic sum which is stored in the A-Accumulator.

NOTE

This instruction may be used as a comparison operation to detect specific locations of like and unlike bits. The bits in the same bit locations must be different to produce a 1 in the result.

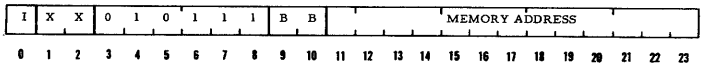
For example:

101	101	101	010	101	010	101	010	A-Acc. word
101	010	101	110	011	010	101	010	Mem. word
000	111	000	100	110	000	000	000	Log. Result

Timing: 2 cycles
Indicators: None
Registers Affected: A-Accumulator

MAA 27

MEMORY AND A-ACCUMULATOR



The contents of the effective memory address and the contents of the A-Accumulator form a logical product which is stored in the A-Accumulator.

NOTE

This instruction may be used in masking operations to separate or delete portions of a computer word. A 1 must occur in the same bit position of each word for a 1 to result.

For example:

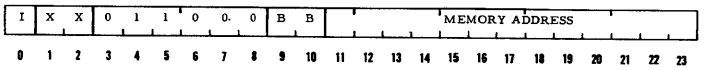
```
000 000 000 000 111 111 111 111 A-Acc. word
101 010 101 010 101 010 101 010 Mem. word

000 000 000 000 101 010 101 010 Log. Product
```

Timing: 2 cycles
Indicators: None
Registers Affected: A-Accumulator

MOA 30

MEMORY OR A-ACCUMULATOR



The contents of the effective memory address and the contents of the A-Accumulator form a logical sum which is stored in the A-Accumulator.

NOTE

This instruction may be used in merging operations to combine separate portions of two words. A 1 in a given bit position of either word produces a 1 in the result.

For example:

```
000 000 000 010 111 111 111 111 A-Acc. word
101 010 101 010 101 010 101 010 Mem. word

101 010 101 010 111 111 111 111 Log. Sum
```

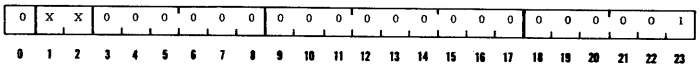
Timing: 2 cycles
Indicators: None
Registers Affected: A-Accumulator

REGISTER CHANGE INSTRUCTIONS

This group of seven instructions is used primarily for manipulating data and creating specific data formats. Two instructions (TAI and TBI) permit the index registers to be loaded from the A- and B-Accumulators. Other instructions in this group permit the transfer of data between the A- and B-Accumulators and interchanging the A- and B-Accumulator contents.

TAI 00-01

TRANSFER A-ACCUMULATOR TO INDEX REGISTER

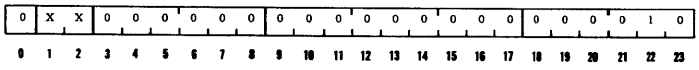


The contents of the designated index register are replaced by the contents of the least significant 15 bits (9-23) of the A-Accumulator. The contents of the A-Accumulator are unchanged.

Timing: 1 cycle
Indicators: None
Registers Affected: Index register 1, 2, or 3

TBI 00-02

TRANSFER B-ACCUMULATOR TO INDEX REGISTER

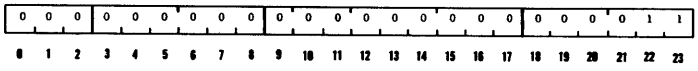


The contents of the designated index register are replaced by the contents of the least significant 15 bits (9-23) of the B-Accumulator. The contents of the B-Accumulator are unchanged.

Timing: 1 cycle
Indicators: None
Registers Affected: Index register 1, 2, or 3

CLA 00-03

CLEAR A-ACCUMULATOR

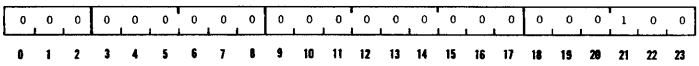


The contents of the A-Accumulator are replaced with all ZERO's.

Timing: 1 cycle
Indicators: None
Registers Affected: A-Accumulator

TBA 00-04

TRANSFER B-ACCUMULATOR

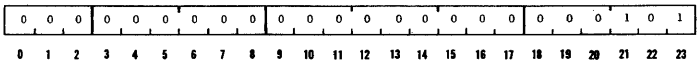


The contents of the B-Accumulator replace the contents of the A-Accumulator. The contents of the B-Accumulator are unchanged.

Timing:	1 cycle
Indicators:	None
Registers Affected:	A-Accumulator

TAB 00-05

TRANSFER A-ACCUMULATOR TO B-ACCUMULATOR

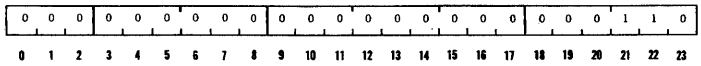


The contents of the A-Accumulator replace the contents of the B-Accumulator. The contents of the A-Accumulator are unchanged.

Timing:	1 cycle
Indicators:	None
Registers Affected:	B-Accumulator

IAB 00-06

INTERCHANGE A- AND B-ACCUMULATORS

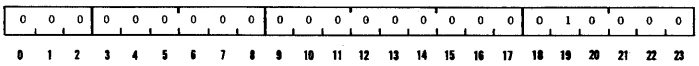


The contents of the A- and B-Accumulators are interchanged.

Timing:	1 cycle
Indicators:	None
Registers Affected:	A-Accumulator, B-Accumulator

ASC 00-20

COMPLEMENT A-ACCUMULATOR SIGN



The A-Accumulator sign bit is complemented.

Timing:	1 cycle
Indicators:	None
Registers Affected:	A-Accumulator

SHIFT INSTRUCTIONS

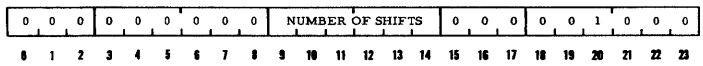
The nine instructions forming the shift group are augmented 00$_8$ instructions with bits nine through 14 containing the binary shift count. While the actual count is in binary code, the number of shifts is usually specified in decimal in symbolic coding. Up to 63 (77$_8$) shifts may be programmed, but only 48 (60$_8$) are required to fully rotate both the A- and B-Accumulators.

There are two types of shift instructions; arithmetic shifts which bypass the sign bit, and logical shifts which move all 24 bits. Right arithmetic shifts move bits from position 1 to 2, 2 to 3, 3 to 4, etc., with bit 1 set to the state of the sign bit, and the bit originally located in bit 23 is shifted off. In left arithmetic shifts, the bits are moved from positions 23 to 22, 22 to 21, 21 to 20, etc., with ZERO's loaded into bit 23, and the original bits shifted off bit position 1. The sign bit remains intact. In right logical shifts, the sign bit is shifted to position 1, 1 to 2, 2 to 3, etc., and ZERO's are shifted into the sign position. In left logical shifts, ZERO's are loaded into bit position 23, 23 to 22, 22 to 21, etc., and the sign bit is shifted off.

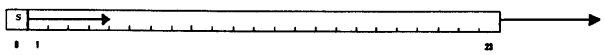
Both accumulators may be shifted together in right arithmetic, left logical rotate, and left normalize modes. The rotate instruction moves the sign bit of the A-Accumulator to bit position 23 of the B-Accumulator as the other bits are moved left from B to A. The normalize instruction left shifts bits one through 23 of the B-Accumulator and bits zero through 23 of the A-Accumulator until the bits in the A-Accumulator sign position and position one are unlike.

RSA 00-10

RIGHT SHIFT A-ACCUMULATOR, ARITHMETIC



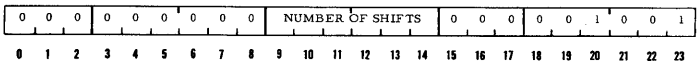
Bits 1-23 of the A-Accumulator are shifted right n places as specified by the code in bit positions 9-14 of the instruction word. The sign bit (bit 0) is unchanged, but does supply bits (ONE's if negative, ZERO's if positive) to bit position one as the most significant bits are shifted right. The least significant bits are shifted off bit position 23.



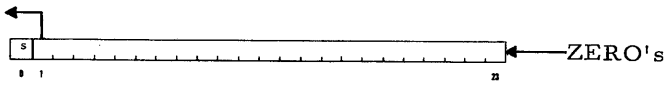
Timing:	1-4 Shifts	2 cycles
	5-8 Shifts	3 cycles
	9-12 Shifts	4 cycles
	13-16 Shifts	5 cycles
	17-20 Shifts	6 cycles
	21-23 Shifts	7 cycles
Indicators:		None
Registers Affected:		A-Accumulator

LSA 00-11

LEFT SHIFT A-ACCUMULATOR, ARITHMETIC



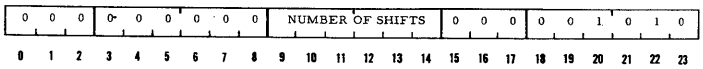
Bits 1-23 of the A-Accumulator are shifted left n places as specified by the code in bit positions 9-14 of the instruction word. The sign bit is unchanged, and the most significant bits are shifted off bit position 1. The least significant bits are replaced by ZERO's entered into bit position 23.



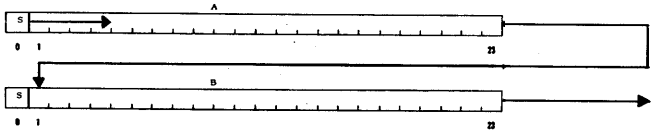
Timing:	1-4 Shifts	2 cycles
	5-8 Shifts	3 cycles
	9-12 Shifts	4 cycles
	13-16 Shifts	5 cycles
	17-20 Shifts	6 cycles
	21-23 Shifts	7 cycles
Indicators:		None
Registers Affected:		A-Accumulator

FRA 00-12

FULL-RIGHT ARITHMETIC SHIFT



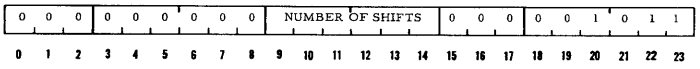
Bits 1-23 of the A- and B-Accumulators are shifted right n places as specified by the code in bit positions 9-14 in the instruction word. Neither sign bit is shifted, but the A-Accumulator sign (bit 0) supplies bits (ONE's if negative, ZERO's if positive) to bit position one of the A-Accumulator as the most significant A-Accumulator bits are shifted right. The least significant bits of the A-Accumulator are shifted off bit position 23 to bit position one of the B-Accumulator. The least significant bits of the B-Accumulator are shifted off bit position 23.



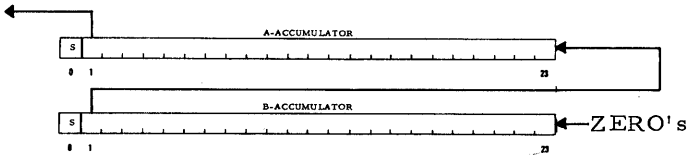
Timing:	1-4 Shifts	2 cycles
	5-8 Shifts	3 cycles
	9-12 Shifts	4 cycles
	13-16 Shifts	5 cycles
	17-20 Shifts	6 cycles
	21-24 Shifts	7 cycles
	25-28 Shifts	8 cycles
	29-32 Shifts	9 cycles
	33-36 Shifts	10 cycles
	37-40 Shifts	11 cycles
	41-44 Shifts	12 cycles
	45-46 Shifts	13 cycles
Indicators:		None
Registers Affected:		A-Accumulator, B-Accumulator

FLA 00-13

FULL-LEFT ARITHMETIC SHIFT



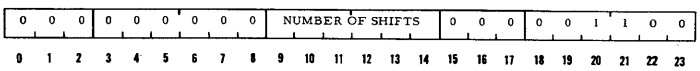
Bits 1-23 of the A- and B-Accumulators are shifted left n places as specified by the code in bit positions 9-14 of the instruction word. ZERO's are entered into bit position 23 of the B-Accumulator as the accumulators are shifted left. The most significant bits of the B-Accumulator are shifted off bit position 1 (B) to bit position 23 of the A-Accumulator. The most significant bits of the A-Accumulator are shifted off bit position 1 (A).



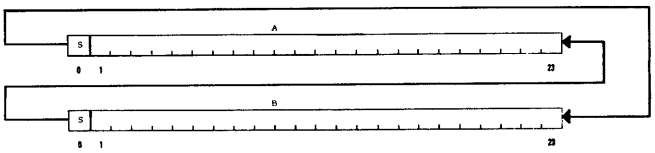
Timing:	1-4 Shifts	2 cycles
	5-8 Shifts	3 cycles
	9-12 Shifts	4 cycles
	13-16 Shifts	5 cycles
	17-20 Shifts	6 cycles
	21-24 Shifts	7 cycles
	25-28 Shifts	8 cycles
	29-32 Shifts	9 cycles
	33-36 Shifts	10 cycles
	37-40 Shifts	11 cycles
	41-44 Shifts	12 cycles
	45-46 Shifts	13 cycles
Indicators:		None
Registers Affected:		A-Accumulator, B-Accumulator

FRL 00-14

FULL-LEFT ROTATE ACCUMULATORS



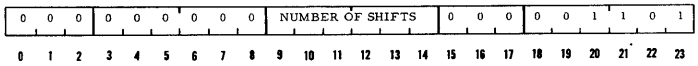
Bits 0-23 of the A- and B-Accumulators are rotated left n places as specified by the code in bit positions 9-14 of the instruction word. The sign bit (bit 0) in the A-Accumulator is shifted to bit position 23 of the B-Accumulator, and the sign bit of the B-Accumulator is shifted to bit position 23 of the A-Accumulator.



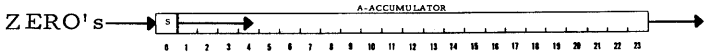
Timing:	1-4 Shifts	2 cycles
	5-8 Shifts	3 cycles
	9-12 Shifts	4 cycles
	13-16 Shifts	5 cycles
	17-20 Shifts	6 cycles
	21-24 Shifts	7 cycles
	25-28 Shifts	8 cycles
	29-32 Shifts	9 cycles
	33-36 Shifts	10 cycles
	37-40 Shifts	11 cycles
	41-44 Shifts	12 cycles
	45-48 Shifts	13 cycles
Indicators:		None
Registers Affected		A-Accumulator, B-Accumulator

RSL 00-15

RIGHT SHIFT A-ACCUMULATOR LOGICAL



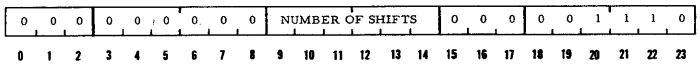
Bits 0-23 of the A-Accumulator are shifted right n places as specified by the code in bit positions 9-14 of the instruction word. ZERO's are entered in bit position zero to replace the most significant bits as the least significant bits are shifted off bit position 23.



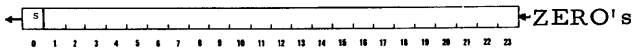
Timing:	1-4 Shifts	2 cycles
	5-8 Shifts	3 cycles
	9-12 Shifts	4 cycles
	13-16 Shifts	5 cycles
	17-20 Shifts	6 cycles
	21-24 Shifts	7 cycles
Indicators:		None
Registers Affected:		A-Accumulator

LSL 00-16

LEFT SHIFT A-ACCUMULATOR LOGICAL



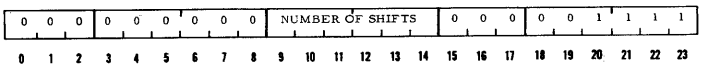
Bits 0-23 of the A-Accumulator are shifted left n places as specified by the code in bits 9-14 of the instruction word. ZERO's are entered into bit position 23 to replace the least significant bits as the most significant bits are shifted off bit position 0.



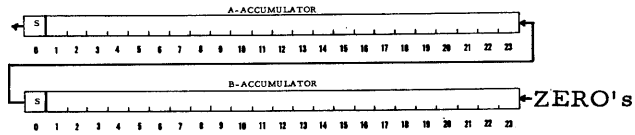
Timing:	1-4 Shifts	2 cycles
	5-8 Shifts	3 cycles
	9-12 Shifts	4 cycles
	13-16 Shifts	5 cycles
	17-20 Shifts	6 cycles
	21-24 Shifts	7 cycles
Indicators:		None
Registers Affected:		A-Accumulator

FLL 00-17

FULL-LEFT SHIFT, LOGICAL



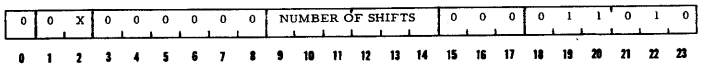
Bits 0-23 of the A- and B-Accumulators are shifted left n places as specified by the code in bit positions 9-14 of the instruction word. The sign bit (bit 0) of the B-Accumulator is shifted to bit position 23 of the A-Accumulator. The most significant bits of the A-Accumulator are shifted off bit position 23 while ZERO's replace the least significant bits of the B-Accumulator.



Timing:	1-4 Shifts	2 cycles
	5-8 Shifts	3 cycles
	9-12 Shifts	4 cycles
	13-16 Shifts	5 cycles
	17-20 Shifts	6 cycles
	21-24 Shifts	7 cycles
	25-28 Shifts	8 cycles
	29-32 Shifts	9 cycles
	33-36 Shifts	10 cycles
	37-40 Shifts	11 cycles
	41-44 Shifts	12 cycles
	45-48 Shifts	13 cycles
Indicators:		None
Registers Affected:		A-Accumulator, B-Accumulator

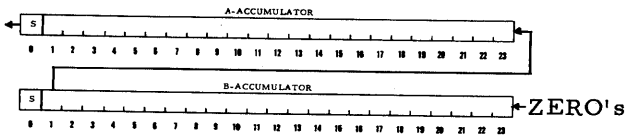
NOR 00-32

NORMALIZE ACCUMULATORS



Bits 0-23 of the A-Accumulator and 1-23 of the B-Accumulator are shifted left until the bit occupying bit position 1 of the A-Accumulator differs from the

A-Accumulator sign bit (bit position 0). ZERO's are shifted into bit position 23 of the B-Accumulator as the most significant bits are shifted off bit position 1 to bit position 23 of the A-Accumulator. The most significant bits of the A-Accumulator are shifted off bit position 0.



NOTES

Note 1: Bit positions 9-14 are coded to specify the maximum number of permissible shifts. The instruction execution is completed upon normalization of the operand, or upon reaching the maximum number of shifts as specified by bits 9-14 (whichever occurs first).

Note 2: The second index bit (bit 2) may be used to specify index register 1 as a shift counter. If the value of bit 2 is ONE, the contents of index register 1 will be decremented each time a shift is performed. This allows the creation of the binary exponent in the index register.

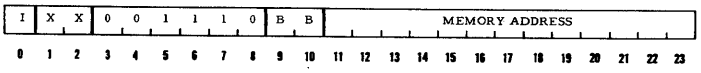
Timing:	1-3 Shifts	2 cycles
	4-6 Shifts	3 cycles
	7-9 Shifts	4 cycles
	10-12 Shifts	5 cycles
	13-15 Shifts	6 cycles
	16-18 Shifts	7 cycles
	19-21 Shifts	8 cycles
	22-24 Shifts	9 cycles
	25-27 Shifts	10 cycles
	28-30 Shifts	11 cycles
	31-33 Shifts	12 cycles
	34-36 Shifts	13 cycles
	37-39 Shifts	14 cycles
	40-42 Shifts	15 cycles
	43-45 Shifts	16 cycles
Indicators:		None
Registers Affected:		A-Accumulator, B-Accumulator

CONTROL INSTRUCTIONS

The seven instructions in this group are used for the general housekeeping functions required by the program. The HLT (halt) instruction stops the computer after loading the next sequential instruction into the instruction register. The NOP (no operation) instruction simply reserves a program slot for a future addition or delays the program to match a real-time input or output rate. The PIE, PID, and EXI (priority interrupt enable/disable and external interrupt) instructions allow program control of priority interrupts. The EXU (execute) instruction allows the execution of an instruction out of the normal program sequence without changing the program counter. The TAC (test and activate) instruction allows the initialization of SM-I/OP's under program control.

EXU 16

EXECUTE INSTRUCTION AT LOCATION



The instruction located at the effective memory address is executed.

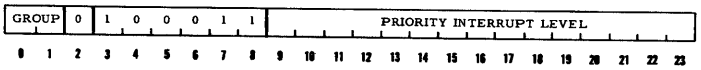
NOTE

This instruction allows the use of instructions out of the normal program counter sequence without changing the program count itself. In addition, the effective address may be modified through the index and indirect flags so that different instructions are executed each time the same EXU instructions occur in a program loop.

Timing:	1 cycle plus the time for the executed instruction
Indicators:	None
Register Affected:	Dependent on the executed instruction

PID 0.43

PRIORITY INTERRUPT DISABLE



This instruction disables any combination of the 15 priority interrupt levels belonging to the priority interrupt group (GRP) selected by the code in bit positions 0 and 1. Bits 23 through 9 are set to ONE's to disable interrupt levels 1 through 15.

NOTE

This instruction allows a selected number of priority interrupt channels to be disabled. The companion PIE

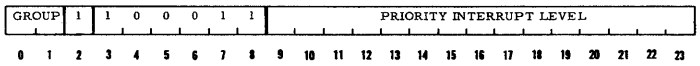
NOTE (Cont'd)

instruction allows any of the same channels to be enabled for operation.

Timing: 1 cycle
Indicators: None
Registers Affected: None

PIE 1.43

PRIORITY INTERRUPT ENABLE



This instruction enables any combination of the 15 priority interrupt levels belonging to the priority interrupt group (GRP) specified by the code in bit positions 0 and 1. Bits 23 through 9 are set to ONE's to enable interrupt levels 1 through 15.

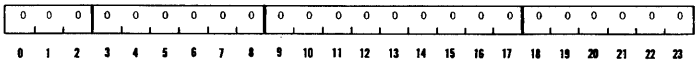
NOTE

See PID instruction.

Timing: 1 cycle
Indicators: None
Registers Affected: None

HLT 00-00

HALT



This instruction halts the operation of the central processor.

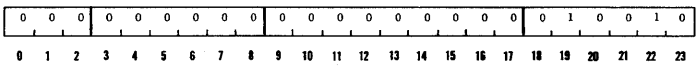
NOTE

The central processor stops with the address of the next instruction in the program counter and the next instruction in the instruction register. The START switch is closed to initiate the I cycle of that addressed instruction. The program counter may be manually reset and set to a new starting address prior to the closing of the START switch.

Timing: 1 cycle
Indicators: Halt
Registers Affected: None

NOP 00-22

NO OPERATION



No operation is performed.

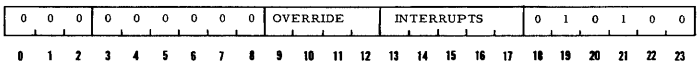
NOTE

This instruction is used to reserve memory locations for instructions to be added within the program encompassing that memory location. It may also be used to delay a program to match a real-time input or output transfer rate.

Timing: 1 cycle
Indicators: None
Registers Affected: None

EXI 00-24

EXTERNAL INTERRUPT

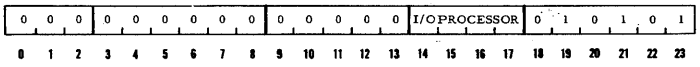


This instruction enables one central processor to interrupt another. Up to six interrupts may be generated. These may be connected to interrupt levels in another CP and used as a means of control communication. Three override levels permit one CP to initiate execution in another CP.

Timing: 1 cycle
Indicators: None
Registers Affected: None

TAC 00-25

TEST AND ACTIVATE



The execution of this instruction causes the specified SM-I/OP to execute the CEU or TEU instruction stored in the dedicated memory location assigned to the SM-I/OP (see table 4-2). The instruction following the TAC is either executed or skipped, depending upon the execution of the CEU or TEU instruction by the SM-I/OP. The skip logic is exactly the same for SM-I/OP execution as for CP-I/OP execution. Therefore, see the CEU and TEU instruction descriptions to determine the skip condition.

Bit positions 17 through 14 are used to select I/OP's 1 through 4, respectively.

NOTE

No attempt should be made to initialize more than one I/OP with a single TAC instruction.

Timing:	2 cycles
Indicators:	I/O hold (input/output processor panel) in the event of a peripheral malfunction or an I/O programming error.

INPUT/OUTPUT INSTRUCTIONS

Six instructions are provided to perform data I/O and external unit control. The basic operation of these instructions is described herein. These instructions are:

Command External Unit (CEU)

Test External Unit (TEU)

Accumulator Word Output to Peripheral (AOP)

Memory Word Output to Peripheral (MOP)

Accumulator Word Input from Peripheral (AIP)

Memory Word Input from Peripheral (MIP)

Two instructions, A input (AIP) and A output (AOP) are provided to enable words or characters to be transferred between the A-Accumulator and peripheral devices. These instructions provide a convenient character assembly/disassembly capability. Each of these instructions occupies a single memory location. The two instructions, memory input (MIP) and memory output (MOP), enable words or characters to be transferred directly between specified memory locations and peripheral devices. The instruction command external unit (CEU) enables all system units connected to the computer to be controlled by the program. The CEU instruction is used to initiate BTC channels as well as to control computer peripheral devices and special system units. The test external unit (TEU) instruction is provided to enable system units and peripheral devices to be tested by the computer. The test result causes the instruction following the TEU to be either executed or skipped. The CEU and TEU instructions can be executed by the shared memory I/O processor as well as by the central processor. Two memory locations are required to store the MIP, MOP, CEU, and TEU instructions.

Data or command word transfer instructions may be executed in either of two modes - wait or skip- as defined in the following paragraphs.

WAIT MODE

In this mode, the transfer is not made until the peripheral device signifies that it is ready. The processor continues to test for the ready indication each machine cycle. Execution of the transfer is made during the first cycle after the peripheral device sends a ready signal to the processor. After the transfer, the unit is disconnected and the next instruction in sequence is executed. The specific meaning of the ready signal is defined in each I/O instruction description.

SKIP MODE

In this mode, the ready status of the peripheral unit is tested only once. If the unit is ready, the transfer is executed. The program counter is then advanced by two, which causes the next instruction in sequence to be skipped. If the peripheral unit is not ready, the unit is disconnected from the I/O bus and the program counter is advanced by ONE. This conditional skip feature enables all I/O instructions (except TEU) to perform the following sequence of operations:

a. Connect the device

b. Test for ready

c. Transfer if ready

d. Disconnect the device

The flow chart showing the execution of the AIP and AOP instructions is shown in figure 2-7. As shown in the flow chart, the state of the wait flag determines whether the instruction is executed in the wait or skip mode. The MIP, MOP, and CEU instructions are executed in the same manner, except that the program counter is advanced by one before the transfer is made in order to obtain the operand address.

Execution of the TEU instruction requires no ready test command. An on-line unit is always ready to be tested. The test word is always transferred to the unit and a test return signal is tested. The result of the test is conditional skip of the next instruction. In addition to providing selectable execution modes, the two-word I/O instructions (MIP, MOP, CEU, and TEU) provide two selectable operand addressing modes - immediate mode and address mode.

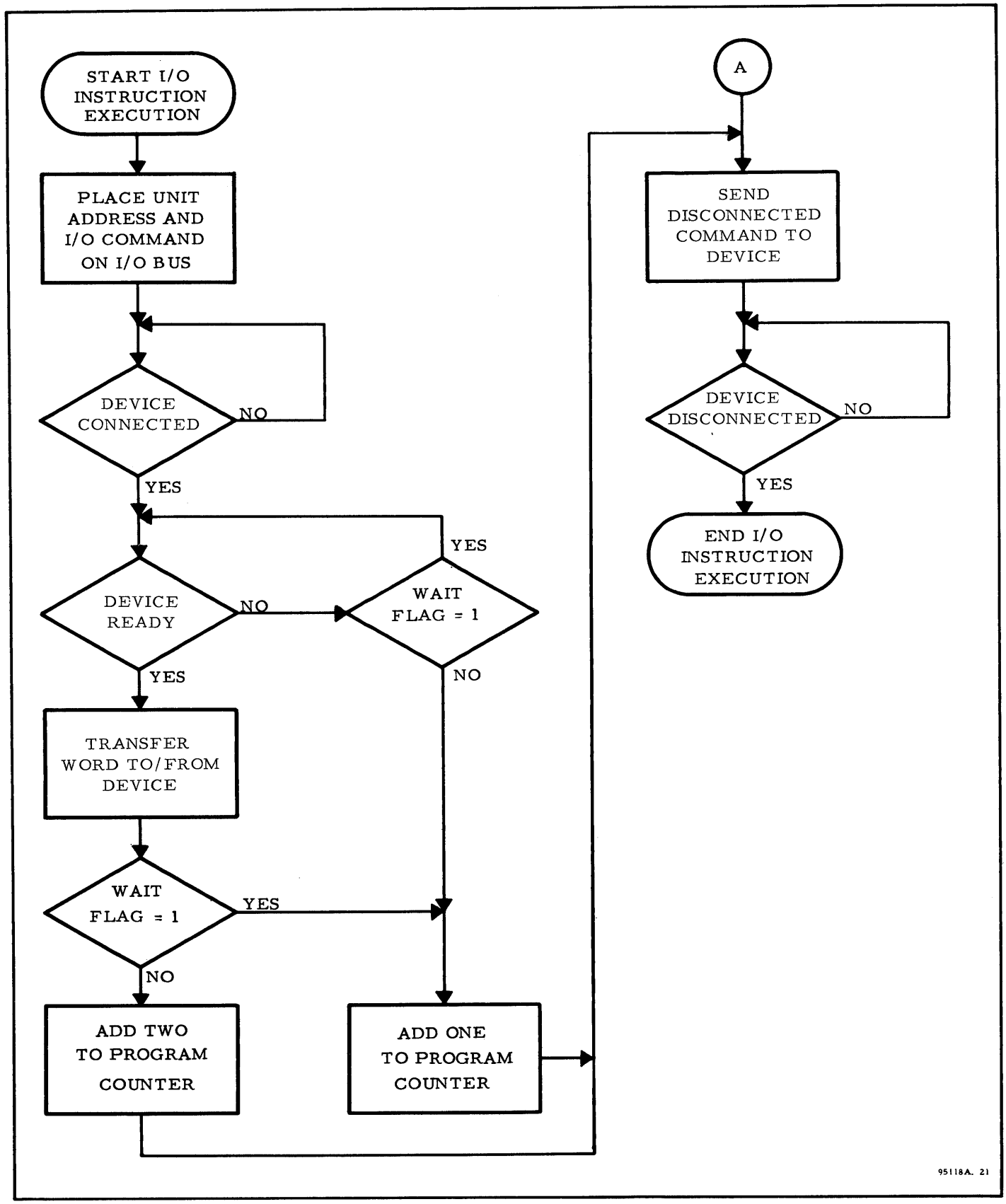


Figure 2-7. AIP/AOP Flow Chart

Immediate Mode

In this mode, the instruction's second word is treated as the operand. In executing MOP, CEU, and TEU instructions, the contents of the instruction's second location are transferred to the specified unit. MIP execution consists of transferring a word or character from a specified unit into the instruction's second location.

Address Mode

In this mode, the instruction's second word is interpreted as the operand address. The indirect address format is used in the instruction's second word. Therefore, indexing and indirect address chaining may be used in addressing the operand virtual address.

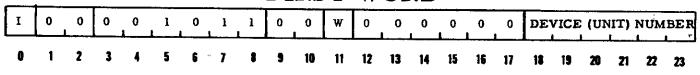
The addressing mode is specified in the first instruction word by the value of the indirect address flag (I). If I is a ONE, the address mode is executed; if I is a ZERO, the immediate mode is executed.

The format of the I/O instruction words is shown in figure 2-8. The definitions of the bits are listed in table 2-2. The specific coding used in each instruction is defined in the individual instruction descriptions.

CEU 130 131 (WAIT)

COMMAND EXTERNAL UNIT

FIRST WORD



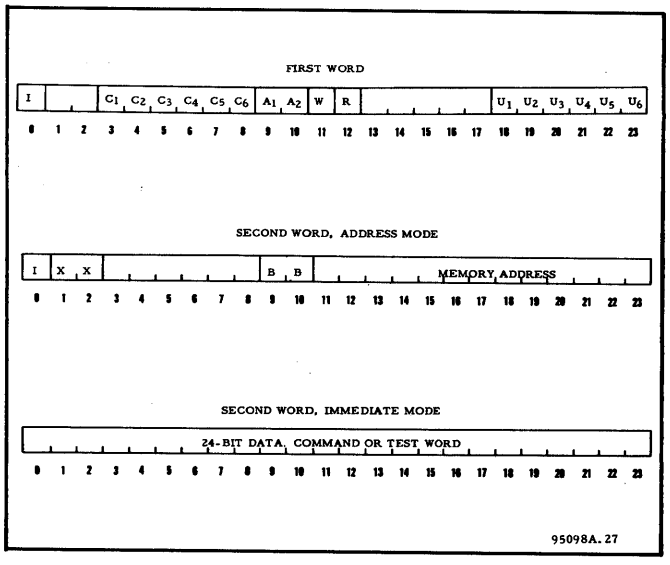
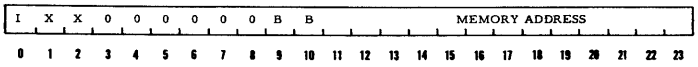


Figure 2-8. I/O Instruction Word Format

Table 2-2. I/O Instruction Bit Definitions

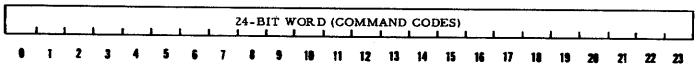
Symbol	Definition	Contained In
C	Command Code	All I/O Instructions
R	Character Merge Flag	AIP
I	Indirect Address Flag	MIP, MOP, CEU, TEU
A	Augmented Command Code	All I/O Instructions
W	Wait Flag	All I/O Instructions except TEU
U	Unit Number ($00-77_8$)	All I/O Instructions
X	Index Bits	MIP, MOP, CEU, TEU
B	BAR Select Bits	MIP, MOP, CEU, TEU

SECOND WORD, ADDRESS MODE



Or

SECOND WORD, IMMEDIATE MODE



Address Mode - Transfers the command(s) contained in the effective memory address to the unit designated by the code in bit positions 18-23.

Immediate Mode - Transfers the command(s) contained in the instruction's second word location to the unit specified by the code in bit positions 18-23 of the first word.

NOTE

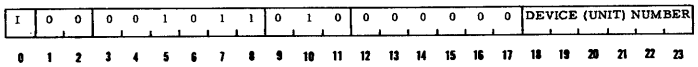
The bits in most unit command codes are micro-programmed. Thus, either one or several function commands may be transferred to a unit by execution of a single CEU instruction. (See Section VI and Appendix C for the definition of the standard unit command codes.)

Operand Address Modes:	Address - I = ONE, Immediate - I = ZERO (first word)
Execution Modes:	Wait - W = ONE Skip - W = ZERO
Transfer Criterion:	A unit indicates a ready to a CEU test if the unit can immediately begin execution of any new function command.
Timing:	3 cycles + wait
Indicators:	I/O HOLD if, for any reason, the processor must wait for the peripheral unit.
Registers Affected:	None

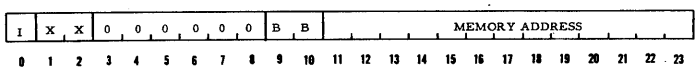
TEU 132

TEST EXTERNAL UNIT

First Word

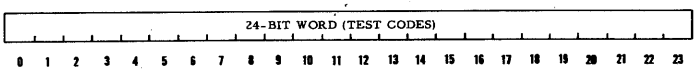


Second Word, Address Mode



Or

Second Word, Immediate Mode



Address Mode - Transfers the test code(s) contained in the effective memory address to the unit specified by the code in bit positions 18-23.

Immediate Mode - Transfers the test code(s) contained in the instructions second word location to the unit specified by the code in bit positions 18-23.

NOTE

The bits in most unit test codes are micro-programmed. Therefore, either one or several function codes may be transferred to a unit by the execution of a single TEU instruction. (See Section VI and Appendix C for the definition of the standard unit test codes.)

Operand Address Modes:	Address - I = ONE Immediate - I = ZERO (first word)
Execution Modes:	TEU is always executed in the same mode. An on-line peripheral unit is always ready to accept a test code. The wait flag feature is not used.

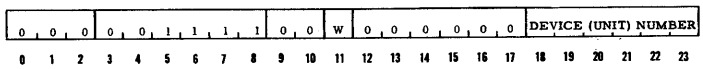
NOTE

When the return signal from the unit is tested by the processor, the program counter is advanced by one or two locations depending on the result of the test. If the test result is true, the program counter is advanced by two locations causing the next sequential instruction to be skipped. If the result is false, the next sequential instruction is executed.

Timing:	3 cycles
Indicators:	None
Registers Affected:	None

AOP 170 171 (WAIT)

ACCUMULATOR WORD OUTPUT TO PERIPHERAL

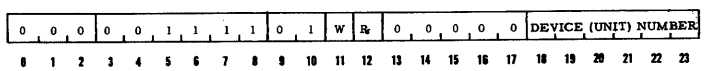


Transfers a word from the A-Accumulator to the unit specified by the code in bit positions 18-23. Character-oriented units accept only bits 0-7 of the A-Accumulator word.

Execution Modes:	Wait - W = ONE, Skip - W = ZERO
Transfer Criterion:	A unit indicates a ready to an AOP test if the unit can immediately accept a new word or character.
Timing:	3 cycles + wait
Indicators:	I/O HOLD, if, for any reason, the processor must wait for the peripheral unit.
Registers Affected:	None

AIP NO MERGE 1720 (WAIT) 1730 MERGE 1724 (WAIT) 1734

ACCUMULATOR WORD INPUT FROM PERIPHERAL



The peripheral unit specified by the code in bit positions 18-23 transfers a word or character to the A-Accumulator. Character-oriented units

transfer bits into bit positions 16-23 of the A-Accumulator.

NOTE

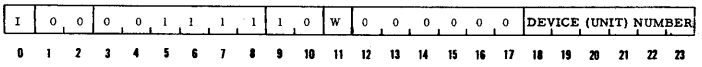
This instruction includes a feature for character assembly in the A-Accumulator. If the merge flag (R) is a ONE, bits 0-15 of the A-Accumulator remain unchanged by the execution of an AIP instruction. Therefore, an 8-bit character can be read into bit positions 16-23, and shifted left eight bit positions prior to reading the next character. If R = ZERO, the A-Accumulator is cleared prior to the reading in of a new character or word.

Execution Modes:	Wait - W = ONE Skip - W = ZERO
Transfer Criterion:	A unit indicates a ready to an AIP test if the unit has a word or character ready for immediate transfer.
Timing:	3 cycles + wait
Indicators:	I/O HOLD if, for any reason, the processor must wait for the peripheral unit.
Registers Affected:	None

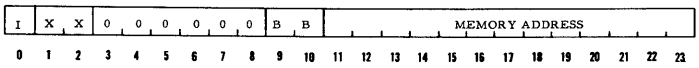
MOP 174 175 (WAIT)

MEMORY WORD OUTPUT TO PERIPHERAL

First Word

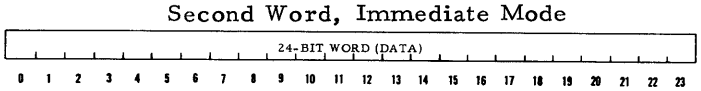


Second Word, Address Mode



Or

Second Word, Immediate Mode



Address Mode - Transfers a word from the effective memory address to the unit designated by the code in bit positions 18-23. Character-oriented units accept only bits 0-7 of the memory word.

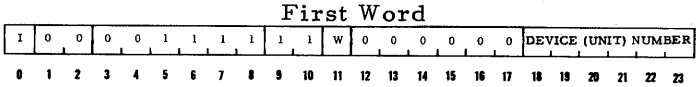
Immediate Mode - Transfers a word from the instruction's second word location to the unit specified by the code in bit positions 18-23. Character-oriented units accept only bits 0-7.

Operand Address Modes:	Address - I = ONE Immediate - I = ZERO (first word)
Execution Modes:	Wait - W = ONE, Skip - W = ZERO
Transfer Criterion:	A unit indicates a ready to an MOP test if the unit can immediately accept a new word or character.
Timing:	3 cycles + wait
Indicators:	I/O HOLD if, for any reason, the processor must wait for the peripheral unit.
Registers Affected:	None

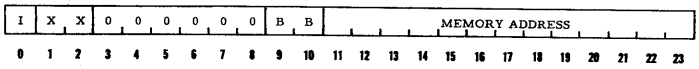
MIP 176 177 (WAIT)

MEMORY WORD INPUT FROM PERIPHERAL

First Word

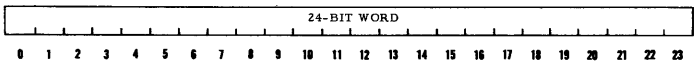


Second Word, Address Mode



Or

Second Word, Immediate Mode



Address Mode - Transfers a word or character from the unit designated by the code in bit positions 18-23 to the effective memory address. Character-oriented units transfer data into bits 16-23 of the effective memory address.

Immediate Mode - Transfers a word from the unit designated by bits 18-23 (first word) into the instruction's second word location. Character-oriented units transfer into bits 16-23 of the memory location.

Operand Address:	Address - I = ONE
Modes:	Immediate - I = ZERO
Execution Modes:	Wait - W = ONE Skip - W = ZERO
Transfer Criterion:	A unit indicates a ready to an MIP test if the unit has a word or character ready for immediate transfer.
Timing:	3 cycles + wait
Indicators:	I/O HOLD if, for any reason, the processor must wait for the peripheral unit.

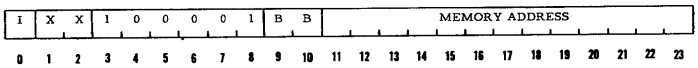
Registers Affected: None

EAU INSTRUCTIONS

This group of 23 instructions applies only to the optional extended arithmetic unit (EAU). Two of the instructions (EDP and EFP) are used to set the EAU operating mode to double-precision, fixed-point or normalized floating-point. Four other instructions (EAD, ESU, EMU and EDV) allow addition, subtraction, multiplication, and division to be performed by the EAU. Three instructions (ELO, ELB, and ELN) are provided for loading the EAU arithmetic registers (EA- and EB-Accumulators) and one (EST) for storing the contents of the EA-Accumulator in memory. Six instructions are skip instructions. Instructions are also provided for manipulating data words in the EAU by transferring words between the EA- and EB-Accumulators (EIA, EAB, EBA), clearing (ECA and ECB) both accumulators and the normalization (ENO) and un-normalize (EUN) of the EA- and EB-Accumulator contents.

ELB 41

EXTENDED LOAD EB-ACCUMULATOR



Fixed-Point, Double-Precision Mode - The contents of the effective memory address and the next sequential memory address replace the previous contents of the EB-Accumulator. The first memory word is loaded into the most significant 24 bit positions of the EB-Accumulator. The contents of the second location replace the least significant 23 bits of the EB-Accumulator; the sign bit of the second memory location is ignored.

Floating-Point Mode - The 38-bit mantissa is loaded into the 38 most significant bit positions of the EB-Accumulator, and the 9-bit exponent is loaded into the EAU exponent register. The remaining 9-bit positions of the EB-Accumulator are set to ZERO.

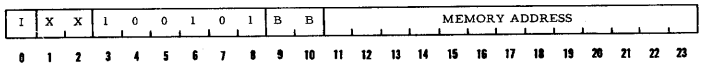
NOTE

In floating-point operations, the 38-bit mantissa is formed by the 24-bit contents of the effective memory address and the 14 most significant bits of the next sequential memory location excluding sign.

Timing:	3 cycles
Indicators:	None
Registers Affected:	EB-Accumulator, exponent registers

EAD 45

EXTENDED ARITHMETIC ADD



Fixed-Point, Double-Precision Mode - The contents of the effective memory address and the contents of the next sequential memory address are added to the total contents of the EA-Accumulator. The sum is placed in the EA-Accumulator.

Floating-Point Mode - The exponents of the augend and addend are compared. If they differ by 38 bits (the size of the mantissa) or more, the larger operand is placed/remains in the EA-Accumulator and the instruction is terminated. If the differences are less than 38 bits, the smaller operand is shifted right until its exponent equals the exponent of the larger operand. The mantissa of the two operands are then algebraically added. The mantissa of the sum is placed in the 38 most significant bit positions of the EA-Accumulator with the remaining 9-bit positions set to ZERO. The exponent remains in the exponent register.

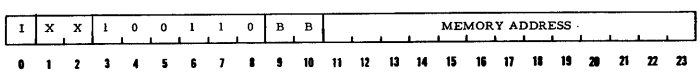
NOTE

The augend must be properly positioned in the EA-Accumulator prior to the EAD instruction. This may be accomplished by an ELO instruction or as the result of a preceding arithmetic operation. The addend is placed in the ET-Register.

Timing:	
Double-precision, Fixed point	3 cycles
Floating-point	3 cycles + 1 cycle for each eight shifts required to renormalize and shift exponents.
Indicators:	OVERFLOW (EAU Panel), if the sum exceeds 46 bits plus sign (double-precision); or if the exponent exceeds eight bits plus sign (floating-point).
Registers Affected:	EA-Accumulator, exponent registers

ESU 46

EXTENDED ARITHMETIC SUBTRACT



Fixed-point, Double-precision Mode - The contents of the effective memory address and the contents of the next sequential memory address are algebraically subtracted from the contents of the EA-Accumulator. The difference is placed in the EA-Accumulator.

Floating-point Mode - After the subtrahend is complemented, the exponents of the minuend and subtrahend are compared. If they differ by 38 bits (the size of the mantissa) or more, the larger operand is placed/remains in the EA-Accumulator and the instruction is terminated. If the exponents differ by less than 38 bits, the smaller operand is shifted right until both exponents are equal. The mantissas are then algebraically subtracted. The mantissa of the difference is placed in the 38 most significant bit positions of the EA-Accumulator. The remaining 9-bit positions of the EA-Accumulator are set to ZERO. The exponent remains in the exponent register.

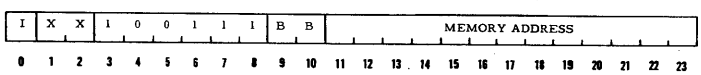
NOTE

The minuend must be properly positioned in the EA-Accumulator prior to the ESU instruction. This may be accomplished by an ELO instruction or as a result of a preceding arithmetic operation. The addend is replaced in the ET-Register.

Timing:	
Double-precision	3 cycles
Floating-point	3 cycles + 1 cycle for each eight shifts required to renormalize and shift exponents.
Indicators:	OVERFLOW (EAU Panel), if the difference exceeds 46 bits plus sign (double-precision); or if the exponent exceeds eight bits plus sign (floating-point).
Registers Affected:	EA-Accumulator, exponent registers

EMU 47

EXTENDED MULTIPLY



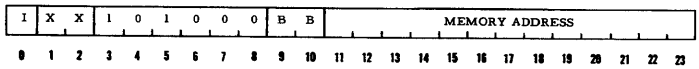
Fixed-point, Double-precision Mode - The contents of the effective memory address and the contents of the next sequential memory location are multiplied by the contents of the EB-Accumulator. The most significant 47 bits of the product are placed in the EA-Accumulator and the least significant 46 bits are placed in the EB-Accumulator. (The EB-Accumulator sign bit, EB bit position 0, is not a part of the product.)

Floating-point Mode - The 38 most significant bits of the double-length contents of the effective memory address and the next sequential memory location are multiplied by the 38 most significant bits of the EB-Accumulator. The 9-bit exponent from the second memory word and from the EAU exponent register are added. The product mantissa is placed in the 38 most significant bits of the EA-Accumulator, and the remaining 9-bits of the EA-Accumulator and all the EB-Accumulator are set to ZERO. The product exponent is stored in the exponent register.

Timing:	9 cycles in both double-precision and floating-point modes.
Indicators:	OVERFLOW (EAU Panel), if the exponent sum exceeds 8-bits plus sign (floating-point); or if both operands are full-scale negative quantities (double-precision)
Registers Affected:	EA-Accumulator, EB-Accumulator, exponent registers

EDV 50

EXTENDED DIVIDE



Fixed-point, Double-precision Mode - The 24-bit contents of the effective memory address and the 23-bit contents of the next sequential memory location are divided into the 93-bit contents of the EA- and EB-Accumulators. The 47-bit quotient is placed in the EA-Accumulator and the 47-bit remainder is placed in the EB-Accumulator.

NOTE

When the double-length division is called from memory, the sign bit of the second word is ignored, thereby creating a 47-bit quantity. When forming the double-length dividend, the EB-Accumulator sign bit is ignored, thus creating a 93-bit quantity.

Floating-point Mode - The 38 most significant bits of the double-length divisor are divided into the 38 most significant bits of the EA-Accumulator contents. The 9-bit divisor exponent is subtracted from the 9-bit dividend exponent. The quotient

mantissa is placed in the 38 most significant bit positions of the EA-Accumulator and the quotient exponent is placed in the exponent register. There is no remainder.

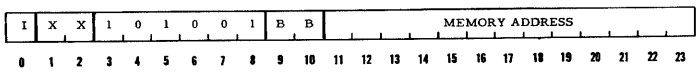
NOTE

The 38-bit divisor mantissa is formed from bits 0-23 of the effective memory address and bits 1-14 of the next sequential memory location. Bits 15-23 of the second word form the divisor exponent. The sign bit (bit 0) of the second memory word is ignored.

Timing:	
Double-precision	15 cycles
Floating-point	14 cycles
Indicators:	OVERFLOW (EAU Panel) In double-precision, fixed-point, if the quotient exceeds the precision of the machine a. |ET| < |EA| b. EA > 0, ET < 0 |ET| = |EA| EB ≠ 0 c. EA < 0, ET < 0 |EA| = |ET| EB = 0 d. EA > 0, ET > 0 |EA| = |ET| In double-precision, floating-point, if the exponent difference exceeds eight bits plus sign, if exponent overflow occurs on the first cycle right shift, or if underflow occurs when normalizing the quotient.
Registers Affected:	EA-Accumulator, EB-Accumulator, Exponent registers

ELN 51

EXTENDED LOAD NEGATIVE



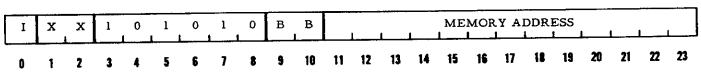
Fixed-point, Double-precision Mode - The contents of the effective memory address and the next sequential memory location are two's complemented and replace the previous contents of the EA-Accumulator. The first memory word is loaded into the most significant 24-bit positions of the EA-Accumulator, and the second memory word is loaded into the least significant 23-bit positions of the EA-Accumulator, the sign bit of the second memory word is ignored.

Floating-point Mode - The contents of the effective memory address and bit positions 1-14 of the next sequential memory address are two's complemented and replace the previous contents of the 38 most significant bits of the EA-Accumulator. The least significant nine bits (15-23) of the second memory word replace the previous contents of the exponent register. The least significant nine bits of the EA-Accumulator are loaded with ZERO's.

Timing:	3 cycles
Indicators:	OVERFLOW (EAU Panel), if the number loaded into EA is full-scale negative and the number loaded into the exponent register is full-scale positive.
Registers Affected:	EA-Accumulator, exponent registers

ELO 52

EXTENDED LOAD



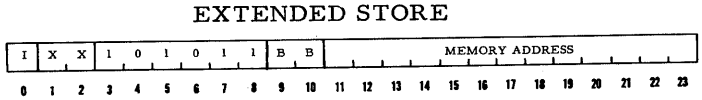
Fixed-point, Double-precision Mode - The contents of the effective memory address and the next sequential memory location replace the previous contents of the EA-Accumulator. The first memory word is loaded into the 24 most significant bit positions of the EA-Accumulator and the second memory word, excluding sign, is loaded into the 23 least significant bit positions.

Floating-point Mode - The 38-bit mantissa is loaded into the 38 most significant bit positions of the EA-Accumulator and the nine-bit exponent is loaded into the exponent register. The remaining nine-bit positions of the EA-Accumulator are loaded with ZERO's. The 38-bit mantissa is formed by the first memory word and by the contents of bit positions 1-14 of the second memory word. The sign bit (bit position 0) of the second memory word is ignored. The exponent is formed by the remaining 9 bits (15-23) of the second memory word.

Timing:	3 cycles
Indicators:	None
Registers Affected:	EA-Accumulator, exponent registers

EST 53

EXTENDED STORE



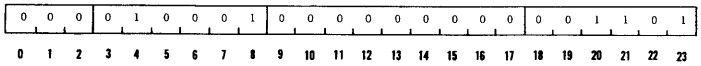
Fixed-point, Double-precision Mode - The contents of the EA-Accumulator replace the previous contents of the effective memory address and the next sequential memory location. The 24 most significant bits of the EA-Accumulator contents are transferred to the effective memory address, and the 23 least significant bits are transferred to bit positions 1-23 of the next sequential memory location. A ZERO is loaded into the sign bit (position 0) of the second memory word.

Floating-point Mode - The 38 most significant bits of the EA-Accumulator and the contents of the exponent register replace the previous contents of the effective memory address and the next sequential memory location. The 24 most significant bits of the EA-Accumulator are stored in the effective memory address. The 14 most significant bits of the EA-Accumulator are stored in bit positions 1-14 of the second memory word. The nine-bit contents of the exponent register are stored in the remaining bits (15-23) of the second memory word. The sign bit (bit position 0) of the second memory word is set to ZERO.

Timing:	3 cycles
Indicators:	None
Registers Affected:	None

ECA 21-15

CLEAR EA-ACCUMULATOR

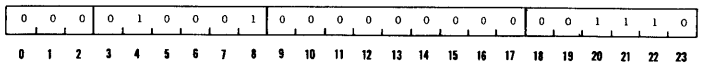


The contents of the EA-Accumulator are destroyed and replaced with all ZERO's. The contents of the nine-bit exponent register are also destroyed and are replaced with the full-scale negative quantity (that is, a ONE in the sign bit and eight ZERO's).

Timing:	1 cycle
Indicators:	None
Registers Affected:	EA-Accumulator, exponent register

ECB 21-16

CLEAR EB-ACCUMULATOR

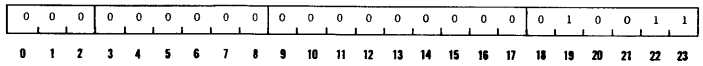


The contents of the EB-Accumulator are destroyed and replaced with all ZERO's.

Timing:	1 cycle
Indicators:	None
Registers Affected:	EB-Accumulator

ESR 00-23

EXTENDED SKIP IF READY

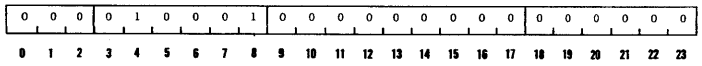


If the EAU is not executing a previous instruction and is operative, the next instruction (NI) is skipped and the second sequential instruction (NI+1) is executed. If the EAU is active or inoperative, the next instruction (NI) is executed.

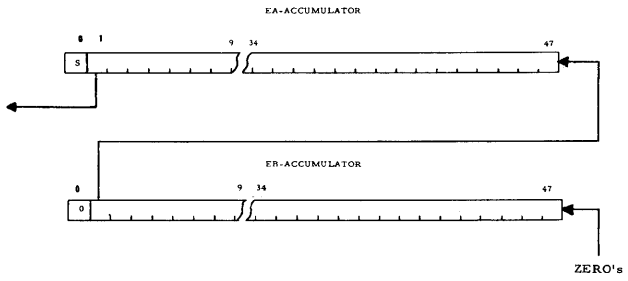
Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter

ENO 21-00

EXTENDED NORMALIZE

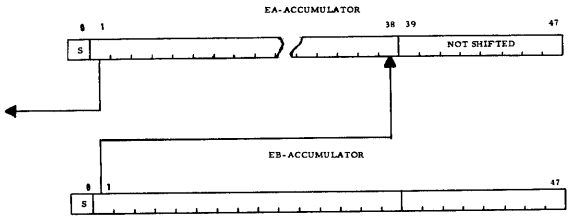


Fixed-point, Double-precision Mode - The contents of the EA- and EB-Accumulators are shifted left together until the most significant bit (bit position 1) of the EA-Accumulator differs from the EA sign bit (bit position 0). The 38 most significant bits of the EA-Accumulator then form the mantissa of the normalized word. The number of shifts required to normalize the contents of the accumulators is counted by the nine-bit exponent register which then holds the exponent at the conclusion of the normalize operation. As the contents of the accumulators are shifted left, ZERO's are entered in the least significant bit position of the EB-Accumulator. In the fixed-point, double-precision mode, the exponent register is cleared at the start of an ENO instruction.



Floating-point Mode - The most significant 38 bits of the EA- and EB-Accumulators are shifted left until the most significant bit (bit position 1) of EA differs from EA sign (bit position 0). As the EA-Accumulator bits are shifted left, bits are entered at bit position 38 from EB, bit 1. The exponent register is used to count the number of shifts and to hold the nine-bit exponent at the conclusion of

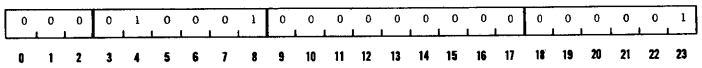
the normalize operation. In the floating-point mode, the exponent register is not cleared at the start of an ENO instruction.



Timing:	1 cycle + cycle for each eight shifts required.
Indicators:	OVERFLOW (EAU Panel) - exponent register overflow (floating-point only)
Registers Affected:	EA-Accumulator, EB-Accumulator, exponent registers

EIA 21-01

INTERCHANGE EA-ACCUMULATOR AND EB-ACCUMULATOR

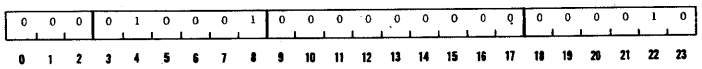


The contents of the EA-Accumulator replace the previous contents of the EB-Accumulator and the contents of the EB-Accumulator replace the previous contents of the EA-Accumulator.

Timing:	1 cycle
Indicators:	None
Registers Affected:	EA-Accumulator, EB-Accumulator

EBA 21-02

TRANSFER EB-ACCUMULATOR TO EA-ACCUMULATOR



The contents of the EB-Accumulator replace the previous contents of the EA-Accumulator. The contents of the EB-Accumulator are unchanged.

Timing:	1 cycle
Indicators:	None
Registers Affected:	EA-Accumulator

EAB 21-03

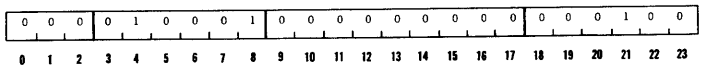
TRANSFER EA-ACCUMULATOR TO EB-ACCUMULATOR

The contents of the EA-Accumulator replace the previous contents of the EB-Accumulator. The contents of the EA-Accumulator are unchanged.

Timing:	1 cycle
Indicators:	None
Registers Affected:	EB-Accumulator

EUN 21-04

EXTENDED UNNORMALIZE



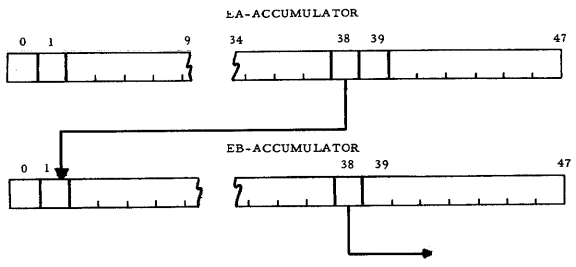
If the EAU is in the floating-point mode and the nine-bit exponent register contains a negative quantity, the contents of the 38 most significant bit positions of the EA- and EB-Accumulators are shifted right until the contents of the exponent register equals zero.

NOTES

Note 1: If the exponent register contents are more negative than -73, the EA- and EB-Accumulators will be cleared and the exponent register contents will be set to full-scale (that is, a ONE in the sign bit and eight ZERO's).

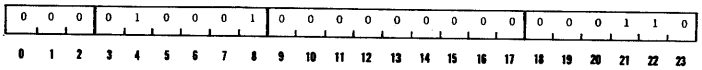
Note 2: If the EAU is in the double-precision mode or if the contents of the exponent register are positive, or zero, the instruction will be terminated.

Timing:	1 cycle + 1 cycle for each eight shifts required.
Indicators:	None
Registers Affected:	EA-Accumulator, EB-Accumulator, exponent register



ESZ 21-05

EXTENDED SKIP IF ZERO

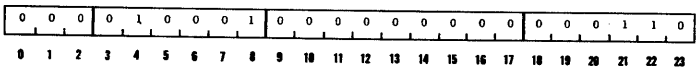


If the contents of the EA-Accumulator equal zero, the next instruction (NI) is skipped and the second sequential instruction (NI+1) is executed. If the contents of the EA-Accumulator are greater or less than zero, the next instruction (NI) is executed.

Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter

EPS 21-06

EXTENDED SKIP IF POSITIVE

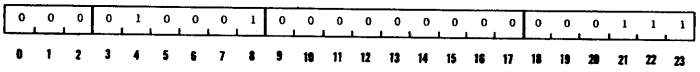


If the contents of the EA-Accumulator are positive, the next instruction (NI) is skipped and the second sequential instruction (NI+1) is executed. If the contents of the EA-Accumulator are negative, the next instruction (NI) is executed.

Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter

ESN 21-07

EXTENDED SKIP IF NEGATIVE

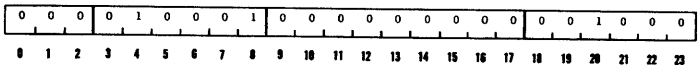


If the contents of the EA-Accumulator are negative the next instruction (NI) is skipped and the second sequential instruction (NI+1) is executed. If the contents of the EA-Accumulator are equal to zero or positive, the next instruction (NI) is executed.

Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter

ESO 21-10

EXTENDED SKIP ON OVERFLOW

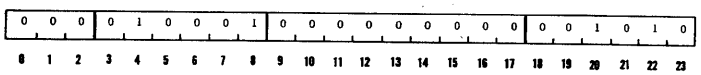


If the EAU overflow latch has been set, the next instruction (NI) is skipped, the second sequential instruction (NI+1) is executed, and the overflow latch is reset. If the overflow latch is reset, the next sequential instruction (NI) is executed.

Timing:	1 cycle
Indicators:	EAU OVERFLOW will be extinguished if previously lit.
Registers Affected:	Program counter

EDP 21-12

EXTENDED DOUBLE-PRECISION FIXED-POINT

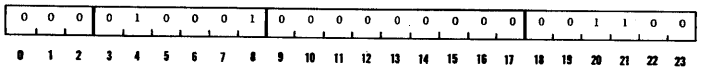


Sets the EAU to the fixed-point, double precision mode. All subsequent inputs will be treated as double-precision data and all subsequent instructions will operate in the double-precision mode.

Timing:	1 cycle
Indicators:	Double-precision (EAU Panel)
Registers Affected:	None

EFP 21-14

EXTENDED FLOATING POINT

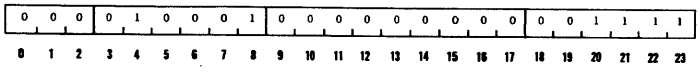


Sets the EAU to the floating-point mode. All subsequent inputs will be treated as floating-point data and all subsequent instructions will operate in the floating-point mode.

Timing:	1 cycle
Indicators:	Floating-point (EAU Panel)
Registers Affected:	None

ESD 21-17

EXTENDED SKIP IF FIXED-POINT DOUBLE-PRECISION



If the EAU is set to the fixed-point, double-precision operating mode, the next instruction is skipped and the second sequential instruction is executed. If the EAU is set to the floating-point operating mode, the next instruction is executed. The next instruction is skipped if EA = 0 or EA > 0.

Timing:	1 cycle
Indicators:	None
Registers Affected:	Program counter

PROGRAM PROTECT INSTRUCTIONS

The optional program protect feature permits an operator to guard areas of memory against

accidental modification or use by another program. This option requires two additional computer instructions which are described below. The use of the program protect bit (PPB) is discussed in detail in Section VII.

PON 62

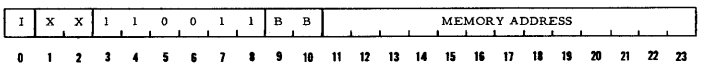
PROTECT BIT ON

Sets the PPB on at the effective memory address.

Timing:	2 cycles
Indicators:	None
Registers Affected:	None

POF 63

PROTECT BIT OFF



Turns the PPB off at the effective memory address.

Timing:	2 cycles
Indicators:	None
Registers Affected:	None

NOTE

The PON and POF instructions function only if the program protect mode console switch is off or, if in the protected mode, their own PPB is on.

SECTION III
ASSEMBLY LANGUAGE PROGRAMMING

GENERAL DESCRIPTION

The general format of the 840MP Computer System symbolic assembly instruction input (source input) consists of five major fields. These fields are the location, operation, address, comments, and identification fields. Figure 3-1 shows an example of a source program written in symbolic assembly language. The following paragraphs describe the coding format.

LOCATION FIELD

The location field (columns 1-4 in figure 3-1) may consist of a symbolic label for the instruction line when it becomes necessary to refer to this location elsewhere in the program. The symbolic label consists of one to four characters; the first character must be a letter and the remaining characters may be either letters or digits. If no further reference to the instruction line is necessary, the location field may be left blank. An asterisk (*) in the first column indicates that the entire instruction line consists of comments only.

OPERATION FIELD

The operation field (columns 6-9 in figure 3-1) consists of a mnemonic computer instruction or pseudo-operation. Lists of mnemonic instructions and pseudo-operations are given in tables 3-2 and 3-4.

Mnemonic instructions consist of three letters. The mnemonic instruction must be left-justified in the operation field, that is, written in columns 6, 7, and 8. If the instruction address is to be made indirect, the three-letter mnemonic instruction is followed by an asterisk in column nine.

Pseudo-operation consists of three or four letters and represents either data definitions or instructions to the assembly program.

ADDRESS FIELD (VARIABLE FIELD)

Memory reference instructions use the variable field to define the operand address; the operand address may be followed by a comma and the digit 1, 2, or 3 to signify the index register to be used for indexing. Certain other instructions, such as shift and I/O operations and certain pseudo-operations, have special formats for the variable field. These special formats are defined in tables 3-1 through 3-3. If no address field definition is required, the address field is left blank.

OPERAND ADDRESS FIELD FORMATS

An operand address may have any of the following formats:

NO ADDRESS

The address field may be left blank if no operand address is required.

SYMBOLIC ADDRESS

The address consists of one to four characters, starting with a letter.

EXTERNAL SYMBOLIC ADDRESS

An external symbolic address consists of a dollar sign ($) followed by one to six characters, the first of which is a letter. This external address is not defined within the program in which it is contained, but refers to a subroutine or item located in a different subprogram or in a program library. No address arithmetic or indexing may be performed on external symbolic addresses.

ABSOLUTE ADDRESS

An absolute address consists of digits only and is presumed to be a decimal number unless preceded by an apostrophe (') in which case the address is octal. This format is used when reference to a fixed memory location is required, or when the address represents a count (that is, the number of shifts specified in a shift operation).

CURRENT LOCATION

Consists of a single asterisk (*) in the address field. This identifies the location of this instruction as the instruction's address. This format allows references to the current or nearby instructions to be made without having to assign a symbolic name. When referencing to nearby addresses, the asterisk must be used in conjunction with address arithmetic (defined in the following paragraphs).

SEL

PROGRAMMER:

PROGRAM:

```
LOC.    OPER.   ADDRESS, INDEX
1       6       11                      25                      50                      72
*                                       SAMPLE ASSEMBLY LANGUAGE CODING
STRT    CEU     1,,W                    UNIT 1 , WAIT FLAG
        DATA    '004000                 SET READ MODE
        AIP     1,,W                    READ IN LEADER
        BAZ     *,-1
        MEA     =,'377                  CHECK IF FIRST NON-ZERO CHAR. IS START CODE
        BAZ     *+2
        HLT
        SPB     INWD                    INPUT STARTING ADDRESS
        TAB
        TBI     ,3                      TRANSFER TO INDEX 3
        STA     CKSM                    INITIALIZE CHECK SUM
        SPB     INWD                    INPUT NEGATIVE WORD COUNT
        STA     NCNT
        AAM     CKSM                    UPDATE CHECK SUM
LOOP    SPB     INWD                    INPUT WORD
        STA     0,3                     STORE
        AAM     CKSM                    UPDATE CHECK SUM
        IMS     NCNT                    INCREMENT NEGATIVE WORD COUNT
        ILB     LOOP,3                  INCREMENT STORAGE ADDRESS AND LOOP
        SPB     INWD                    INPUT CHECK SUM
        MEA     CKSM                    CORRECT?
        BAZ     *+4
        SPB     $DMPH                   GO TO EXTERNAL SUBROUTINE TO
        DAC     ERRM                    OUTPUT ERROR MESSAGE
        DATA    4
        HLT                             FINISHED
INWD    ZZZ     **                      SUBROUTINE TO INPUT A WORD
        AIP     1,,W                    FIRST CHAR. IN
        LSL     8
        AIP     1,,W,R                  SECOND CHAR. IN
        LSL     8
        AIP     1,,W,R                  THIRD CHAR. IN
        BRU*    INWD                    RETURN
CKSM    ZZZ     **
NCNT    ZZZ     **
ERRM    DATA    '' CHECK SUM ERROR ''
        END
```

95098A.31

Figure 3-1. Assembly Coding Sample

ADDRESS ARITHMETIC

Any current location (*), symbolic (NAME) or absolute (decimal, 1234 or octal, '1234) address may be joined with a constant, any other current location (*), symbolic (NAME), or absolute (1234) address by an intervening plus (+) or minus (-) operator to define an effective address (that is, NAME + 4). This addressing technique may be extended to two or more operands (that is, A-B+3).

LITERAL ADDRESS

Literal addresses allow a constant to be defined, assigned to a memory cell, and that assignment location used as the address for the instruction. A literal address consists of an equal sign (=) followed by a constant. All constants defined in literal addresses will optimize storage so that all identical constants (regardless of their format) will be assigned only once. Any decimal integer, octal number, single asterisk (current location), previously defined symbolic name or combination of these formats joined by a + or - may follow the equal sign (=) in a literal address.

LOCATION TO BE FILLED

A double asterisk (**) indicates the address of this instruction is to be filled in by the object program at run time and is identical to an absolute address of 00000.

Table 3-1 lists examples of address field entries.

Table 3-1. Sample Address Field Entries

Field Entry	Description
	No Address
0, 1	Absolute Zero Address, Indexed
ALPH	Symbolic Address
ALPH, 1	Symbolic Address, Indexed
519	Absolute Decimal Address
'1067, 1	Absolute Octal Address, Indexed
NAME+4	Address Arithmetic
COMN-2, 1	Address Arithmetic, Indexed
ALPH-PHPA +2, 1	Symbolic Address Arithmetic, Indexed

Table 3-1. Sample Address Field Entries (Cont'd)

Field Entry	Description
=100	Literal Decimal Constant
='41237	Literal Octal Constant
=DLTA	Literal Symbolic Constant (Predefined)
*	Current Location
*-3, 1	Nearby Address, Indexed
**	Address to be Filled
**, 1	Indexed Address to be Filled
$SQRT	External Symbolic Address

COMMENTS FIELD

The comments field starts immediately after the first space in the variable address field. The comments field has no effect on the SEL 840MP Assembly Program but will be printed out on the symbolic listing if a listing is requested.

Any line which has an asterisk (*) in the first character position of that line will be considered a line of comments. (See line 1 of figure 3-1).

NOTE

Because of carriage width limitations on the console typewriter, comments appearing after column 50 will be listed only on the line printer.

IDENTIFICATION FIELD

This field is not checked by the SEL 840MP Assembly Program and is considered as part of the comments field. The identification field is provided as a programmer's aid. It may, for example, be used to identify a card or cards in a deck or for sequencing the card deck. The identification field is located in column 73 through 80.

MNEMONIC COMPUTER INSTRUCTIONS

Table 3-2 lists the mnemonic instructions that will be accepted by the 840MP Assembly Program. The allowable fields column shows the permissible fields that may be used with each instruction in their proper sequence. Required fields are underlined. Any symbolic notations described in this

Table 3-2. SEL 840MP Mnemonic Instructions (Cont'd)

Mnemonic Instruction	Allowable Fields	Description
EXU	EXU* Addr, Index	Execute Memory
HLT	HLT	Halt
EXI	EXI	External Interrupt
NOP	NOP	No Operation
TAC	TAC I/OP	Test and Activate (I/OP)
PID	PID Level, Group	Priority Interrupt Disable
PIE	PIE Level, Group	Priority Interrupt Enable
AOP	AOP Unit, Wait	A out to Peripheral Device
AIP	AIP Unit, Wait, Merge	A in from Peripheral Device
MOP	MOP Unit, Wait DATA OR MOP* Unit, Wait DAC* Addr, Index	Memory out to Peripheral Unit; Immediate Mode OR Address Mode
MIP	MIP Unit, Wait DATA OR MIP* Unit, Wait DAC* Addr, Index	Memory in from Peripheral Unit; Immediate Mode OR Address Mode
CEU	CEU Unit, Wait DATA Com. Code OR CEU* Unit, Wait DAC* Addr, Index	Command External Unit Immediate Mode OR Address Mode
TEU	TEU Unit DATA Test Code OR TEU* Unit DAC* Addr, Unit	Test External Unit Immediate Mode OR Address Mode
EFP	EFP	Set Floating Point (FP)
ECA	ECA	Clear EA
EDP	EDP	Set Double-Precision (DP)
ECB	ECB	Clear EB
ENO	ENO	Normalize EA
EIA	EIA	Interchange EA and EB
EBA	EBA	Transfer EB to EA
EAB	EAB	Transfer EA to EB

Table 3-2. SEL 840MP Mnemonic Instructions (Cont'd)

Mnemonic Instruction	Allowable Field	Description
ESR	ESR	Skip if EAU is Ready
ESZ	ESZ	Skip if EA is Zero
EPS	EPS	Skip if EA is Positive
ESN	ESN	Skip if EA is Negative
ESO	ESO	Skip on Overflow
ELB	ELB* Addr, Index	Load EB
EUN	EUN	Unnormalize
EST	EST* Addr, Index	Extended Store
ELN	ELN* Addr, Index	Extended Load Negative
EMU	EMU* Addr, Index	Extended Multiply
ESU	ESU* Addr, Index	Extended Subtract
EAD	EAD* Addr, Index	Extended Add
ELO	ELO* Addr, Index	Extended Load
EDV	EDV* Addr, Index	Extended Divide

Table 3-3. SEL 840MP Absolute Notation Formats

Variable Field Designation	Absolute Notation
Addr. (Operand Address)	5 octal digits ('00000-'77777), 5 decimal digits (00000-32767)
Index	1 digit (1, 2, or 3)
Switch No.	2 decimal digits, 0-23
Count (number of shifts)	2 octal digits ('00-'77), 2 decimal digits (00-63)
Group	1 digit (0, 1, 2, or 3)
Level	4 octal digits ('0001-'7777)
Unit	2 octal digits ('01-'77 representing units 0 to 63), or 2 decimal digits (representing units 0 to 63)
Wait	W for Wait
Merge	R for Merge
Common Code	See Appendix C
Test Code	See Appendix C

PSEUDO-OPERATION INSTRUCTIONS

This group of instructions is used to instruct the SEL 840MP Assembly Program and is not executed by the computer. A description of each of the pseudo-operations follows:

ABS — Set the mode of the assembly program to ABSolute. When in this mode, all symbolic addresses will be assigned relative to location 00000 and the object program will be in a nonrelocatable format.

REL — Set the mode of the assembly program to RELative. When in this mode, all symbolic addresses will be assigned relative to the start load address (assigned when loading the program into memory), and the object program will be in a relocatable format. The assembly program will remain in the relative mode until changed by an ABS pseudo-operation.

ORG — The variable field specifies an address. When the assembly program is in the ABSolute mode, this address specifies the location of the next instruction. When in the RELative mode, this address will be added to the start load address to specify the location of the next instruction. In either case, all subsequent instructions will be stored sequentially until another ORG pseudo-operation is given.

BAR — The BAR pseudo-operation is used to convey information to the loader when the program is being loaded into memory. The primary purpose of the BAR pseudo-operation is to select areas of memory for program loading. The coding format for the BAR pseudo-operation is as follows:

> BAR in the Operation field (columns 6-9); an apostrophe ('), signifying octal numbers, in column 11 of the variable field; BAR 0 contents in columns 12 and 13; BAR 1 contents in columns 14 and 15; BAR 2 contents in columns 16 and 17; and BAR 3 contents in columns 18 and 19.

The contents of the BAR's must not exceed '14. If, for example, the contents of BAR 0, BAR 1, BAR 2, and BAR 3 were 0, 1, 2 and 3, respectively, the operation and variable fields would appear as follows: BAR '00010203.

If BAR 0 - 3 are set as shown above and it is desired to load the program into memory locations 32K-40K, one of the BAR settings would need to be changed to a 04 and an ORG pseudo-operation to the proper location would be written following the BAR. In normal use, a BAR pseudo-operation will always be followed by an ORG.

EQU — The symbol in the location field will be assigned the address or value specified in the variable field. In a one-pass assembly mode, EQU should appear before any reference is made to the symbol in the location field. Any symbol in the variable field must be previously defined in both one-pass and two-pass assembly modes.

DATA — The variable field of this pseudo-operation may contain any number and any mixture of the following data item formats. If the location field contains a symbol, it will be assigned the location of the first data item. If more than one data item is present (separated by commas), they will be assigned sequential storage locations.

a. Octal Data Item - Format: An optional sign (+ or -), followed by an apostrophe character ('), followed by 0 to 8 digits (0 through 7). If less than 8 digits are present, the number will be right justified with leading zeros added. If a minus sign is present, the number will be two's complemented; a plus sign is ignored by the Assembly Program.

b. Decimal Integer - Format: An optional sign (+ or -) followed by 0 to 7 decimal digits (0 through 9). The number will be converted to binary and stored at a scale of B23. The number will be stored postively unless a minus sign is present. A minus sign will cause the two's complement of the number to be stored.

c. Fixed-Point, Single-Precision Decimal Data - Format: An optional sign (+ or -), 0 to 7 decimal digits (0 through 9), mixed with an optional decimal point, the letter B, followed by a decimal number between +23 and -23.

Example: -3.14157B6. A minus sign will cause the two's complement of the number to be stored. One word will be generated.

NOTE

The assembler assumes the radix point immediately to the right of the sign bit position for a B0 scale factor.

NOTE (Cont'd)

A positive B factor moves the radix point to the right, a negative one moves it to the left.

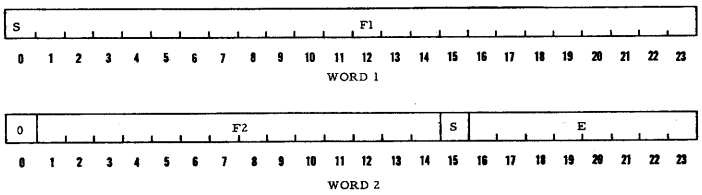
d. Fixed-Point, Double-Precision Data - Format: An optional sign (+ or -), 0 to 14 digits mixed with an optional decimal point, the letter C, followed by a decimal number between +46 and -46.

Example: 103.63794223C10. A minus sign will cause the two's complement of the number to be stored. Two words will be generated.

NOTE

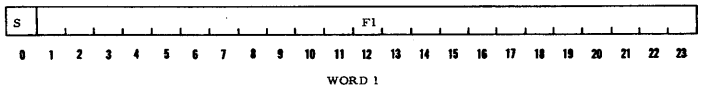
The C factor operates the same as the B factor in single-precision data.

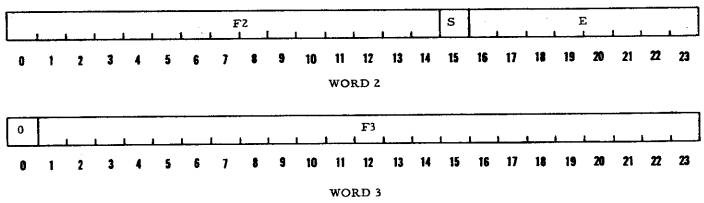
e. Floating-Point Data - Format: An optional sign (+ or -), 0 to 12 decimal digits (0 through 9) mixed with an optional decimal point, and optionally followed by a decimal exponent consisting of the letter E, preceding a decimal number between +75 and -75. (Either the decimal point, the letter E, or the sign of the exponent must be present). Two sequential memory cells are generated(for each floating-point data item) using the following format:



E = Characteristic (two's complement if negative)
F1 and F2 - Fraction (two's complement if negative)

f. Floating-Point, Double-Precision Data - Format: An optional sign (+ or -), 0 to 19 digits mixed with an optional decimal point, the letter D, followed by a decimal number between +75 and -75. Three sequential memory cells are generated (for each double-precision, floating-point item) using the following format:





E = Characteristic (two's complement if negative)
F1, F2 and F3 = Fraction (two's complement if negative)

g. Alphanumeric Data - Format: Two apostrophe characters (' ') followed by any number of characters (including blanks) until another pair of apostrophe characters is read. All characters between the apostrophe pairs will be stored 4 per word (6 bits each) and the last word will be left justified, if necessary.

Example:
```
01142010
01402405   ' ' ALPHA TEST ' '
23245640
```

The above example is in-line printer code (truncated ASCII code). This code will be used internally by the assembler to represent alphanumeric data. The I/O drivers will translate from external to internal code and vice versa when necessary depending upon the I/O device in use.

h. Symbolic Address Data - Format: Any symbolic address optionally followed by address arithmetic. The effective 15-bit address will be stored in memory (similar to that generated by a DAC pseudo-operation). The address may not be tagged as indexed or indirect.

BSS Reserve a block of memory starting at the current location and extending for the number of computer words specified in the variable field. (If the variable field is symbolic, it must be previously defined.) The location field is optional; however, if a symbolic location is written, it will refer to the first word in the block of memory.

BES Same as BSS except if a symbolic location is used, it refers to the last word in the block, plus one.

CALL This pseudo-operation will generate the necessary coding and actions to call a subroutine from a library tape into memory. The CALL pseudo-operation is then replaced by a subroutine transfer instruction.

(SPB) to this subroutine. The variable field contains the subroutine name and the location field, when occupied, refers to the resulting SPB instruction. Logic is contained within the loader to assure that only one copy of a subroutine is called into memory from the library tape regardless of the number of CALL's for that subroutine. The subroutine name must start with a letter and may contain from 1 to 6 characters. Another way to call external subroutines would be with a leading dollar sign on the subroutine's name.

Examples:
```
        SPB    $SQRT
   or   CALL   SQRT
```

NAME When writing subroutines for inclusion into a library tape, the name by which the subroutine is to be called is specified by the NAME pseudo-operation. This must appear as the subroutine's first instruction line(s). The variable field consists of two symbolic names. The first is the name of the subroutine and is 1 to 6 characters long (FORTRAN IV compatible). The second name is the symbolic entry location for the subroutine and is 1 to 4 characters long, the first character being a letter.

More than one NAME pseudo-operation may be included in a subroutine if alternate names for the subroutine exist with either the same or different entry points. Also, external variables may be defined by the NAME pseudo-operation.

ZZZ The instruction bits (3 to 8) are set to 000000. The rest of the instruction is determined by the variable field and the presence of an indirect indicator (*) following the pseudo-operation (ZZZ*).

DAC The instruction bits (3 to 8) are set to 000000. (Same as ZZZ). The reset of the instruction is determined by the variable field and the presence of an indirect indicator (*) following the pseudo-operation (DAC*).

*** Same as ZZZ, but indicates that the instruction field will be filled in at program run time.

MOR This pseudo-operation causes a pause in the assembly process. This is useful when the source program is on more than one tape, and a pause is needed to change tapes.

END This pseudo-operation must appear as the last instruction in any program or subroutine being assembled. END tells the assembly program that assembly is complete. If the variable field is not blank, it should specify the starting location of the program just assembled.

REM This pseudo-operation causes top of form on line printer listing.

Table 3-4 summarizes the SEL 840MP pseudo-operation instructions and gives specific examples of the use of each.

Table 3-4. Summary of SEL 840MP Pseudo-Operations

Symbolic Location	Pseudo-Operation	Variable Field Entry	Description
	ABS		Set mode to absolute
	REL		Set mode to relative
	ORG	'1000	Set starting location of program
	BAR	'00010203	Select desired memory bank
ALPH	EQU	BETA +2	Set symbol equal to symbol
IND	EQU	2	Set symbol equal to value
OCT	DATA	'12734, -'21 + '6470	Octal Data
	DATA	9876, -3000, +24	Decimal integer data

Table 3-4. Summary of SEL 840MP Pseudo-Operations (Cont'd)

Symbolic Location	Pseudo-Operation	Variable Field Entry	Description
MIX	DATA	123.456B10, -3B6, 12B	Fixed point data
	DATA	1.23456789C5, 10C23	Double-precision fixed point data
FLOT	DATA	22.3344E0, .12345E2	Floating point data
	DATA	1/23456789D4, 1D30	Double-precision floating point data
	DATA	X4, A-DELT + 1	Symbolic address data
ALPA	DATA	"HELP", "12-34, A.E2"	Alphanumeric data
	DATA	3, '77, 1.23B4, -1233E-3, X4	Mixed format data
TBL	BSS	100	Block Storage (Front Label)
	BES	5	Block Storage (End Label)
	CALL	SIN	Library Tape Call
	NAME	SIN, SIN	Library Subroutine Name
	ZZZ	ALPHA	Instruction Bits = 00000
	DAC	ALPHA	Instruction Bits = 00000
	REM		Top of Form (Line Printer)
	***	ALPHA	Instruction to be set at Program Run Time
	***	**	Word to be filled at Program Run Time
	MOR		Pause during Assembly
	END	STRT	End of Main Program
	END		End of Subroutine

SECTION IV
INPUT / OUTPUT

INTRODUCTION

The SEL 840 Multiprocessor Computer System I/O structure is designed primarily to meet the needs of one-line, real-time computer applications where large memory capacity and simultaneous multiprocessing is required. This application area imposes the most severe requirements on I/O capabilities due to the large number and wide variety of peripheral devices employed and the time-sharing mode of operation encountered. In addition to the standard data processing devices (that is, card reader/punch, paper tape reader/punch, magnetic tape, disc and keyboard/printer units) many real-time systems contain data acquisition and display units, and control and communication interface. Therefore, the I/O structure must enable the connection of a large number of peripheral devices to permit (a) time-sharing communication with the computer, and (b) simultaneous processing.

The standard 840MP Computer I/O structure consists of the Central Processor Input/Output Processor (CP-I/OP) which provides communication with up to 64 peripheral devices or device controllers. A single peripheral device controller may communicate with several peripheral devices. Up to six automatic Block Transfer Control (BTC) units and two Computer Graphics Processors (CGP) may be added to the basic CP-I/OP. In addition to the CP-I/OP, up to four Shared Memory Input/Output Processors may be present in an 840 Multiprocessor Computer System containing a shared memory system. Each SM-I/OP may have up to three fully buffered BTC units capable of communicating with up to 48 peripheral devices or peripheral device controllers. Two CGP's can be connected to the SM-I/OP in place of two BTC units. Each CGP displaces one BTC.

The CP-I/OP alone is capable of meeting the I/O requirements of many systems. It is ideally suited to real-time applications in that each I/O instruction causes the device addressed by the instruction to be connected to the Central Processor, the transfer to be made, and the device to be disconnected from the processor. Therefore, the successive transfer of data words to/from two or more different peripheral devices requires no intervening housekeeping operation such as channel and unit testing and connection.

The time-sharing capability of the CP-I/OP is further enhanced by the fact that all standard Systems Engineering Laboratories peripheral devices contain their own data buffers. Therefore, the CP-I/OP is never occupied by buffering data to be transferred to/from a peripheral device. As a result, the data transfer can be executed in three machine cycles (5.25 microseconds), and the CP-I/OP can be released immediately for transfer to/from a different device.

Figure 4-1 is a simplified block diagram illustrating the functional connection of peripheral devices to the 840MP Computer via the input/output bus. BTC units may be added to the CP-I/OP to provide fully-buffered block data transfer capability between all private and shared memory and peripheral devices. BTC units enable blocks of data up to 98,304 words in length to be transferred. One memory cycle is stolen from the central processor for each word transferred, except that some basically low-speed peripheral devices take a second cycle per output word transfer.

The primary reason for adding BTC units to the central processor is to free the mainframe to perform internal processing functions while data is being transferred between memory and peripheral devices at high rates. For example, a continuous stream of data words can be read into memory, blocked, and recorded on magnetic tape in gapped format (using two BTC units) resulting in a loss of only slightly over two machine cycles (one for input, one for output) per word transferred. For a typical word rate of 20 kc, only an average of 3.5 of each 50 microseconds of machine time are used to accomplish the total input and output transfer function. The remainder of the time is available for performing such functions as scaling or limit checking of the data. This same transfer function can be accomplished without use of BTC's; however, a 20 kc word rate would utilize almost all available mainframe cycles.

The optional Computer Graphics Processor (CGP), Model 84-235MP, is a high-speed data transferring control unit designed to satisfy the specialized needs of the SEL Computer Graphics Systems. The control unit is similar to the 840MP optional BTC with the exception of its specialized operating characteristics and added control functions. A BTC, when outputting data, is unmindful of the nature

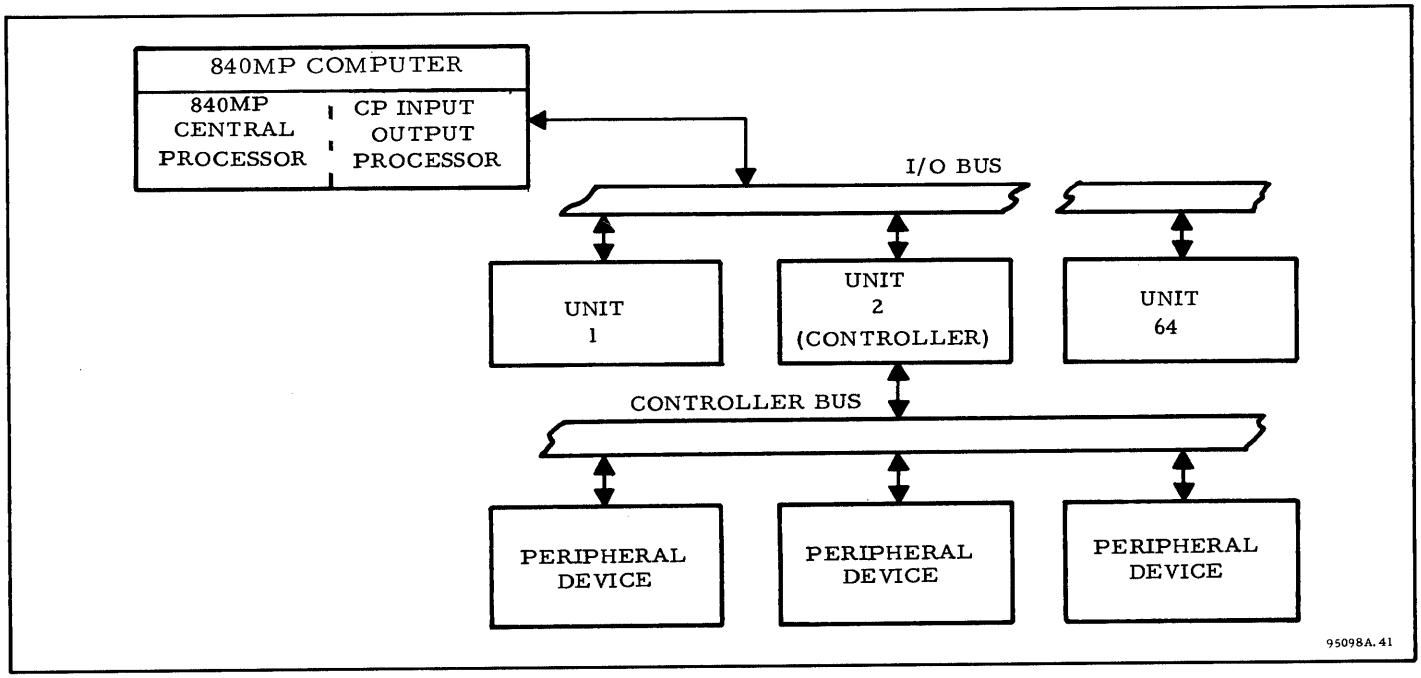


Figure 4-1. Connection of Peripheral Devices to the 840MP Computer

of this data. However, the CGP examines each word as it comes from memory and either interprets the word as data and sends it to the Computer Graphics System or as an instruction and takes appropriate action.

The instructions allow the CGP to operate on its address counter, thereby freeing the 840MP Computer from much of the control unit servicing, and allowing it more time to operate on the buffer areas of the system. This feature allows the use of subroutines to generate frequently used patterns.

The CGP is used in conjunction with the SEL 816A Computer Graphics System to provide the most efficient method of transferring data from the 840MP Computer to the display unit. Using the CGP minimizes the amount of computer memory and transfer time required to support the display. The CGP also provides a high degree of flexibility in display format generation since the CGP contains the capability of executing the following instruction: (1) Branch Unconditionally, (2) Store, Place, and Branch, and (3) Stop. The first two instructions (which have the same execution capabilities as the corresponding computer instructions) enable the contents of noncontigous memory areas to be transferred automatically to the display. This capability enables display programs to be organized to provide maximum usage of closed subroutines that are stored in memory a single time and used as often as required in a given display format. The Stop command enables the display unit to automatically control the refresh rate, and maintain a fixed rate regardless of the amount of data being displayed.

In a typical 840 Multiprocessor Computer System configuration, the 840MP Computer can be relieved of many time-consuming I/O functions through the use of the SM-I/OP's. The SM-I/OP's are initialized by the execution of a single instruction - test and activate (TAC) - in the 840MP Computer. Once initialized, the SM-I/OP proceeds independently to control the transfer of data to/from the Shared Memory and external peripheral devices.

Figure 4-2 illustrates, in a simplified block diagram, the functional connection of an SM-I/OP in an 840 Multiprocessor Computer System. The SM-I/OP utilizes the BTC principle exclusively for data transfer between external peripheral units and shared memory. From one to three BTC units may be employed in each SM-I/OP.

CENTRAL PROCESSOR—INPUT /OUTPUT PROCESSOR (CP-I / OP)

Figure 4-3 is a detailed block diagram of the 840MP Computer private I/O structure. It shows the connection of the I/O structure to the 840MP Computer mainframe and memory and the connection of the peripheral devices to the I/O bus. Three of the five peripheral units shown have additional connections to BTC units. However, all five units shown and additional units (up to 64 total) can be

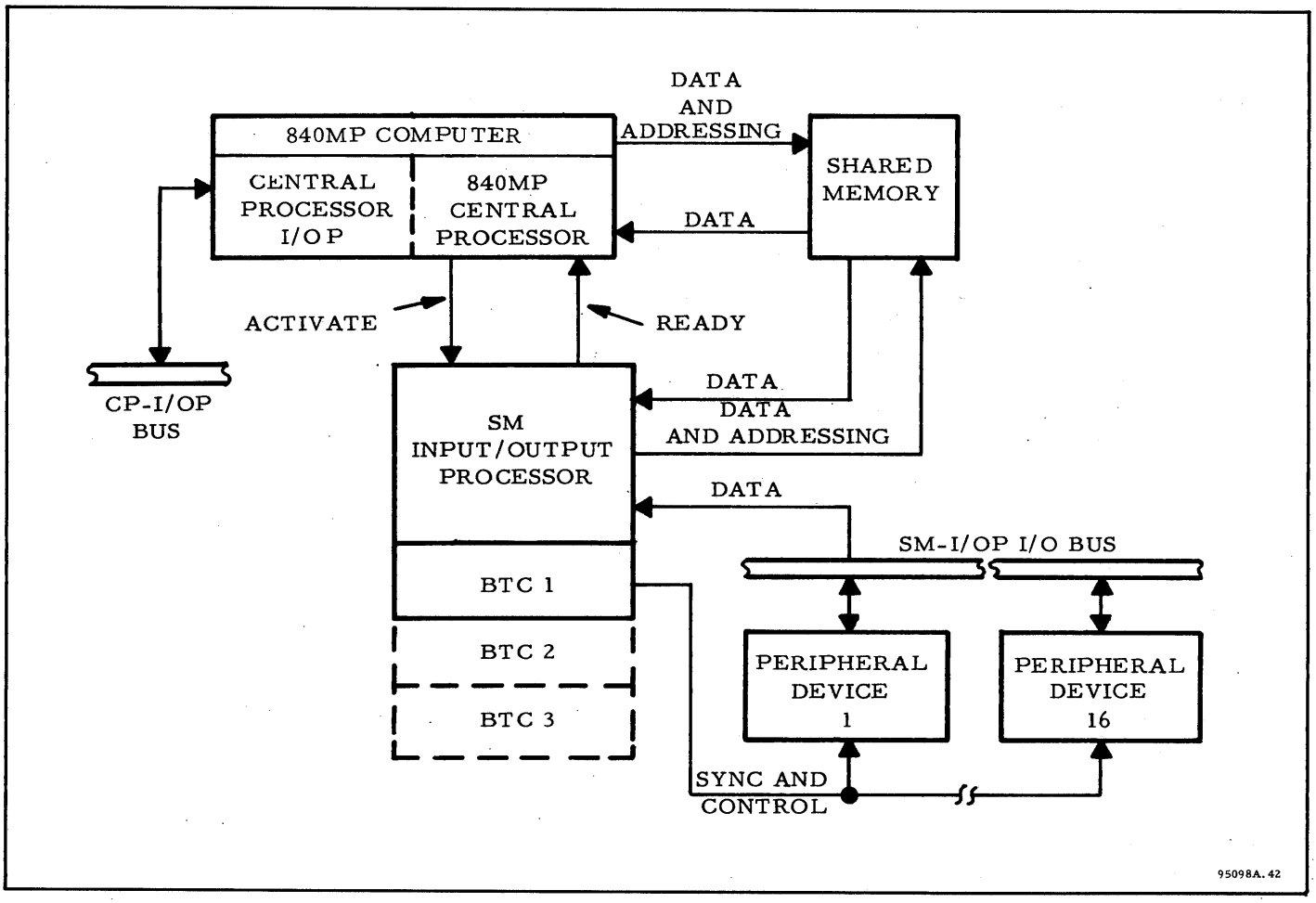


Figure 4-2. Connections of Peripheral Devices to Shared Memory Input/Output Processor

commanded by, and can communicate with, the 840MP Computer under single-word program control. The operation of the BTC units is discussed separately in this section.

The CP-I/OP provides a positive synchronization control for data flow between the central processor and peripheral units. It can synchronize data transfer between a peripheral unit and either memory or the A-Accumulator. The data path for each word or character transferred is controlled by the program which can execute any of the I/O instructions:

The basic, automatic execution sequence for all I/O instructions in the central processor consists of three steps:

a. Connect the unit specified by the instruction to the I/O bus.

b. Execute the test, command, or data transfer between the processor and the peripheral unit.

c. Disconnect the peripheral unit from the I/O bus.

Three significant features of this execution sequence are:

a. The unit is always specified by the I/O instruction.

b. The unit is always connected to, and disconnected from, the processor by the execution of the instruction.

c. Data transfers are always made directly between the specified unit and the processor with no intermediate buffering.

The result of these three features is that the Central Processor I/O structure is always available for use without testing. It is never busy, except during the times that I/O instructions are being executed. Channel testing or selection are never required. In addition, no unit selection instructions are required, since each I/O instruction

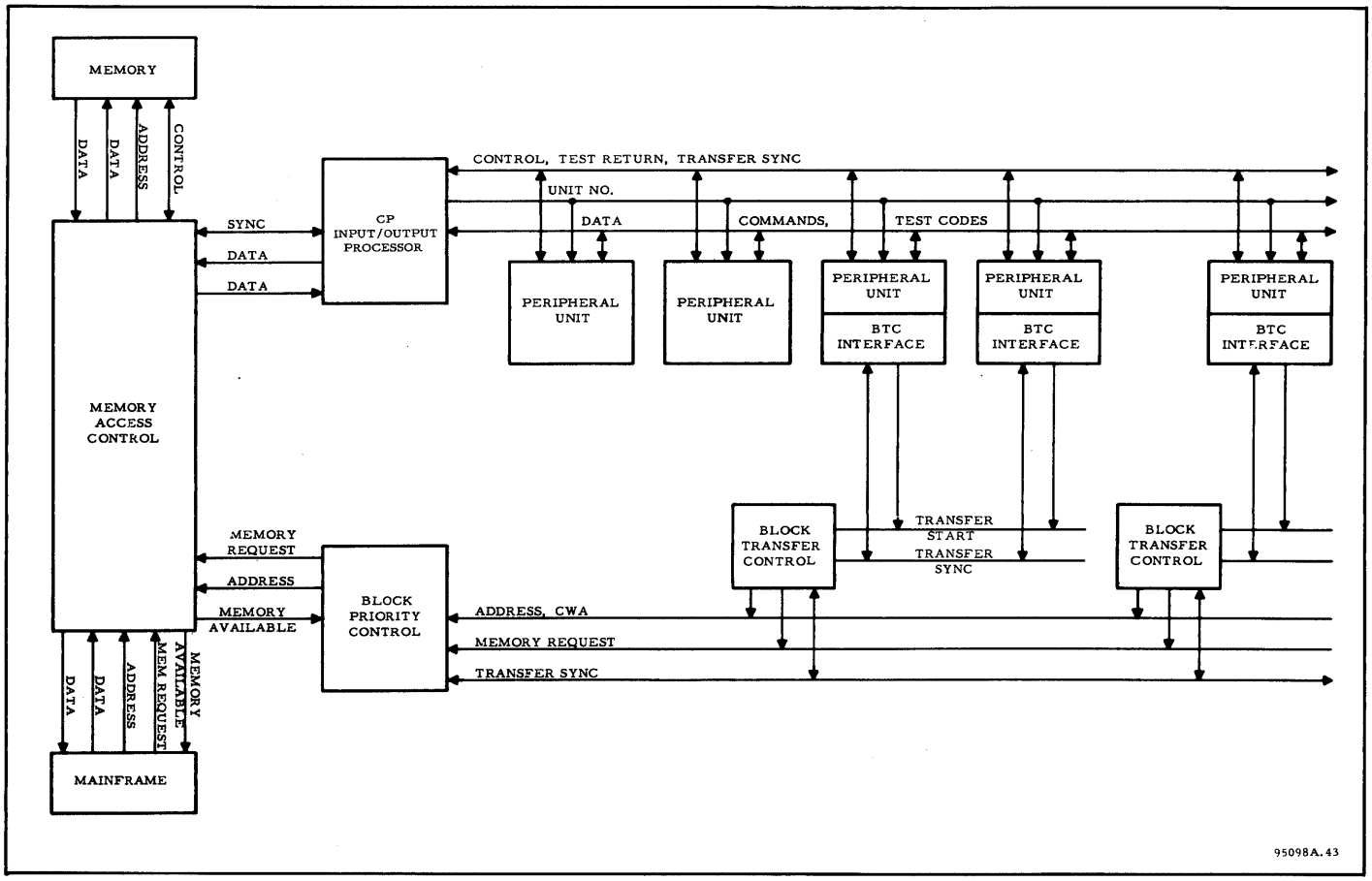


Figure 4-3. Input/Output Configuration and Computer Interface

causes the unit specified by the instruction to be selected for transfer.

The CP-I/OP I/O bus connects all peripheral units to the CP-I/OP in a daisy-chained manner, as shown in figure 4-4. The I/O bus contains 24 data lines, six unit number lines, and numerous control lines. An additional parity line is optionally available. The 24 data lines provide two-way communication paths. All data, CEU command words, and TEU test words are transferred over these lines. Word-oriented units such as acquisition subsystems may contain a full set of 24 cable drivers and terminators for the data lines. Character-oriented units having character assembly buffers such as magnetic tape control units also contain a full set of 24 cable drivers and terminators. Character-oriented units having character buffers such as paper tape punches and readers contain only eight or more (as required) cable drivers and/or terminators. In this case, data commands and test codes are received from the 840MP Computer on the eight lines corresponding to 840MP Computer bit positions 0-7. Some units also receive commands from bits 16 and 23. Single characters are always transferred to the 840MP Computer on the data lines corresponding to bit positions 16-23. Characters having less than eight bits are right justified in the eight-bit field. The data lines connected to each peripheral device are defined in Section VI. The six device (unit) number lines connected to each device permit up to 64 individual devices to be addressed by the central processor. The control lines contain the signal names in table 4-1.

These lines are used to enable I/O instructions to be executed in the following basic sequence (TEU differs):

a. The 840MP Computer initiates execution by sending out the unit number contained in the instruction. The 840MP Computer also sends out the instruction sync and instruction command (data, command, test, input/output) signals.

b. The addressed unit responds by sending the unit sync return and unit test return signals to the computer.

c. After recognizing the unit sync return signal, the computer tests the unit test return

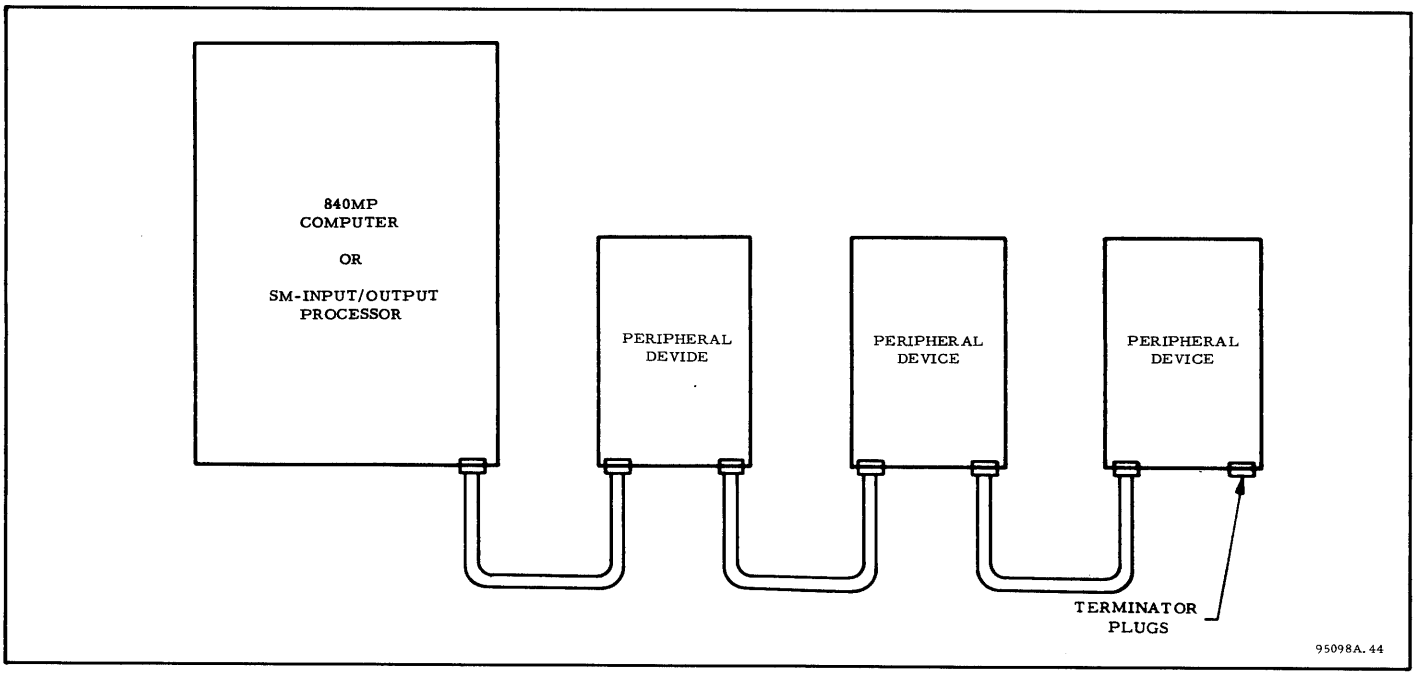


Figure 4-4. Peripheral Connections - Simplified

signal for the unit status (ready to execute the command, or not ready).

d. If the unit indicates ready, the data transfer is made. The data here and data accepted signals synchronize the transfer. For computer input transfer, the unit ready signal also indicates data here.

e. After the transfer is completed, the central processor tests the control lines from the unit to insure that they have returned to the off level. The next instruction is started in the following machine cycle.

The normal execution time for each I/O instruction is three machine cycles. However, one or more cycles may be added if the total I/O cable in the chain exceeds approximately 50 feet. In addition, presence of the wait bit in an I/O instruction delays completion of instruction execution until the unit indicates a ready status. The use of the wait bit is described in Section II.

The execution sequence is similar for all I/O instructions except TEU. When a TEU instruction is executed, no ready test is made before transfer of the test word. Transfer is made following recognition of the unit sync return signal. The test return line is tested after the test word has been transferred to the unit. The return signal is a particular unit status gated on the test return line by the value of the test word transferred to the unit.

SHARED MEMORY—INPUT / OUTPUT PROCESSOR (SM-I / OP)

GENERAL

The SM-I/OP used with the 840 Multiprocessor Computer System consists of the following major functional units:

Table 4-1. I/O Control Signals

Signals	840MP Computer Commands
Instruction Sync	All I/O Instructions
Data Instruction	AIP, AOP, MIP, MOP
Command Instruction	TEU
Input/Output	AIP, AOP, MIP, MOP
Wait Flag	AIP, AOP, MIP, MOP, CEU
Unit Test Return	All I/O Instructions
Unit Sync Return	All I/O Instructions
Computer Data Here	AOP, MOP, CEU, TEU
Computer Data Accepted	AIP, MIP
Unit Data Accepted	AOP, MOP, CEU, TEU
Computer Clock	
Master Clear	

a. Timing and access control logic to provide synchronization between the SM-I/OP elements, external peripheral devices, and shared memory; and to handle communication between the SM-I/OP and the 840MP Computers.

b. Priority logic to regulate requests for SM-I/OP use from the central processor.

c. BTC units.

d. Temporary data storage and gating circuits.

A functional block diagram of the SM-I/OP is shown in figure 4-5. Each of the four SM-I/OP's has dedicated locations in shared memory set aside as defined in table 4-2.

SHARED MEMORY INPUT/OUTPUT PROCESSOR INITIALIZATION

The SM-I/OP dedicated memory location is used to store a command or test function code which is to be transferred to a peripheral unit. Before the SM-I/OP can execute a command (CEU) or test (TEU) instruction the SM-I/OP dedicated location must be loaded by the 840MP Computer with a word having the format illustrated in figure 4-6.

As shown in figure 4-6, bit nine must be set to a ONE to initiate a command operation, or a ZERO for a test operation.

The command and test function bits are the same as shown in the CEU and TEU second word formats

Table 4-2. Input/Output Processor Dedicated Memory Locations (Shared Memory)

Unit	Dedicated Location	Contents of Dedicated Location	Unit	Dedicated Location	Contents of Dedicated Location
I/OP No. 1	100000_8	Command or Test Word	I/OP No. 3	100020_8	Command or Test Word
BTC No. 1	100002_8	First Word Address	BTC No. 1	100022_8	First Word Address
	100003_8	Word Count		100023_8	Word Count
BTC No. 2	100004_8	First Word Address	BTC No. 2	100024_8	First Word Address
	100005_8	Word Count		100025_8	Word Count
BTC No. 3	100006_8	First Word Address	BTC No. 3	100026_8	First Word Address
	100007_8	Word Count		100027_8	Word Count
I/OP No. 2	100010_8	Command or Test Word	I/OP No. 4	100030_8	Command or Test Word
BTC No. 1	100012_8	First Word Address	BTC No. 1	100032_8	First Word Address
	100013_8	Word Count		100033_8	Word Count
BTC No. 2	100014_8	First Word Address	BTC No. 2	100034_8	First Word Address
	100015_8	Word Count		100035_8	Word Count
BTC No. 3	100016_8	First Word Address	BTC No. 3	100036_8	First Word Address
	100017_8	Word Count		100037_8	Word Count

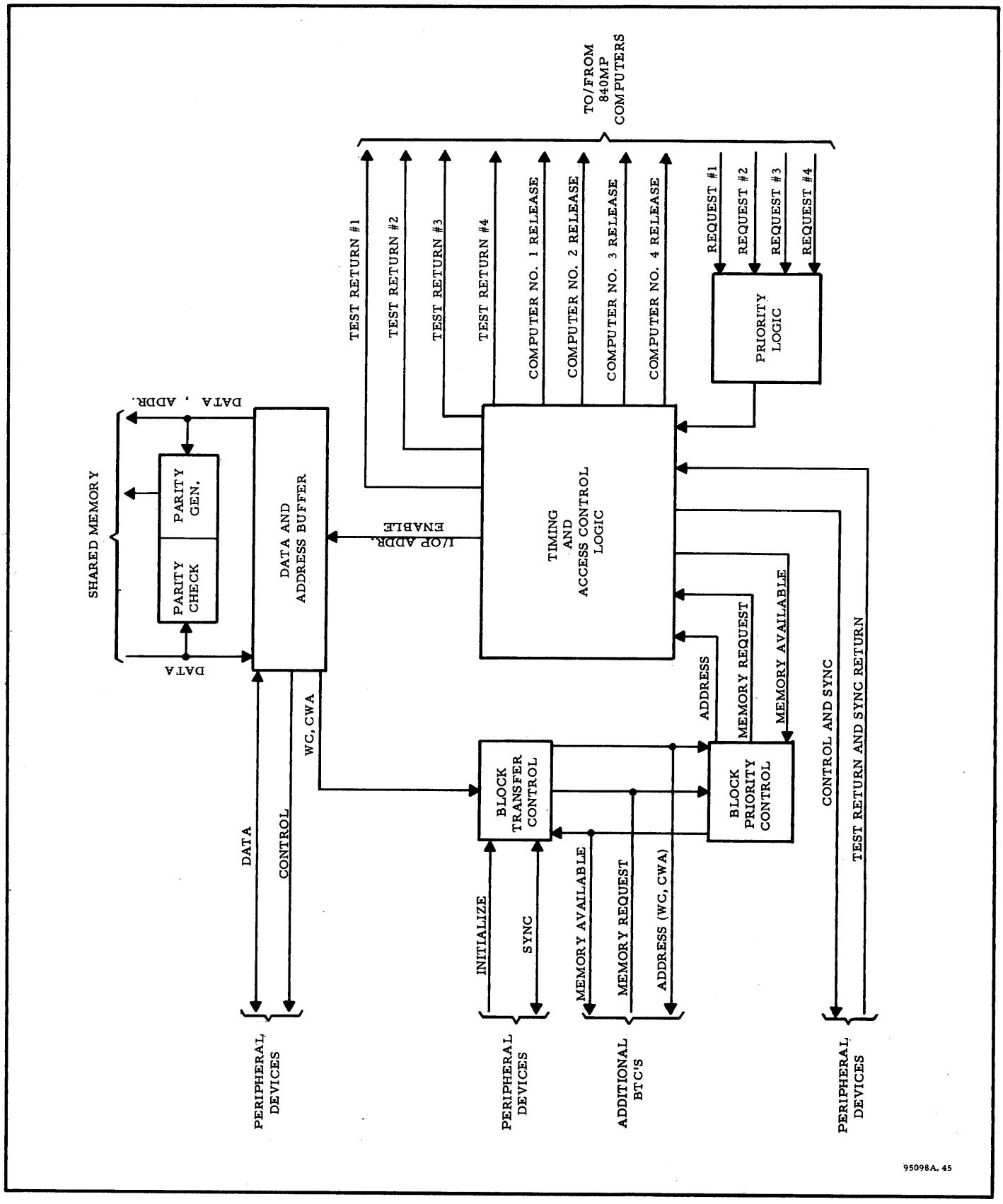


Figure 4-5. Shared Memory Input/Output Processor Functional Block Diagram

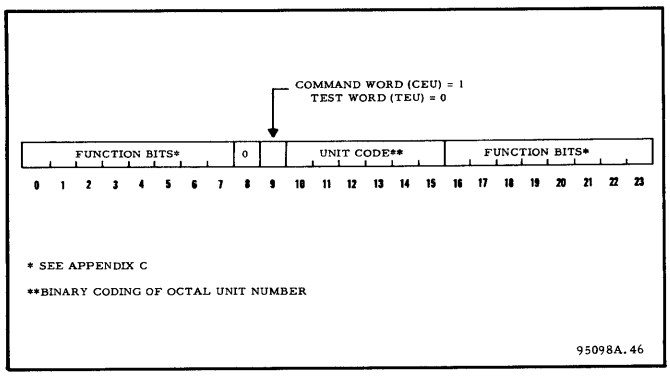


Figure 4-6. Shared Memory I/OP, CEU, and TEU Word Formats

shown in Appendix C. The unit code in bit positions 10-15 is set to the binary unit number assigned to the peripheral unit to be commanded or tested. The one-bit operation code (CEU = 1, TEU = 0) is stored in bit nine. The instruction stored in the first SM-I/OP dedicated location is executed when a multiprocessor executes a TAC instruction, which addresses the SM-I/OP. The execution sequence for a TAC instruction is as follows:

a. The 840MP Computer sends a request signal to the SM-I/OP signifying that it wants to use the SM-I/OP.

b. The 840MP Computer waits for a release signal from the designated SM-I/OP.

c. When the release signal is received, the computer advances the program counter by either one or two memory locations, depending upon the value of the test return status signal. The next instruction is then executed.

TAC is a skip instruction; unlike the 840MP Computer I/O instructions, TAC has no wait flag feature. If the TAC initiates a command operation and the peripheral unit is ready, the command is executed, the test return status (figure 4-5) signal is generated, and the next sequential instruction is skipped. If the peripheral unit is not ready, the test return status signal is not generated, and the next sequential instruction is executed. If a test operation is initiated by the TAC instruction, the result of the test determines whether or not the test return status signal is generated and, consequently, whether the next sequential instruction is skipped or executed by the 840MP Computer. The skip logic is exactly the same in the SM-I/OP as in the 840MP Computer. If the test result is true the next sequential instruction is skipped.

The request signal from the 840MP Computer is processed in the SM-I/OP priority logic. The priority logic contains anticoincidence circuits and four latches which are used to determine which 840MP Computer takes precedence if simultaneous request signals are received. If no higher-priority request is being serviced by the SM-I/OP, the incoming request is handled immediately. If a higher-priority 840MP Computer request is being serviced, the incoming request must wait for the higher-priority operation to be completed plus an additional six machine cycles. This six-cycle delay is a hardware-controlled function which ensures that an 840MP Computer regardless of its priority, will be able to maintain control of a given SM-I/OP until all its operations are completed. During the six cycle delay (after the release signal has been generated), the 840MP Computer presently in control maintains the highest priority. If this particular 840MP Computer requests another SM-I/OP operation during the delay period, the request will be stored in the SM-I/OP priority logic and serviced immediately. No other SEL 840MP Computer request will be acknowledged by the priority logic during this period. The only restriction is that the 840MP Computer in control must request another SM-I/OP operation (execute a TAC instruction) before the seventh machine cycle commences. To accomplish this, the total execution time for the instruction between requests (TAC instructions) must not exceed four cycles.

The release signal is generated automatically during the execution of a TAC instruction. The release signal allows the 840MP Computer to proceed with the execution of the instruction. The only condition under which the release signal is not generated is an SM-I/OP I/O hold condition. In the SM-I/OP, an I/O hold condition generally indicates a peripheral unit malfunction or programming error (that is, an invalid unit code). Recognition of an I/O hold by the SM-I/OP will cause the appropriate I/O HOLD indicator (on the SM-I/OP control panel) to light. During an I/O hold condition, both the SM-I/OP and the SEL 840MP Computer which initiated the operation will stop execution. The SM-I/OP and SEL 840MP Computer can be released from an I/O hold by depressing the appropriate I/O HOLD RELEASE switch on the SM-I/OP control panel.

SM-I/OP DATA TRANSFER

Data transfer between external peripherals and shared memory are handled by the BTC units in the SM-I/OP. BTC operation is discussed in the following paragraphs.

BLOCK TRANSFER CONTROL UNIT

GENERAL

The Block Transfer Control (BTC) is an Input/Output control unit which enables the fully-buffered

transfer of blocks of data between peripheral devices and memory at relatively high rates of speed. BTC units are optional on the SEL 840MP Computer; however, the SM-I/OP uses BTC units for all data transfer. At least one BTC is required in an SM-I/OP and each peripheral device connected to that BTC must have a BTC interfacing circuitry installed. The principle features of the BTC are listed below:

Bits per Transfer	Full memory word (24 bits in parallel)
Maximum Words per Block-BTC in CP-I/OP	98,304 (private memory)
-BTC in SM-I/OP	65,536 (shared memory)
Minimum Word per block	One
Maximum Transfer Rate	572,000 words per second
Memory Cycles Stolen per Transfer	One or more, depending on cable length
Block Transfer Reinitialization	Automatic
Maximum Number of BTC Units per CP-I/OP	Six (optional)
per SM-I/OP	Three (one required)
Maximum Number of Peripheral devices per BTC	Sixteen
Maximum Number of Optional CGP units	
CP-I/OP	Two
SM-I/OP	Two

BTC OPERATION

The BTC unit contains two binary counters plus transfer initialization and synchronization logic. One of the counters stores the current word address (CWA) and the second stores the word count (WC). CWA defines the storage location for each word transferred to/from memory. WC defines the number of words to be transferred. The initial values for CWA and WC are obtained from dedicated locations in memory (private or shared) by the BTC each time a new block transfer is initiated.

NOTE

The initial value of CWA is identified as the First Word Address (FWA). This allows the BTC starting address to be defined and distinguished from any other CWA.

The dedicated Shared Memory Locations for the BTC units associated with the SM-I/OP are listed in table 4-2. Table 4-3 lists the BTC private memory assignments for use with the SEL 840MP Computer.

Each time a word is transferred between memory (private or shared) and the specified peripheral units, CWA (or FWA at initial location) is incremented and WC is decremented. The block transfer is completed when WC equals zero. The BTC automatically initiates a new block transfer by obtaining a new set of initial values for CWA and WC from the dedicated locations in private or shared memory. The block transfer sequence is ended by placing a terminate code in the WC word.

Table 4-3. Central Processor BTC Dedicated Memory Locations (Private Memory)

Unit	Dedicated Location	Contents of Dedicated Location
BTC No. 1	60_8	First Word Address
	61_8	Word Count
BTC No. 2	62_8	First Word Address
	63_8	Word Count
BTC No. 3	64_8	First Word Address
	65_8	Word Count
BTC No. 4	66_8	First Word Address
	67_8	Word Count
BTC No. 5	70_8	First Word Address
	71_8	Word Count
BTC No. 6	72_8	First Word Address
	73_8	Word Count
CGP No. 1	74_8	First Word Address
	75_8	Word Count
CGP No. 1	76_8	First Word Address
	77_8	Word Count

The terminate code is a ONE in bit position 0 of the memory word.

NOTE

The BTC, end of block signal is connected to the SEL 840MP Computer priority interrupt system. A priority interrupt is generated each time the end of block (WC = 0) signal occurs. Unless the terminate code is a ONE, a new block transfer is initiated prior to generation of the interrupt signal.

The value of CWA may be transferred from private memory to the dedicated memory location by the execution of a CEU instruction containing a ONE in bit position 21 of the second word (see Appendix C), and addressed to the unit in use. To obtain the CWA value from shared memory, the central processor must execute a TAC instruction (addressed to the appropriate SM-I/OP) which will cause the proper command to be transferred to the peripheral unit in use. Bit position 21 of the command word (in the SM-I/OP dedicated location) should contain a ONE to transfer the CWA to the dedicated BTC shared memory location.

BTC INITIALIZATION AND DATA FLOW

The BTC unit is initialized through the peripheral unit to/from which the block transfer is to be made. Figures 4-3 and 4-5 illustrate the data and control paths involved when BTC units are connected to the CP-I/OP structure and the SM-I/OP, respectively.

When operating with the CP-I/OP structure, execution of the proper command external unit (CEU) instruction causes the unit addressed by the CEU to send an initialize signal to the BTC to which it is connected. When using an SM-I/OP, the SEL 840MP Computer must initialize the appropriate SM-I/OP by executing a TAC instruction. The TAC execution causes the proper command code to be transferred to the peripheral unit which, in turn, generates an initialize signal and returns it to the BTC in the SM-I/OP. When a BTC receives an initialize signal from the peripheral unit, it requests a memory cycle from the access control logic in either private (CP-I/OP) or shared (SM-I/OP) memory.

BTC also generates the address of the First Word Address dedicated location assigned to it. When the memory cycle is granted, the FWA value is transferred from private or shared memory to the CWA counter in the appropriate BTC. The BTC then requests a second memory cycle and places the address of the WC dedicated location on the address lines. When the second cycle is granted, WC is transferred from memory to the WC counter in the BTC. The terminate bit (bit 0) in the WC word is tested and a latch is set if the terminate bit is a ONE. This signifies that no more block transfers are to be made until commanded by this particular BTC after completion of the current transfer. (The terminate bit cancels the automatic re-initialization feature.)

The maximum time for the initialization procedure is five cycles for the CP-I/OP. Three cycles are normally used for the execution of the CEU instruction when transferring to/from private memory, and two cycles are used to transfer the contents of the two dedicated memory locations to the BTC. Five additional cycles are normally required when the SM-I/OP is used - four for transferring the command word to the SM-I/OP dedicated location and one additional cycle for execution of the TAC instruction.

After initialization, data is transferred between the selected peripheral unit and memory over the CP-I/OP data lines or the SM-I/OP data lines under the joint control of the BTC, the block priority control (BPC) and the access control circuits in the central processor or the SM-I/OP. A word transfer is initiated by the peripheral unit which sends a data transfer request signal to the BTC. This signal causes the BTC to request a memory cycle through the access control logic. When the access control logic determines that a memory cycle can be granted, a memory available signal is generated and sent to the BTC. The BTC, in turn, sends a signal to the peripheral unit which causes it to connect to the processor data lines, execute the word transfer, and then disconnect from the data lines. After completion of a word transfer, the CWA value is incremented and the WC value is decremented in the counters in the BTC. The entire data block is transferred by this process which is always initiated by the peripheral unit.

When the value of WC is decremented to zero, the block transfer is completed. If the terminate latch in the BTC is not set, a new block transfer will be initiated. Re-initialization consists of acquiring new initial values of FWA and WC from the dedicated BTC location in private or shared memory.

After re-initialization, a priority interrupt is generated which signifies that the transfer of the last block is completed and a new block transfer is initialized. The interrupt processing routine can then store (in the dedicated locations) the FWA

and WC values for the next block transfer any time prior to the completion of the current block transfer. This re-initialization technique reduces the problem of re-initializing block transfers under program control between the occurrence of two successive words in a continuous data stream.

If the Terminate latch in the BTC had been set by the Terminate bit in the last WC word acquired from memory, an interrupt is generated when the value of WC is decremented to zero and no new transfer is initiated by the BTC. In addition, the data transfer request line from the peripheral unit is disconnected until a new initialize signal is received.

BTC PRIORITY AND TIMING

BTC units are granted memory cycle requests on a priority basis. The priority ordering function is performed by a BPC unit located in the CP-I/OP and SM-I/OP. A unique priority number is assigned to each BTC. The BPC logic is structured similar to the priority interrupt logic, insuring that higher priority BTC requirements are always serviced before lower priority units. However, once a word transfer is initiated, it will not be interrupted by a request from a higher priority BTC. The BPC will instead store the request and, when the current word transfer is completed, will service the stored requests on a priority basis.

BTC requests for memory cycles always take precedence over cental processor requests. If the particular device data transfer rate is high enough, the BTC could lock out the central processor. The value of the SM-I/OP becomes evident here. The central processor can initialize the SM-I/OP by the execution of a single instruction, and proceed with its program execution while the BTC units in the SM-I/OP are controlling data transfers to/from shared memory at the maximum rate.

The maximum collective transfer rate for a BTC (or group of BTC's) is 572,000 words per second. Cycle stealing (or lockout) from the program is automatic and each BTC word transferred removes one cycle from the program. The BTC can gain access to the memory after a delay of one cycle except during the time of execution of the IMS, IAM, and AAM instructions which create a two-cycle delay; and the I/O instructions (CEU, AOP, AIP, MOP, MIP, and TEU) which create a three-cycle delay.

SECTION V
PRIORITY INTERRUPT SYSTEM

GENERAL DESCRIPTION

The 840MP Computer can have up to 61 levels of priority interrupts. With the exception of two special interrupts (power fail safe and stall alarm), each level can be selectively enabled and disabled under program control. Three levels of interrupt are supplied with the basic 840MP Computer. Additional optional levels are available in modules of 15 levels each with the exception of group 0, which contains 13 optional levels. Table 5-1 lists the available optional interrupt levels.

Table 5-1. Optional Interrupt Levels

Module	Number	Levels
1	12	Levels 1-13, Group 0
2	15	Levels 1-15, Group 1
3	15	Levels 1-15, Group 2
4	15	Levels 1-15, Group 3

Assignment of interrupts is highly flexible. Internal processor functions such as overflow and memory parity can be connected to interrupt levels. BTC re-initialization and end of block signals, as well as signals from external peripherals or acquisition systems, are connected to the levels which best suit the particular system requirement. In systems using the SEL 840MP Real-Time Monitor, the interrupt levels, other than special external system levels, are fixed by the monitor requirements.

A special interrupt, power fail safe/restore is assigned the highest priority in the system. A second special interrupt, the optional stall alarm, has second priority in the system. Power fail safe/auto start and stall alarm (when installed) are always enabled. Interrupt signals at these levels override all other processor functions, including halt, I/O hold, and indirect address chaining.

A unique location in private memory is assigned to each interrupt level. Power fail safe/auto start and stall alarm are assigned locations 100_8 and 101_8, respectively. Location 102_8 is assigned to the third highest priority level, location 103_8 to the fourth highest level, etc. Table 5-2 shows the assignment of interrupt locations.

The main program may be interrupted by any level in the system provided that the interrupt level has been previously enabled, and that no higher level interrupt level is active.

If a higher level interrupt is active when an interrupt signal occurs, the interrupt will be stored until the completion of the higher level interrupt processing routine. The lower level routine will then be initiated. It will continue until completed or until interrupted by a higher level interrupt signal. In this case, the lower level routine will be completed after completion of the higher level routine. Program control will then be returned to the original point of interrupt. The priority logic enables any number of interrupt levels to be active at the same time. Routine execution is always performed in the order of priority of the active interrupts.

By means of a special instruction, external interrupt (EXI), one 840MP Computer may interrupt another. The address field of this instruction enables each 840MP Computer to generate up to six interrupt signals. These may be connected to the interrupt levels of any 840MP Computer in the same system and used as a means of control communication.

INTERRUPT ROUTINE PROCESSING

When a priority interrupt signal is recognized by the 840MP Computer, an EXU L (execute the instruction contained in location L) is performed. This execution is an automatic function of the 840MP Computer hardware. L is the address of the memory location assigned to the interrupt level (see table 5-2). By storing a store, place and branch (SPB) instruction inL, a linkage is provided to any effective virtual address in the first 8192 words of memory. The SPB stores the contents of the program counter, plus one (the next sequential instruction in the interrupted program), at the entry point of the interrupt processing routine. This provides a means of returning to the main program at the point of

Table 5-2. Interrupt Level Memory Location Assignments

Memory Location (Octal)	INTERRUPT ASSIGNMENT		Memory Location (Octal)	INTERRUPT ASSIGNMENT		Memory Location (Octal)	INTERRUPT ASSIGNMENT	
	Level	Group		Level	Group		Level	Group
100	Power Fail Safe/Restore		125	5	1	153	11	2
			126	6	1	154	12	2
101	Stall Alarm		127	7	1	155	13	2
102	2	0	130	8	1	156	14	2
103	3	0	131	9	1	157	15	2
104	4	0	132	10	1	161	1	3
105	5	0	133	11	1	162	2	3
106	6	0	134	12	1	163	3	3
107	7	0	135	13	1	164	4	3
110	8	0	136	14	1	165	5	3
111	9	0	137	15	1	166	6	3
112	10	0	141	1	2	167	7	3
113	11	0	142	2	2	170	8	3
114	12	0	143	3	2	171	9	3
115	13	0	144	4	2	172	10	3
116	14 (Std.)	0	145	5	2	173	11	3
117	15 (Std.)	0	146	6	2	174	12	3
121	1	1	147	7	2	175	13	3
122	2	1	150	8	2	176	14	3
123	3	1	151	9	2			
124	4	1	152	10	2	177	15	3

the interrupt. The SPB instruction also stores the contents of BAR 0 and the status of the overflow latch in the interrupt routine entry location. If the system is equipped with an extended arithmetic unit (EAU) and the program protect option, the EAU mode status (double-precision or floating-point) and the status of the protect latch are also stored in the interrupt routine entry location. This information is stored in the interrupt routine entry location in the format illustrated in figure 5-1.

NOTE

The SPB instruction stores the contents of BAR 0, the status of the overflow and protect latches, and the EAU mode status only upon entry into an interrupt processing routine. At any other time in the interrupt routine (or main program), the SPB instruction stores the address bits (9-23) only.

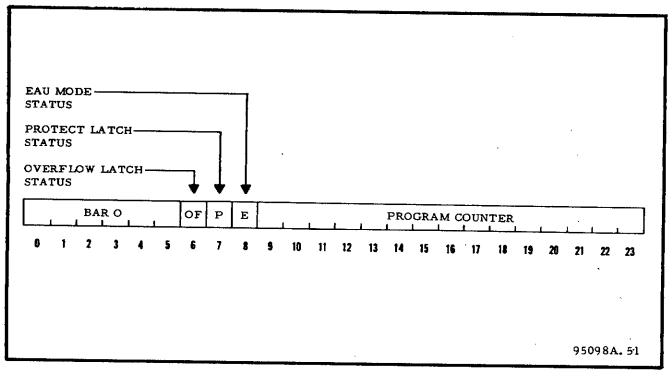


Figure 5-1. Word Format - Interrupt Routine Entry Location

After storing the BAR 0 contents and the status of the overflow latch (and the protect latch and EAU mode statuses, if installed), the contents of BAR 0 are cleared and the overflow (and EAU mode) latch(es) is (are) reset. (See Section VI for a description of the protect latch operation.)

Upon completion of the interrupt routine entry, three important requirements have been satisfied:

a. A means for returning to the main program has been provided.

b. A means for re-establishing the operational condition (overflow, etc.) at the time of interrupt has been furnished.

c. The BAR and major control latches have been freed for use during the processing of the interrupt.

The interrupt processing routine may now be executed, using the BAR's and control latches in any desired fashion. All locations in private or shared memory may be accessed, except for the entry and bank switching locations.

At the end of the interrupt routine, a priority interrupt return (PIR) instruction is executed. The execution of the PIR causes the contents of the interrupt routine entry location (figure 5-1) to be stored in the program counter and BAR 0, and restores the original status of the overflow latch in the SEL 840MP Computer and restores the EAU mode. (See Section VI for a description of the protect latch operation.)

INTERRUPT CONNECTION

Three levels of priority interrupts are supplied with the basic 840MP Computer. One interrupt, power fail safe/auto start, occupies the highest priority in the system and is always connected and enabled. The two remaining standard interrupt levels are assigned to levels 14 and 15, group 0. The two standard interrupt level assignments allow optional levels (table 5-1) of both higher and lower priorities to be made available. The two standard levels are connected through the CP-I/OP cables to peripheral units. Interrupt signals in each unit are connected to one or the other of the two levels under program control.

Connection of a unit interrupt to a priority level is performed by execution of the CEU instruction which initiates data flow in the unit. The format of the second word of the CEU instruction for the peripheral units contains three bits used to connect and disconnect the unit interrupts. The interpretation of the three bits is given in table 5-3.

Table 5-3. Standard Interrupt CEU Bit Functions

Bit	Bit State	Function
1	ONE	Connect levels designated in bits 2 and 3
1	ZERO	Disconnect levels designated in bits 2 and 3
2	ONE	Connect/Disconnect level 14, group 0
2	ZERO	Leave level 14, group 0 in its present state
3	ONE	Connect/Disconnect level 15, group 0
3	ZERO	Leave level 15, group 0 in its present state

Appendix C lists the interrupt signals that are connected to levels 14 and 15, group 0, from each standard peripheral unit by means of a CEU instruction.

The user may arrange the interrupt systems to suit the particular system requirements through the use of the optional interrupt levels. Interrupts can be arranged so that they may be connected/disconnected under program control or permanently connected to peripheral units, permitting certain peripherals to communicate with the processor strictly on a priority interrupt basis.

INTERRUPT ENABLING / DISABLING

Interrupt levels may be selectively enabled and disabled by the priority interrupt enable (PIE) and

priority interrupt disable (PID) instructions. An entire group (up to 15 levels), or any number of levels within a given group, may be enabled or disabled by the execution of a single instruction. The format of the PIE and PID instructions is illustrated in figure 5-2.

The group field (bit positions 0-1) is binarily coded, group 00 being the highest priority group. The level field (bits 9-23) is unitarily coded, a ONE in bit 23 signifying the highest priority level within a group (level 1). An instruction which will cause the five highest levels in group 2 to be enabled is written in assembly language as:

```
PIE     '37,2
```

Execution of this instruction leaves the 10 lower levels within group 2 (if present) unaffected. They remain either enabled or disabled.

A special interrupt card is available which provides an alternate means of effectively disabling interrupts. When this card is inserted at any interrupt level, the level may be made active under program control by execution of a PIE instruction specifying that level. Making the special level active inhibits any lower level from becoming active. Therefore, the lower levels are effectively disabled. Execution of a PID instruction addressing the special level causes the active (A) latch for that level to be reset.

If the special interrupt card is inserted at the highest level (group 0, level 1), a means is provided for keeping higher priority levels from becoming active while lower priority levels are being processed. An interrupt processing routine at any level may be executed without interruption by having a PIE 1, 0 immediately after the entry location and a PID 1, 0 immediately before the exit (PIR) location. The real-time monitor requires one special interrupt card.

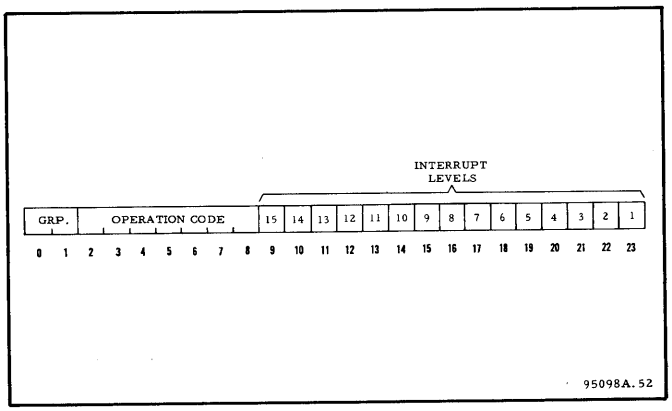


Figure 5-2. Word Formats - PIE and PID Instructions

INTERRUPT LEVEL LOGIC

The logic for each interrupt level consists of three latches and associated gates. The latches are designated request (R), enable (E), and active (A). The R latch is set by the external interrupt signal. It provides storage for the signal until the level becomes active and the interrupt routine is executed. The R latch is reset by execution of the PIR (Priority Interrupt Return) instruction, which normally terminates each interrupt routine.

The R latch is also reset for all designated levels each time the PID instruction is executed. The E latch is set by the proper group and level bits in the PIE instruction. It is reset by the same bit pattern in the PID instruction. The logic expression for the signal (in) which interrupts the mainframe from interrupt level n is

$$IN = Rn\ En\ \overline{(A_1 + A_2 + \text{---} + An\text{-}1)}$$

where A_1 is the active signal for the highest interrupt level. The mainframe tests for the presence of any "In" signal while fetching each instruction from memory, but prior to execution. If "In" is present, "An" is set and the instruction

```
EXU    100 + n (Group 1)
```

is executed in one machine cycle time. The A latch is reset by the signal generated by the execution of the PIR instruction. Each time this instruction is executed, both the highest level A and R latches are reset, releasing the channel to accept a new interrupt signal.

INTERRUPT ROUTINE PROGRAMMING

The basic structure of an assembly language interrupt processing routine is illustrated in figure 5-3.

If it is desired that higher level interrupts be disabled while a lower level (that is, level 4, group 0) is processed, then the interrupt routine entry and exit should be similar to that illustrated in figure 5-4.

The SEL 840 Multiprocessor Computer System does not permit the interruption of the SEL 840MP Computer immediately after the execution of the following instructions:

```
EXU    (Execute the instruction at location n)
SPB    (Store, Place and Branch)
PIE    (Priority Interrupt Enable)
PID    (Priority Interrupt Disable)
CSB    (Copy Sign of B-Accumulator)
```

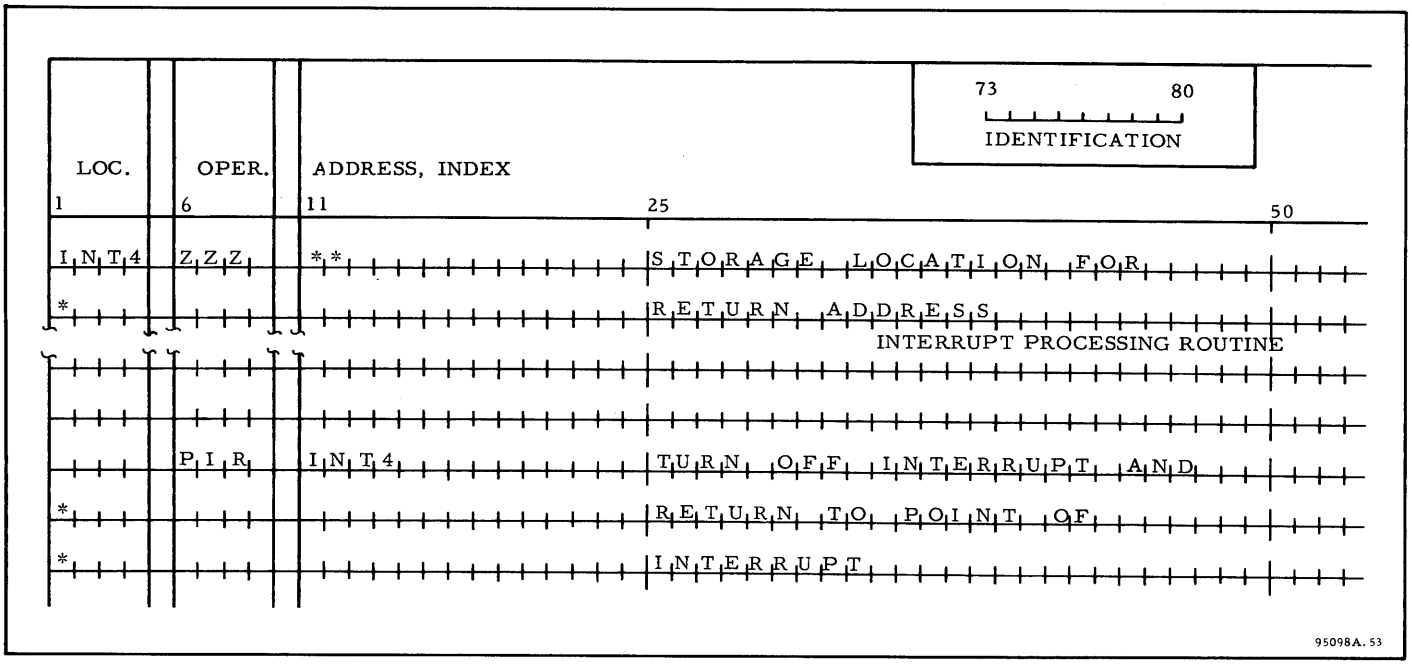


Figure 5-3. Sample Assembly Language Interrupt Processing Routine

An interrupt is not permitted immediately after an SPB instruction in order to allow a PID instruction to be executed if desired. Thus, if desired, an interrupt routine may always be completed by disabling higher level interrupts in the manner indicated in the second example.

Interrupts are not processed after a PIE instruction to guarantee that the exiting interrupt routine may be completed without interrupt. This provision enables the same interrupt routine to be entered by two or more interrupt levels and assures that the return linkage will not be overlaid.

The CSB instruction is the fifth non-interruptable instruction. This provision assures that the AMA instruction which normally follows a CSB can be executed before an interrupt is processed. Therefore,

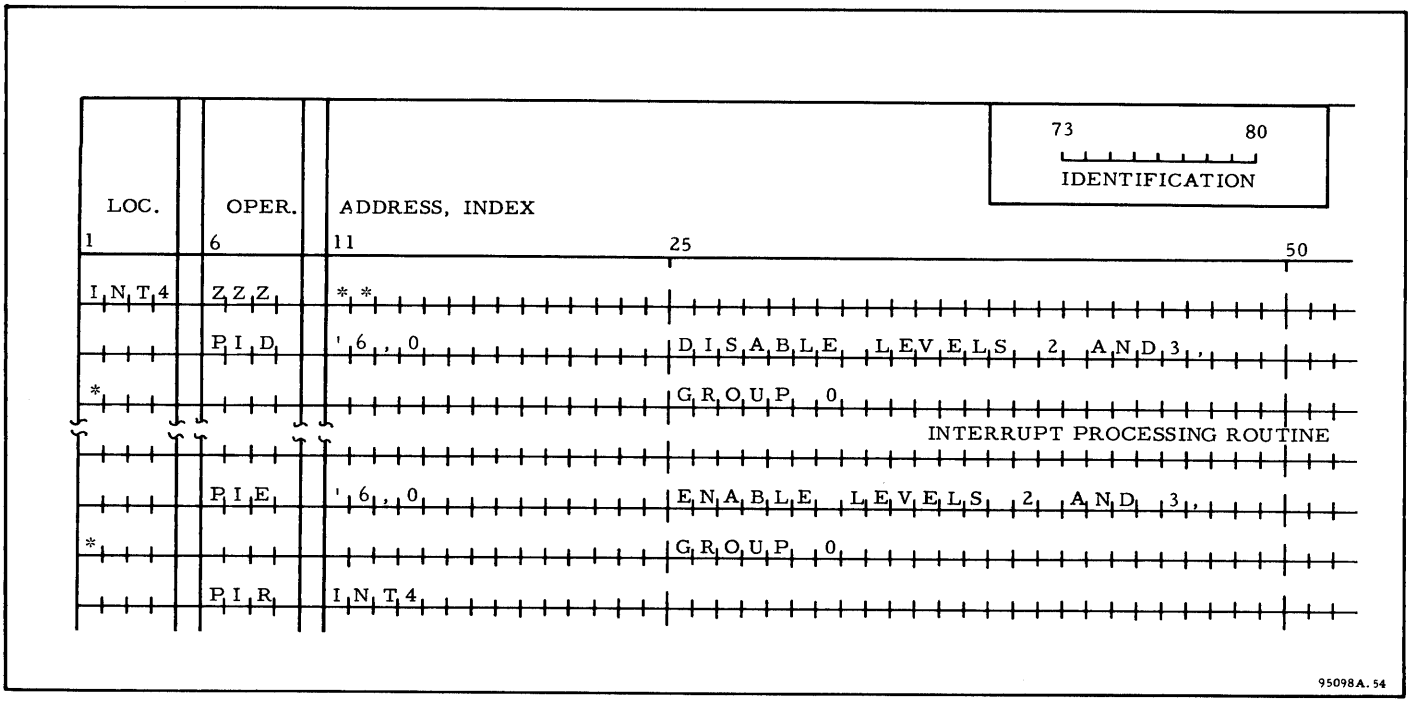


Figure 5-4. Sample Assembly Language Interrupt Entry and Exit Routine

the contents of the carry latch may be changed by an interrupt routine at no risk of affecting the interrupted program.

The PID instruction should be used with caution. This instruction clears as well as disables the designated interrupt levels. The instruction pair

```
PID   X,Y
PIE   X,Y
```

may be used to clear and enable a selected group of interrupts. Execution of this instruction pair insures that interrupt signals which occurred prior to the time of enabling will not be processed. However, when the PID instruction is used in an interrupt processing routine, only higher level interrupts should be disabled in the routine to prevent the chance of inadvertently clearing unprocessed interrupts at lower levels. It is impossible to clear an enabled interrupt at a higher level because the action of the priority logic insures that higher level interrupts are processed before any lower interrupt is recognized.

When multiple interrupt signals are connected to the same level, the typical interrupt routine contains the test instructions that determine which signal occurred.

SECTION VI
PERIPHERAL DEVICES

GENERAL

This section contains brief descriptions of the standard peripheral devices available for use with the 840 Multiprocessor Computer System. Although a brief discussion of physical and functional characteristics is included, the emphasis is placed on the programming aspects of the devices.

CONSOLE INPUT / OUTPUT STATION (DEVICE NO. 1)

The standard console I/O station supplied with the 840MP Computer is a modified Teletype Model ASR-33 send/receive typewriter. The modifications introduced by Systems Engineering Laboratories include the interface and control electronics that increase the flexibility of the basic device when used with the 840MP Computer. The console I/O station consists of a typewriter keyboard and page printer plus an eight-level paper tape punch and reader. The complete unit is integrally mounted at the console. A manual switch is provided to enable either on-line operation with the 840MP Computer or off-line independent operation.

When the console I/O station is operated on-line, the input devices (keyboard and reader) operate separately from the output devices (printer and punch). As a result, the paper tape reader can operate independently at the rate of 20 characters per second. However, the keyboard and the paper tape reader cannot be used simultaneously. If printing and punching of input characters are desired, each character read into the 840MP Computer is transferred back to the output devices under program control. Both printing and punching occur unless the punch is turned off manually. Printing and punching can be performed at rates up to 10 characters per second.

The console I/O station is not equipped with BTC capabilities, and therefore, cannot be used in operations involving an SM-I/OP.

The major specifications for the console I/O station are listed in table 6-1.

Table 6-1. Console I/O Station Specifications

Characteristic	Specification
Paper Tape Input Speed	20 characters per second
Keyboard Input Speed	10 characters per second (maximum)
Output Speed	10 characters per second for printing and punching
Character Code	ASCII
Number of Printable Characters	63
Characters per Line	72
Paper Tape	Standard one-inch roll
Power	115 vac, 60 cps, single phase

OPERATION AND PROGRAMMING

The console I/O station interface circuitry contains two character buffers, one for input and one for output. The presence of two buffers plus the functional separation of the input and output devices enables both input and output data transfers to occur, both at the maximum rate. For example, a paper tape can be read at 20 characters per second while an independent set of characters is printed at 10 characters per second. The two character buffers can be connected to the two standard peripheral unit interrupt levels under program control. If the input buffer is connected to the interrupt system, it will generate an interrupt at level 14, group 0 each time it receives a character from the keyboard or paper tape reader. If the output buffer is connected to the interrupt system, it will generate an interrupt at level 15, group 0 each time it is emptied and is ready to receive a new character from the processor. The output buffers store characters for printing and punching.

The CEU commands sent to the console I/O station select the desired input mode and connect and disconnect the two interrupts. The single output mode

is selected by the on-line switch and requires no CEU command execution. As a result, any time an AOP or MOP instruction having a unit 1 address is executed, a character will be printed. The character will also be punched if the punch is turned on. The bit coding for the CEU second word function codes is shown in table 6-2. Bit numbering corresponds to the bit positions (0-23) in an 840MP Computer word.

Table 6-2. Console Input/Output Station CEU Function Bit Coding

Function	Bit = One
Priority Interrupt Connect	1
Processor Input Interrupt	2
Processor Output Interrupt	3
Enable Reader Mode	4
Enable Keyboard Mode	5
Clear Input Mode	6

Bits 1-3 permit the two console I/O station interrupts to be selectively connected to the two standard peripheral unit interrupt levels. The input interrupt, used with keyboard and reader entry, is connected to PI, level 14, group 0 by execution of the instruction:

```
CEU     1
DATA    '30000000
```

It is disconnected by execution of the instruction:

```
CEU     1
DATA    '10000000
```

The corresponding instructions which connect and disconnect the output mode interrupt to PI, level 15, group 0 are:

```
Connect     CEU     1
            DATA    '24000000

Disconnect  CEU     1
            DATA    '04000000
```

The connect commands must be preceded by a PIE instruction to enable the central processor to be interrupted. Execution of the input mode select commands (enable reader mode, enable keyboard mode, and clear input mode) cause the input buffer to be cleared as well as the specified input mode to be selected. Therefore, when transferring between the reader and keyboard modes, it is not necessary to clear the input mode. The clear input mode command allows for turning off reader and keyboard modes simultaneously.

NOTE

Interrupt commands and input mode commands can be mixed, providing timing problems do not occur. However, only one input mode command bit at a time should be used.

Reader operation is under complete control of the 840MP Computer program. The reader control operates on an automatic character call-up philosophy. Data (a single character) is transferred from the reader (or keyboard) to the 840MP Computer by the execution of an AIP or MIP instruction. As a result of each character transfer, action is initiated to start calling up the next character on the tape. This feature simplifies the programming by turning the reader on and off (with a CEU instruction) only once for each group of characters desired.

NOTE

Each time a character is read, the tape advances, and the character cannot be read again. The programmer must insure that a CEU to clear the reader mode (select keyboard mode or select mode clear) and turn the reader off is generated within 10 milliseconds after each desired group of characters is accepted by the 840MP Computer. Otherwise, the next character on the tape will be read, stored in the input data buffer, and subsequently lost when the buffer is cleared or loaded with new data.

No test (TEU instruction) unit hardware is provided on the console I/O station. Although a test for input busy is built into each CEU instruction, it may become desirable to perform this test without attempting to command the unit. To accomplish this, a CEU to unit 1 with no bits set in the second word will skip when not busy without changing the unit.

For the most part, the console typewriter input section operates independently under one unit number. For reference, the approximate busy times are as follows:

Reader 50 milliseconds*

* First 10-12 milliseconds of each cycle left open to allow turn-off.

Keyboard: 100 milliseconds

Output Devices: 100 milliseconds

The reader control switch (START/STOP/FREE) is not used for on-line operation and can be left in either the START or the FREE position while running.

Useful keyboard features include the following keys:

a. Here Is: Punches all zeros. Can be used to generate leader or spaces for off-line operation. Recommended way of pulling tape into punch. Will not input into computer.

b. Rub Out: Punches all ones.

c. Repeat: Repeats characters as long as both are held down.

d. Line Feed, Carriage Return: Self-explanatory.

e. Control: Used to generate special functions written on top of character buttons. Can be used to generate the ringing of the TTY bell.

NOTE

The TTY bell is a convenient audible alarm that can be activated by the program.

The correspondence between the SEL 840MP Computer bit positions and paper tape levels is shown in figure 6-1.

A programming example for keyboard input and printer output is illustrated in figure 6-2. This routine transfers three characters from the keyboard to the SEL 840MP Computer's A-Accumulator, packs them in the B-Accumulator, and stores them in one location in memory.

PAPER TAPE READER—MODEL 84-510A (DEVICE NO. 2)

The SEL Model 84-510A paper tape reader reads eight-level, one-inch paper or mylar (opaque) tapes at asynchronous speeds up to 300 characters per second. Specifications for the paper tape reader are listed in table 6-3.

Table 6-3. Paper Tape Reader Specifications

Characteristic	Specifications
Speed	300 characters per second
Levels	8
Operation	Asynchronous start/stop
Size	19 inches wide x 7 inches high x 24 inches deep
Sensing	Photocell
Drive	Pinch roller
Power	115 volts, 60 Hz ±10%, 1.4 amps nominal
Temperature	10°C to 35°C
Controls	Power on/off; run/load

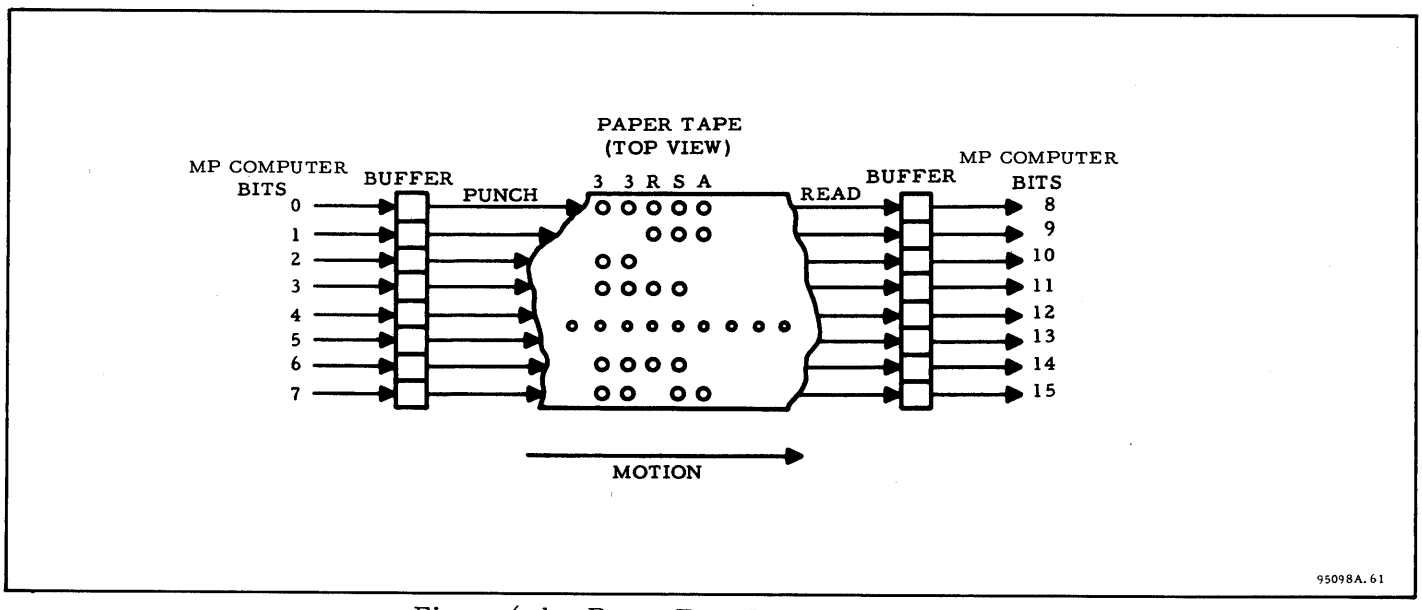


Figure 6-1. Paper Tape Data Flow Diagram

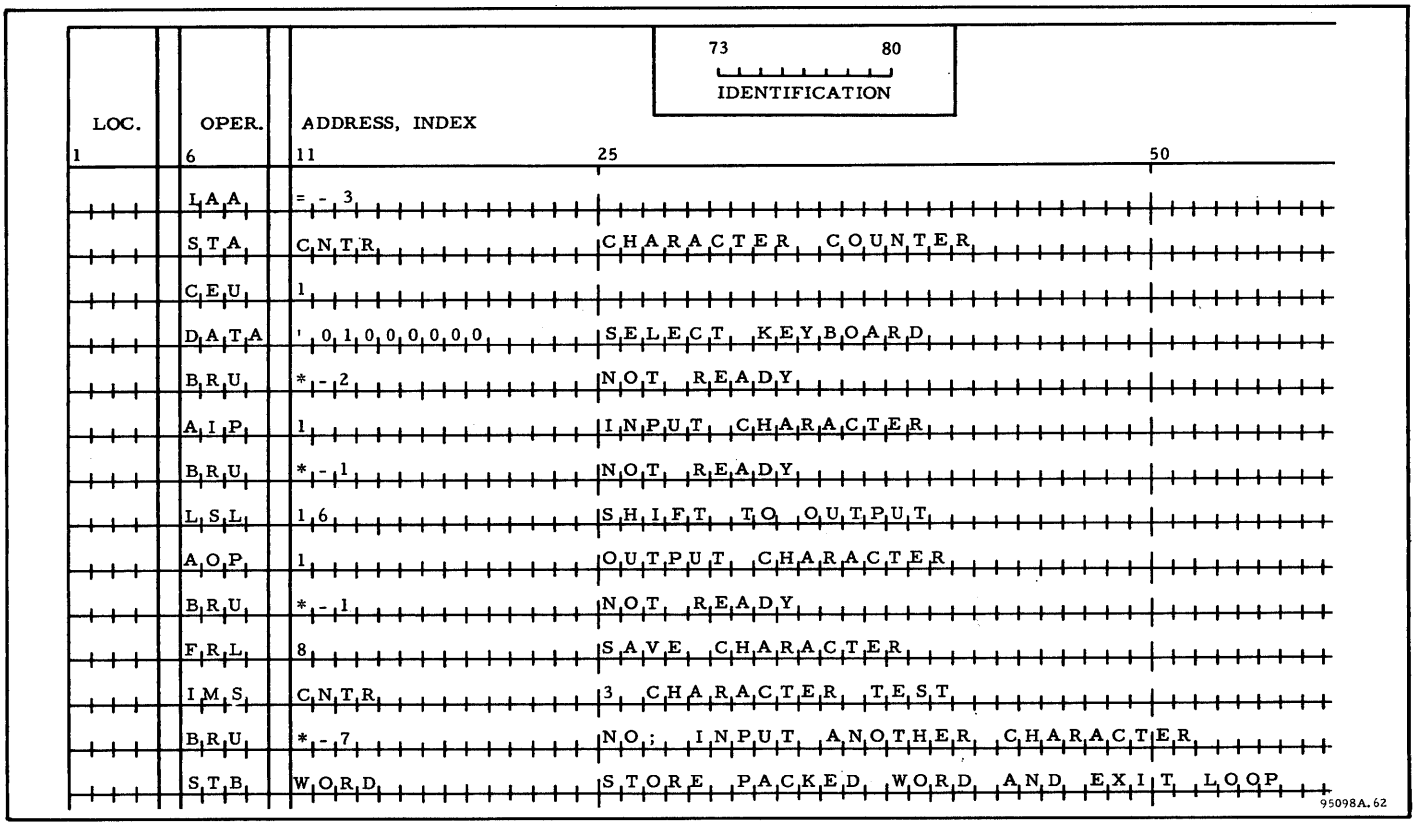


Figure 6-2. Sample Program For Console Input/Output Station

PAPER TAPE PUNCH—MODEL 84-520A (DEVICE NO. 2)

The SEL Model 84-520A paper tape punch punches eight-level paper or mylar tapes at speeds up to 110 characters per second. A sprocket hole is punched with each character. Power may be turned on or off under program control. Specifications for the paper tape punch are listed in table 6-4.

Table 6-4. Paper Tape Punch Specifications

Characteristic	Specifications
Speed	110 characters per second
Levels	8
Operation	Asynchronous start/stop
Size	19 inches wide x 14 inches high x 24 inches deep
Punches	High-carbon steel pins
Drive	Sprocket
Power	115 volts, 60Hz ±10%, 2 amps nominal
Temperature	10°C to 35°C
Controls	Power on/off; line feed

Other features include built-in tape storage reel, tape cutter and transparent chad box.

PAPER TAPE SPOOLER—MODEL 80-530A

The SEL Model 80-530A paper tape spooler supplies and spools 5, 6, 7, or 8 level paper tapes at rates up to 400 characters per second. The tape is rewound by a manually-controlled switch at a 1000 characters per second rate. No command or test functions from the processor are required. Specifications for the paper tape spooler are listed in table 6-5.

Table 6-5. Paper Tape Spooler Specifications

Characteristics	Specifications
Feed Speed	Up to 400 characters per second
Rewind Speed	1000 characters per second
Tape Levels	5, 6, 7, or 8, interchangeable
Reel Hubs	Standard NAB reel dimensions
Reel Diameter	8 inches O. D.
Reel Capacity	400 feet of 4 mil tape
Interlock	Tape break or no tape

Table 6-5. Paper Tape Spooler Specifications Cont'd

Characteristic	Specifications
Mounting	Upright
Size	19 inches wide x 10-1/2 inches high x 8-5/8 inches deep
Power	115 volts, 60 Hz, ±10%
Temperature	5°C to 45°C
Controls	Power on/off; rewind

HIGH-SPEED PAPER TAPE SYSTEM—MODEL 84-525A (DEVICE NO. 2)

The SEL Model 84-525A high-speed paper tape system consists of a high-speed photoelectric reader and a high-speed punch. The reader is capable of reading 6, 7, or 8 level paper or mylar (opaque) tape at rates up to 300 characters per second. The punch punches eight-level tape at a maximum rate of 110 characters per second. If both the punch and the reader are given the same unit number, they can be considered, from a programming standpoint, as one unit with both input and output capabilities. Since they each have a separate buffer, they can be operated at maximum speed simultaneously.

OPERATION AND PROGRAMMING

There are two CEU commands that can be given the paper tape reader. These commands are reader enable and reader disable. When the reader enable command is issued, the buffer will be filled with a character and the tape will advance one character position (this will occur only if the buffer is initially clear). While the reader is enabled, the tape will advance one position and the buffer will be filled each time an AIP or MIP is serviced by the reader. When the reader disable command has been issued, the reader will not advance when an AIP or MIP is serviced.

There are two CEU commands that can be given the punch. They are punch power on and punch power off. The Punch Power ON command must be given before outputting to the punch. After punch power has been turned on, the punch will punch the tape and advance one character position each time an AOP or MOP is executed.

There are two standard interrupts furnished with the SEL High-Speed Paper Tape System. One is the Buffer Ready interrupt from the punch and the other is the Buffer Ready interrupt from the reader.

A sample program, for copying paper tape by using the SEL High-Speed Paper Tape System, is illustrated in figure 6-3.

CARD READER—MODEL 84-450A-400 CPM (DEVICE NO. 4)

The SEL Model 84-450A Card Reader reads standard 80-column punched cards at a maximum rate of 400 cards per minute. Reading is column by column. The card reader specifications are listed in table 6-6.

OPERATION AND PROGRAMMING

The card reader responds to the feed card and reader stacker offset commands. The feed card signal initiates a card reading cycle by actuating the feed mechanism. Data is read from the card in a column by column format. Data is transferred from the reader to the processor by executing an AIP or MIP instruction. The reader stacker offset command causes the card currently being read to be slightly elevated as it enters the stacker. This feature allows easy access to a card for indexing.

Figure 6-4 illustrates an example of card reader programming. This sample routine stores 80 characters packed into 40 locations. The routine is called by the sequence.

```
L    SPB    CARD
     DAC    IBUF
```

Table 6-6. Card Reader Specifications

Characteristics	Specifications
Speed	400 cards per minute
Read Mechanism	Star wheels
Hopper Capacity	450 cards
Dimensions	18 inches wide x 14 inches high x 11-1/4 inches deep
Power	115 volts or 230 volts, ±10%, 50 or 60 Hz, ±3 Hz

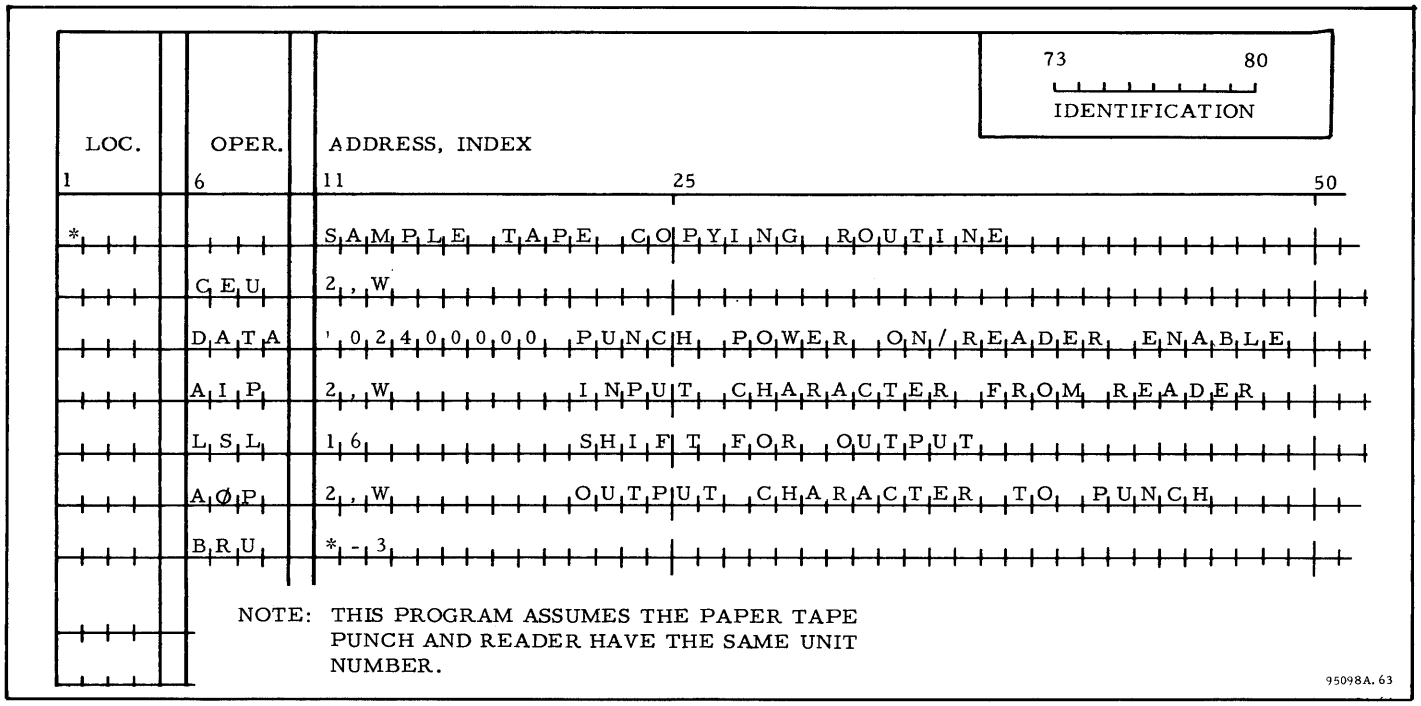


Figure 6-3. Sample Program for Copying Paper Tape Using High-Speed Paper Tape System

HIGH-SPEED LINE PRINTER—MODEL 80-730 SERIES (DEVICE NO. 5)

The three models of line printers differ only in printing speed as follows:

Model 84-733A, 300 lines per minute

Model 84-731A, 600 lines per minute

Model 84-732A, 1,000 lines per minute

The printers use plain or preprinted standard perforated multipart, fan-folded paper stock. Horizontal formatting is computer controlled while vertical format may be determined by either computer commands or a vertical control loop. Specifications for the SEL 80-730A Series Line Printers are listed in table 6-7.

Table 6-7. Line Printer Specifications

Characteristic	Specifications
Printing Speed	300, 600, or 1,000 lines per minute
Characters per line	120 maximum
Paper Width	4 inches to 20 inches
Printing Area	Horizontal - 10 characters per inch Vertical - 6 lines per inch
Type Font	Open Gothic
Number of Characters	64
Code Wheel Sensing	Photocell
Vertical Form Control	8-channel tape loop
Power Requirements	115 volts, 60 Hz
Dimensions	56 inches wide x 54 inches high x 30 inches deep
Controls	Power on/off Top of form Master Clear Single space

OPERATION AND PROGRAMMING

The line printer responds to various commands from the central processor and may be tested for various functions by the central processor. The commands or tests may be initiated by the execution of a CEU or TEU in the SEL 840MP Computer, or by using the command or test word (during the execution of a TAC instruction by the central processor) stored in the I/OP dedicated location. Table 6-8 lists the line printer commands

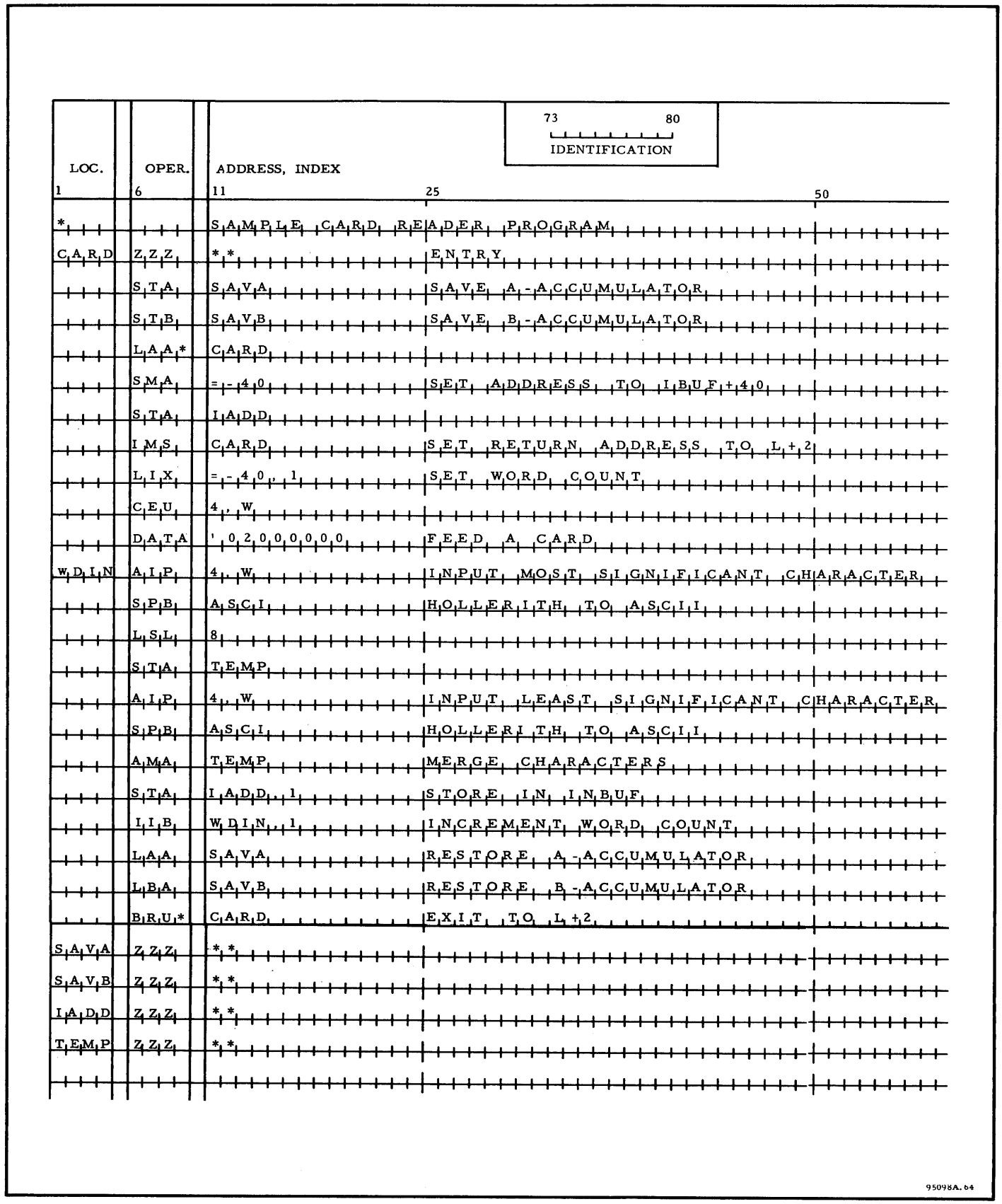


Figure 6-4. Sample Program For Card Reader

Table 6-8. Line Printer Command and Test Functions

FUNCTION BITS				Command/Test
CEU	TEU	I/OP Command Word	I/OP Test Word	
0				Initialize BTC (Optional)
4				Paper Advance to Loop Channel (n)
5				Paper Advance One Line
6				Paper Advance to Top of Form
7				Print; or Format Tape Channel No. if Bit 4 = ONE
16				Clear Buffer; or Format Tape Channel No. if Bit 4 = ONE
17				Fill Buffer; or Format Tape Channel No. if Bit 4 = ONE
21				Transfer Current Word Address (Used with BTC)
	4			Test for Busy
	6			Test for Parity Error
	16			Test for Bottom of Form
	17			Test for Printer Inoperable
		0		Initialize BTC (Required)
		4		Paper Advance to Loop Channel (n)
		5		Paper Advance One Line
		6		Paper Advance to Top of Form
		7		Print; or Format Tape Channel No. if Bit 4 = ONE
		16		Clear Buffer; or Format Tape Channel No. if Bit 4 = ONE
		17		Fill Buffer; or Format Tape Channel No. if Bit 4 = ONE
		21		Transfer Current Word Address
			4	Test for Busy
			6	Test for Parity Error
			16	Test for Bottom of Form
			17	Test for Printer Inoperable

and tests that may be performed. The table also lists the function bits that must be set in the second word of the CEU or TEU, or in the I/OP command or test word, in order to execute the commands or perform the tests.

The SEL 80-730 Series Line Printers contain a 120 character buffer. This buffer must be loaded in the order that the characters are to appear on the printed line. If the entire buffer is not to be filled, it must be cleared prior to loading. When

a command to clear the buffer is issued, ZERO's are loaded into the buffer by the line printer logic. The printer logic will not reply to an attempt to transfer more than 120 characters and will stall permanently if the 121st transfer is attempted while using the wait flag with the data transfer instruction.

The line printer will accept simultaneous commands as long as they are not in logical or mechanical conflict. For example, the buffer cannot be commanded to clear and fill at the same time. Also, no two mechanical commands such as print and advance paper one line can be executed simultaneously. The advance to format N command should never be combined with another command.

There are three modes for advancing paper on the line printer. One mode is to advance the paper one line. Another is to advance the paper to the top of form, which is the logical top of page (where the printing is to begin). The third mode is to advance to format N. This may be only one line or as much as a full page, depending on where the next punch is in the channel specified by N.

An example of line printer programming is illustrated in figure 6-5. This routine will cause one line to be printed.

MAGNETIC TAPE SYSTEM—MODEL NO. 84-600 SERIES (DEVICE NOS. 6 AND 7)

The SEL 84-600 Series Magnetic Tape System consists of from one to eight tape transports coupled to the processor by means of a tape control unit (TCU). The seven-track units are IBM 729-compatible and the nine-track units are IBM 2400-compatible.

The SEL Model 84-615A Magnetic Tape Transports use 1/2-inch mylar tape and have either seven or nine tracks. Specifications for the transports are given in table 6-9.

The TCU provides the interface between the processor and the tape transports. The SEL Model 84-610-78A TCU controls seven-track transports and the Model 84-610-98A TCU controls nine-track transports.

The TCU status panel contains the following indicators:

a. TRACKS - For display of the contents of the read register or write register in the seven (or nine) track positions.

b. TRANSPORT NUMBER SELECTED - Displays which one of the eight (maximum) transports has been selected for use with the TCU.

c. CRC ERROR (Optional on a nine-track system only) - Indicates the detection of a check register character error.

d. READY - Indicates the ready condition of the selected transport.

e. WRITE STATUS - Indicates the execution of write commands.

f. READ STATUS - Indicates that write commands are not being processed and therefore, the transport is ready to accept read commands.

Table 6-9. Magnetic Tape Transports Specifications

Model Number	Tape Speed	Start or Stop Time*+	MINIMUM GAP SPANNING TIME*+	
			7 Track	9 Track
84-615-07	45 ips	9 msec.	25.4 msec.	21.7 msec.
84-615-09	75 ips	6 msec.	16.4 msec.	14.2 msec.
84-615-11	120 ips	3.8 msec.	10.2 msec.	8.9 msec.
84-615-12	150 ips	3.5 msec.	9.2 msec.	8.1 msec.

*Nominally +20 percent

+Time between the writing of the last character in one record to the writing of the first character in the next record.

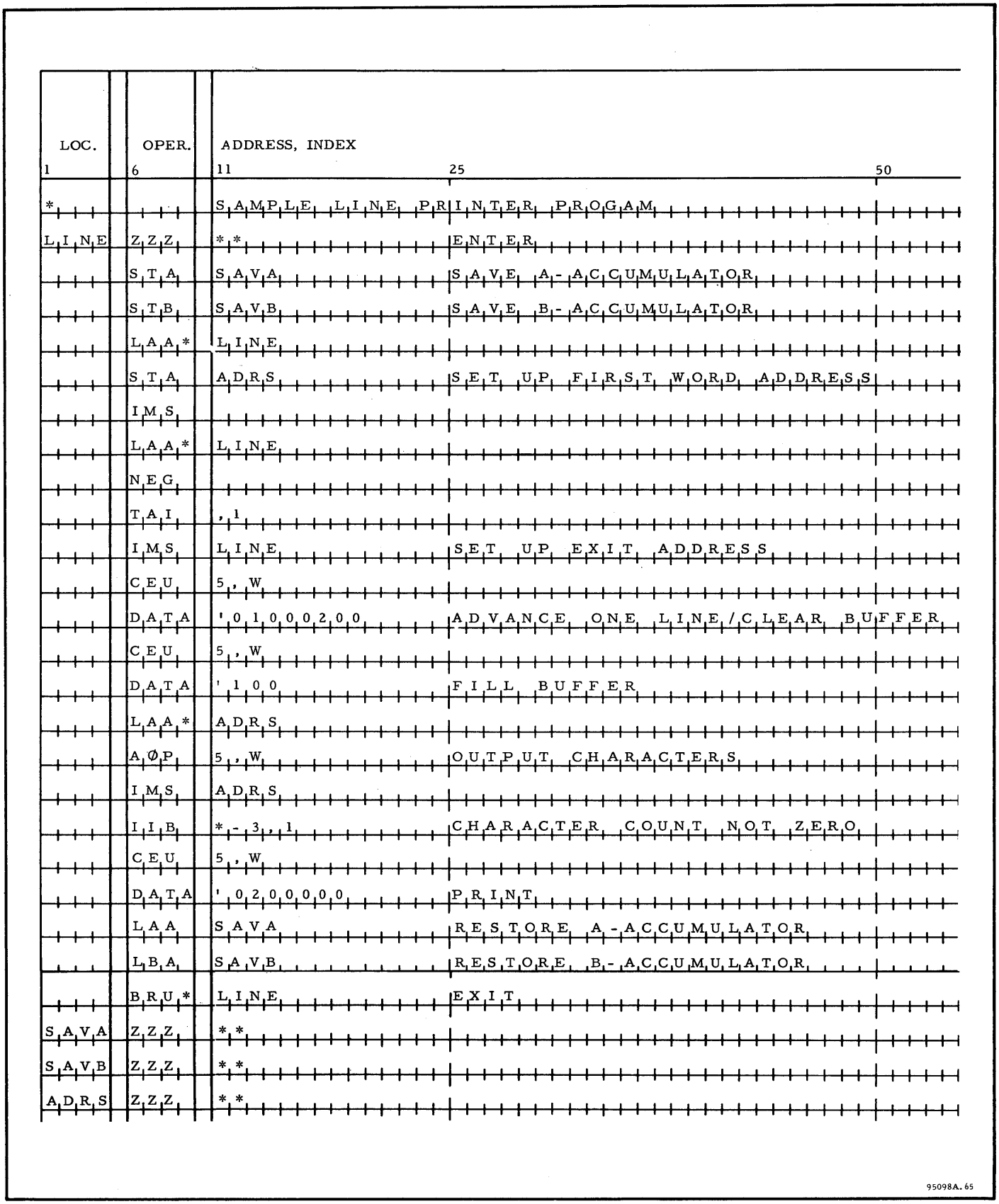


Figure 6-5. Sample Program For Line Printer

g. BUSY - Indicates that the selected transport is active.

h. EOF - Indicates the detection of end of file during read or write.

i. EOT - Indicates that end of tape has been sensed by the selected tape transport.

j. LD PT - Indicates that the selected transport has been positioned at the load point.

k. LAT PARITY ERROR - Indicates the detection of a lateral parity error.

l. LONG PARITY ERROR - Indicates the detection of a longitudinal parity error (longitudinal redundancy check character error).

m. BIN - Indicates that binary data is to be transferred.

n. BCD - Indicates that binary-coded-decimal data is to be transferred.

o. DENSITY (200/556/800) - Indicates the selected tape packing density in bits per inch.

p. CHAR/WORD (1/2/3/4) - Indicates which characters per word grouping has been selected by the program.

OPERATION AND PROGRAMMING

MAGNETIC TAPE SYSTEM COMMANDS

In order to represent all of the magnetic tape system functions, two command word formats are used. These two formats are called Format 0 and Format 1. The desired format is selected by placing ZERO (Format 0) or a ONE (Format 1) in bit position 4 of the CEU second word or the SM-I/OP command word. More than one magnetic tape system function may be executed with one CEU instruction (or I/OP command) provided the selected functions are not in mechanical or logical conflict with each other.

Format 0

The initial tape operation command must use the format 0 command word. Format 0 functions include the selection of binary or BCD data for transfer, the selection of the desired tape packing density, the specific transport to be used in the data transfer, and the number of characters per word to be used. These functions must be performed in every format 0 command word. Certain control functions (that is, rewind, erase four inches of tape, transfer BTC current word address) may also be performed with a format 0 command word. Table 6-10 lists the functions of the various command word function bits for format 0.

Table 6-10. Magnetic Tape Commands, Format 0

Bit*	State	Function
0	ZERO	Not used. Must be set to ZERO.
1	ONE	Connects the interrupts designated by bits 2 and 3. (If bits 2 and 3 are zero, the state of bit 1 is ignored.)
1	ZERO	Disconnects the interrupts designated by bits 2 and 3. (If bits 2 and 3 are ZERO, the state of bit 1 is ignored.)
2	ONE	Selects word transfer ready interrupt for connection/disconnection to/from standard interrupt level.
3	ONE	Selects end of record interrupt for connection/disconnection to/from standard interrupt level.
4	ZERO	Selects the format 0 command word.
**5	ONE	Rewinds selected tape transport. Bits 18-20 must contain desired transport number.
**6	ZERO	Erases 4 inches of tape on the transport selected by bits 18-20.
7	ONE	Sets up the TCU to transfer BCD data with even parity.
7	ZERO	Sets up the TCU to transfer binary data with odd parity.
8-15		Not used with CEU. Used only in I/OP words to specify command or test operations and unit numbers.

*Corresponds to the bit positions in the CEU second word or the I/OP command word.

**These commands should never be used simultaneously.

Table 6-10. Magnetic Tape Commands. Format 0 (Cont'd)

Bit*	State	Function
16 and 17		Select tape packing density. 00_2 = 200 bits/inch; 01_2 = 556 bits/inch; 10_2 = 800 bits/inch.
18-20		Select tape transport (0-7).
21	ONE	Transfers the BTC current word address into the corresponding dedicated location.
22 and 23		Select characters per word. 01_2 = 1 character/word; 10_2 = 2 characters/word; 11_2 = 3 characters/word; 00_2 = 4 characters/word.

*Corresponds to the bit positions in the CEU second word or the I/OP command word.

**These commands should never be used simultaneously.

Figure 6-6 illustrates the magnetic tape format for a format 0 command word.

Format 1

Format 1 command word functions are listed in table 6-11. Care must be taken to assure that only logical bit combinations are used.

At the termination of an erase four inches of tape, write record, or write end of file operation, the TCU is left in a write status. All other operations will leave the TCU in a read status.

MAGNETIC TAPE SYSTEM TESTS

Tests are performed to determine the status of a given TCU. The tests may be performed by executing a TEU instruction in the SEL 840MP Computer, or as the result of an SM-I/OP test operation. In the SEL 84-600 Series Magnetic Tape Systems, all status conditions are reset by each motion command issued to the TCU. The transport tested is the one set up (format 0 command) at the time the test is performed. The test codes are set up to skip the next sequential instruction on a positive result. More than one condition may be tested by a single test word; however, any one condition producing a negative result will inhibit the skip and cause the next sequential instruction to be executed. In this case, the program would not be able to determine which condition produced the negative test result. Table 6-12 lists the magnetic tape tests that may be performed by controlling the state of the bit positions in the test word.

USE OF THE BLOCK TRANSFER CONTROL

The automatic BTC unit is an optional central processor I/O control unit which is normally associated with high-speed and/or high-density tape handling equipment. When using the magnetic tape system in conjunction with an SM-I/OP, a BTC unit must be employed. For an operational description of

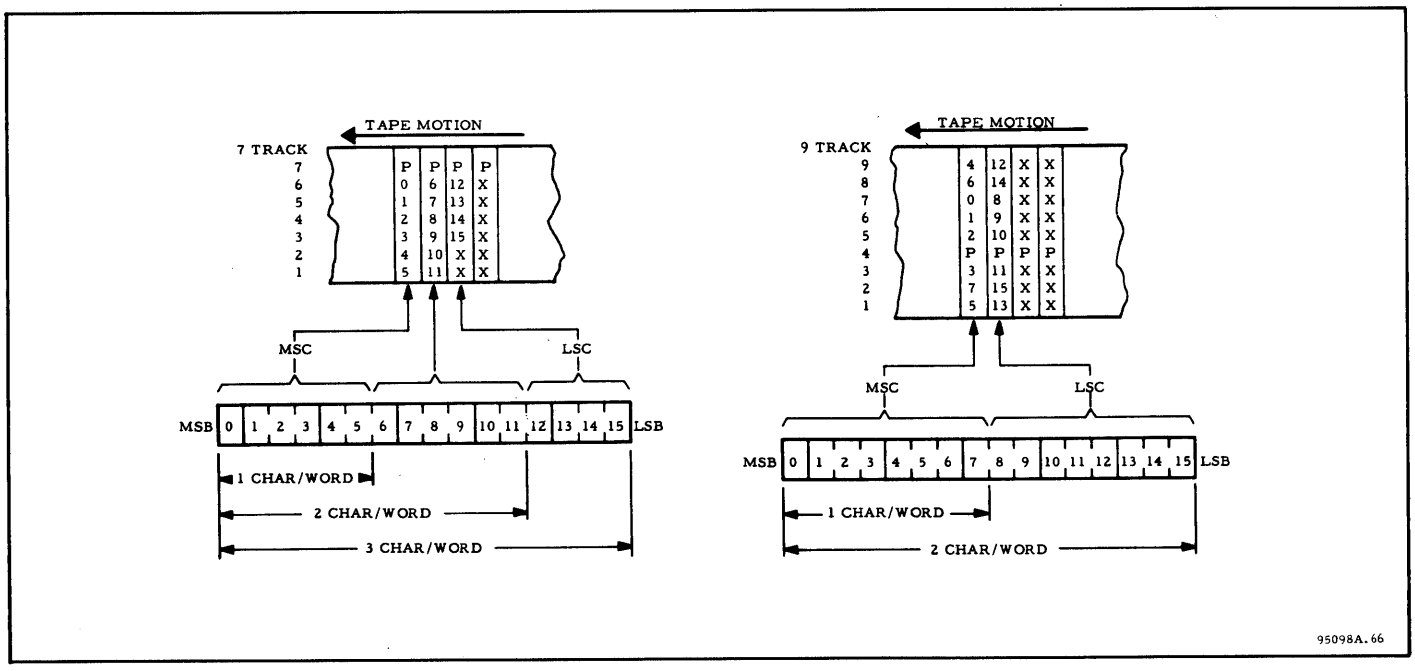


Figure 6-6. Magnetic Tape Format 0 Data Word

Table 6-11. Magnetic Tape Commands, Format 1

Bit*	State	Function
0	ONE	Initializes BTC unit. (Bit 0 must not be ONE if bit 21 is ONE.)
1	ONE	Connects the interrupt designated by bits 2 and 3. (If bits 2 and 3 are ZERO, the state of bit 1 is ignored.)
1	ZERO	Disconnects the interrupt designated by bits 2 and 3. (If bits 2 and 3 are ZERO, the state of bit 1 is ignored.)
2	ONE	Selects word transfer ready interrupt for connection/disconnection to/from standard interrupt level.
3	ONE	Selects end of record interrupt for connection/disconnection to/from standard interrupt level.
4	ONE	Selects the format 1 command word.
**5	ONE	Write record - The tape will move forward as the information being sent from the processor replaces the previous information on the tape. When this information ceases to come from the processor, a longitudinal check character is written and a record gap is generated. In the case of a 9-track system, the CRC character is written 4 character times previous to the LRCC character, and follows the last data character by 4 character times.
**6	ONE	Approximately 3-1/2 inches of tape are erased and an EOF mark is written.
**7	ONE	Read record - The tape will move forward to the next record gap leaving the data on the tape undisturbed. When the tape is in motion, the data is transferred to the processor as it is encountered on the tape. This transfer will continue to take place until the next record gap is reached or until the processor stops requesting the data. If the processor stops requesting the data before the next record gap is encountered, a data overflow condition exists. This condition may be tested. ***
8-15		Not used with CEU. Used only in I/OP words to specify command or test operations and unit numbers.
**16	ONE	Advances tape forward one record, leaving the read/write heads in the middle of the next record gap. The data on the tape is undisturbed. ***
**17	ONE	Advances tape one file, leaving the read/write heads in the middle of the record gap following the next end of file (EOF) mark. The data on the tape is undisturbed. ***

*Corresponds to the bit positions in the CEU second word or the I/OP command word.

**Only one motion command may be used in a single command word.

***These commands should not follow an erase four inches of tape (format 0), write record, or write end of file operation.

Table 6-11. Magnetic Tape Commands, Format 1 (Cont'd)

Bit*	State	Function
**18	ONE	Backspaces one record, leaving the read/write heads in the middle of the previous record gap. The data on the tape is undisturbed.
**19	ONE	Backspaces one file, leaving the read/write heads in the gap preceding EOF mark. The data on the tape is undisturbed.
20		Not used.
21	ONE	Transfers BTC current word address into the corresponding dedicated location. (Must not be used when bit 0 is a ONE.)
22 and 23		Not used.

*Corresponds to the bit positions in the CEU second word or the I/OP command word.

**Only one motion command may be used in a single command word.

Table 6-12. Magnetic Tape Test Functions

Bit*	State	Function
0	ONE	Skip on not busy - When the TCU is capable of executing a motion command, the next instruction is skipped. If the TCU is not capable of executing a motion command, the next instruction is executed.
1	ONE	Skip on no end of file - When EOF is not present, the next instruction is skipped. When EOF is present, the next instruction is executed.
2	ONE	Skip on no overflow - When no overflow condition exists, the next instruction is skipped. When an overflow condition is present, the next instruction is executed. (An overflow occurs when the data request from the processor is dropped before an end of record gap is reached on the tape. Overflow occurs only during a read operation.)
3	ONE	Skip on load point - If the read/write heads are positioned at the load point, the next instruction is skipped. If the read/write heads are not positioned at the load point, the next instruction is executed.
4	ONE	Skip on end of record interrupt - If the magnetic tape end of record generated an interrupt, the next instruction is skipped. If the end of record did not cause an interrupt, the next instruction is executed.
5	ONE	Skip on no parity error - When no parity error is present, the next instruction is skipped. When a parity error is present, the next instruction is executed.
6	ONE	Skip on write ring in - When the write ring (or file protect ring) is in the tape reel, the next instruction is skipped. When the write

*Corresponds to the bit positions in the TEU second word or the I/OP test word.

Table 6-12. Magnetic Tape Test Functions (Cont'd)

Bit*	State	Function
6 (cont'd)		ring is not in the reel, the next instruction is executed. (The write ring must be in the reel in order to write on the tape. The absence of the write ring prevents the destruction of data on tapes, such as library tapes.)
7	ONE	Skip on no end of tape - If the end of tape mark has not been sensed, the next instruction is skipped. If the end of tape mark has been sensed, the next instruction is executed. (The tape transports will not stop when the end of tape mark is sensed.)
8-15		Not used with TEU - Used only in I/OP words to specify command or test operations and unit numbers.
16	ONE	Skip on rewinding - If the transport that is selected is mechanically rewinding, the next instruction is skipped; if it is not rewinding, the next instruction is executed. When a rewind command is issued to the TCU, the rewind status does not exist until the mechanical motion has started. Therefore, before disconnecting a tape transport which has been given a rewind command, it is necessary to test for the rewind status. Once the mechanical motion has started, the rewinding will continue while another tape transport is being commanded.
17	ONE	Skip on CRC error - If no cycle redundancy check error exists, the next instruction is skipped. If a cycle redundancy check error exists, the next instruction is executed. (Nine track system only.) The nine-track tape system, when writing tape, generates a cycle redundancy character (CRC) and writes it on the tape after each record. When the tape is being read back into the computer, the CRC is generated again and is compared with the one written on the tape. If the two characters do not compare, a CRC error exists and can be tested for with a TEU instruction or SM-I/OP test.
18-23		Not used.

*Corresponds to the bit positions in the TEU second word or the I/OP test word.

the BTC, refer to Section IV. All magnetic tape control units can be used with BTC.

USE OF WORD TRANSFER (NO BTC)

Word transfer is only used when data is being transferred between the SEL 840MP Computer and a peripheral unit connected to the CP-I/OP. This method of transfer is normally associated with low-speed and/or low-density tape handling equipment. When using this method, the data is transferred by a series of AIP or MIP (input), AOP or MOP (output) instructions. When writing an end of record is generated when data flow to the tape unit ceases. When reading, EOR is sensed in the normal manner. Care must be taken when transferring data to the TCU without the BTC to ensure that the data is available to the TCU in the time required by the speed of the transport and the tape packing density. When reading, the processor must accept the data as it is read or it will be lost.

USE OF INTERRUPTS

There are two standard interrupts available with SEL 84-600 Series Tape Systems. One interrupt is an end of record interrupt that occurs when a EOR is written or sensed. The second standard interrupt available is the word interrupt which is used if no BTC is available. In the condition of

output, or writing on the tape, the TCU will interrupt (starting with the second word) anytime its word buffer is ready to receive data. In the condition of input, or reading from the tape, the TCU will interrupt (starting with the first word) anytime its word buffer is ready to send data. Other interrupts such as EOF interrupt, parity error interrupt, end of tape interrupt, and information overflow interrupts are optionally available.

PROGRAMMING EXAMPLE

Figure 6-7 illustrates a sample routine for programming the magnetic tape system. The execution of this routine will cause one record to be read from tape and will terminate the BTC on completion of the transfer.

X-Y INCREMENTAL PLOTTERS—MODELS 84-810 AND 84-812 (DEVICE NO. 11)

The SEL Models 84-810A and 84-812A Incremental Plotters are high-speed digital, two-axis plotters. The actual plot is produced by the movement of a pen over the surface of the chart paper. Specifications for the plotters are listed in table 6-13.

OPERATION AND PROGRAMMING

The y-axis plot is produced by lateral movements of the pen carriage and the x-axis plot by rotary motion of the chart drum. Provisions for z-axis modulation is also incorporated. Six basic plotter movements can be performed under program control. These are:

Pen Up

Pen Down

Drum Up

Drum Down

Carriage Right

Carriage Left

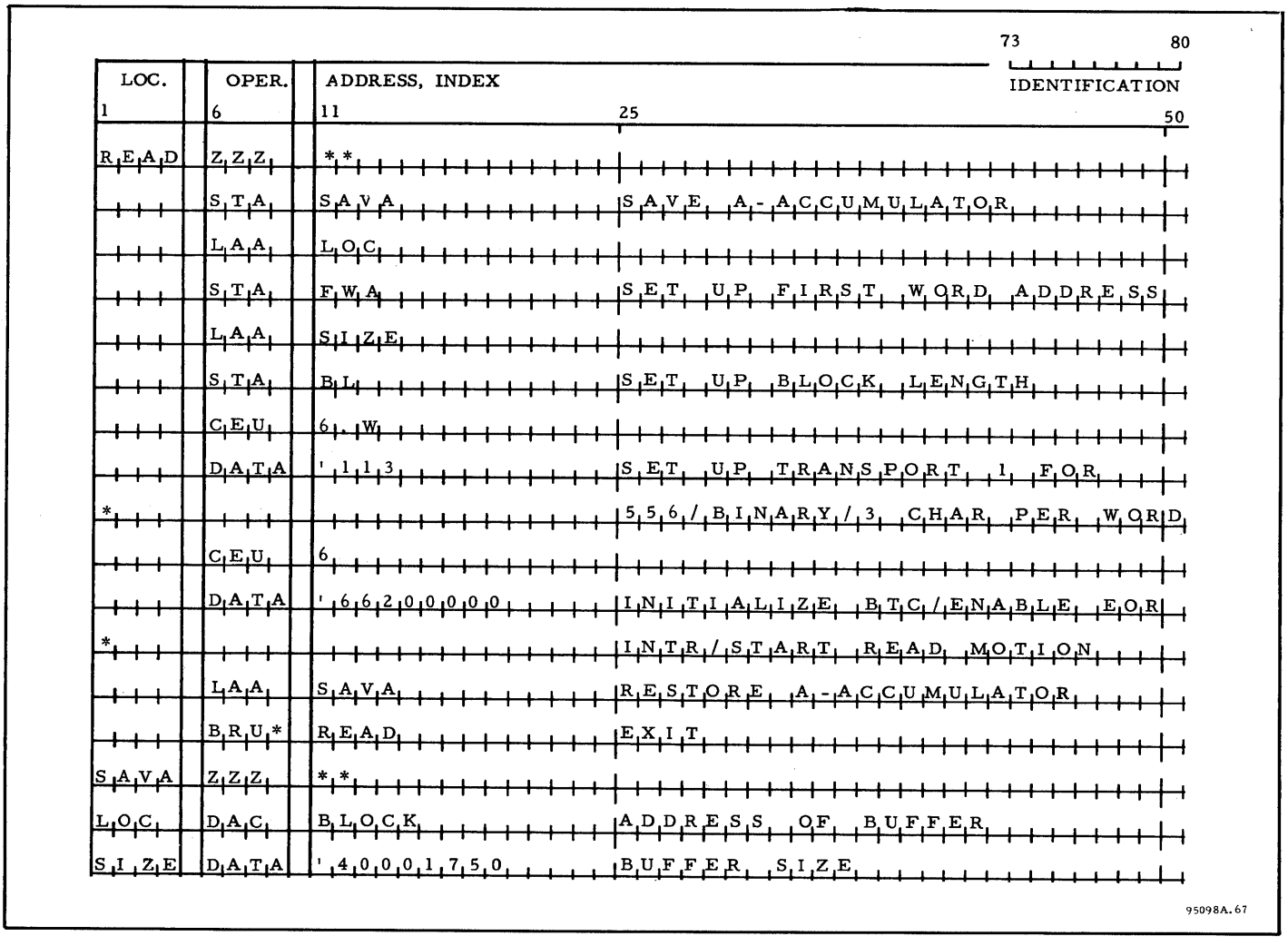


Figure 6-7. Sample Program For Magnetic Tape System

Table 6-13. X-Y Plotter Specifications

Characteristic	Model 84-812A	Model 84-810A
Chart Width	31 inches	12 inches
Speed (X, Y direction) (Pen up/down)	12,000 steps per minute 600 per minute	18,000 steps per minute 600 per minute
Resolution	0.01 inches	0.005 inches
Plot Width	29.5 inches	11 inches
Chart Drive	Sprocket	Sprocket
Power	115 volts, ±10%, 60 Hz, 1.6 amps nominal	
Temperature	10°C to 35°C	10°C to 35°C
Manual Controls	Drum forward/reverse	
	Carriage right/left	
	Pen up/down	
	Single step and continuous modes	
	Power on/off	

The basic movements on each axis (x and y) are at 0 and 90 degrees. The x, y (drum, carriage) combinations yield 45-degree movements. The plotter can be controlled manually (off-line) on a continuous or single-step (incremental) basis. Under processor control (on-line), only incremental movements are provided.

Control of the x-y plotter is established by means of the CEU instruction. Each CEU instruction execution sequence causes the specified command to be executed by the plotter for one predetermined increment. Table 6-14 lists the x-y plotter commands and the CEU second word coding for each command.

Figure 6-8 illustrates a sample x-y plotter programming routine. This routine plots an octagon (using all the possible movements, including Pen Up and Pen Down) at a predetermined scale factor. The A-Accumulator is loaded with the scale factor. This routine lowers the pen, draws the octagon, and raises the pen before halting. The calling sequence is:

```
SPB     CDIA
```

Table 6-14. X-Y Incremental Plotter Commands

Coding (CEU Second Word)	Command
'1000000	Pen Up
'2000000	Pen Down
'200000	Drum Up (x-)
'400000	Drum Down (x+)
'100	Carriage Right (y-)
'200	Carriage Left (y+)
'200100	Drum Up, Carriage Right (x-, y-)
'400100	Drum Down, Carriage Right (x+, y-)
'200200	Drum Up, Carriage Left (x-, y+)
'400200	Drum Down, Carriage Left (x+, y+)

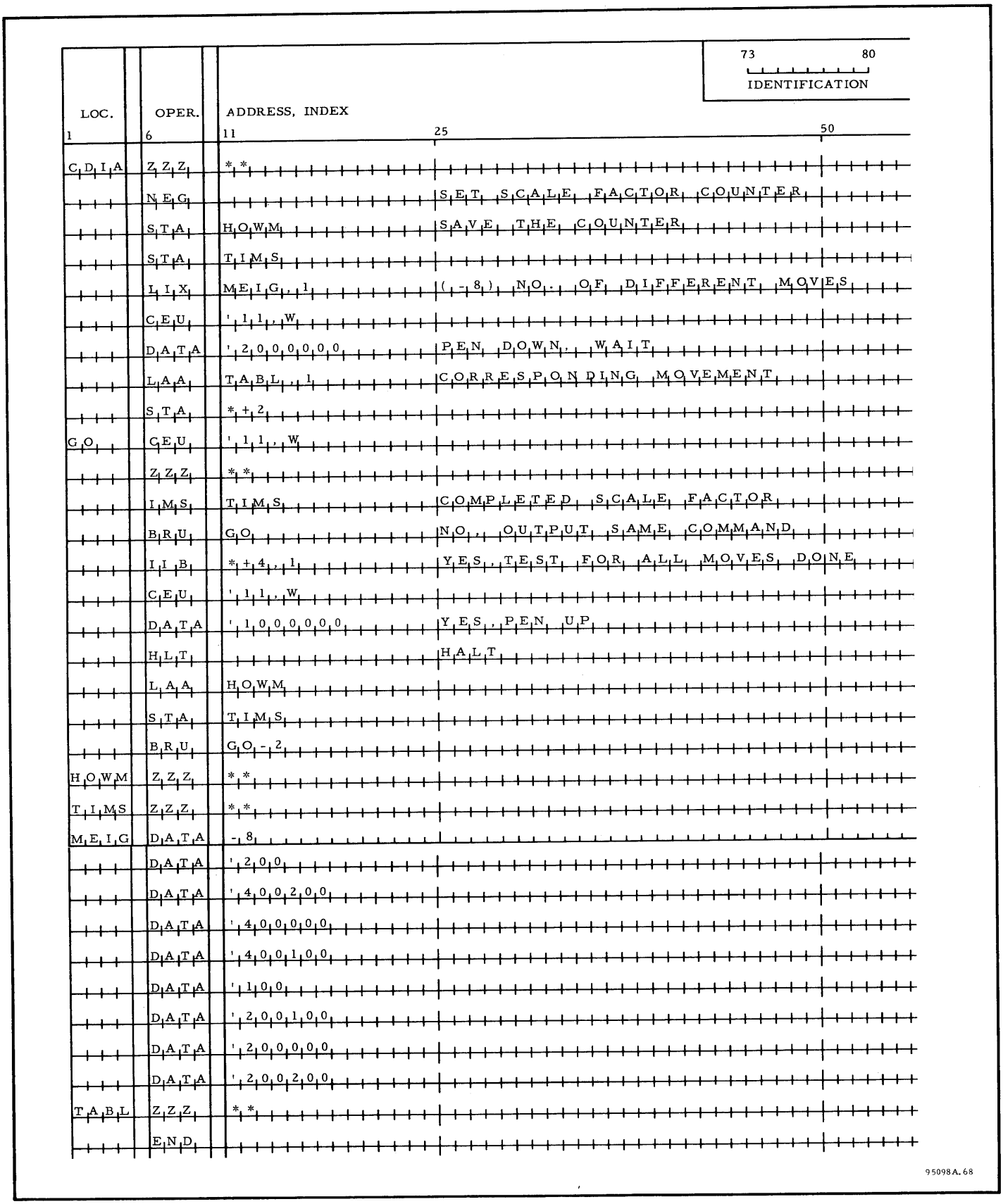


Figure 6-8. Sample Program For X-Y Plotter

MOVABLE HEAD DISC STORAGE—MODEL 84-653A (DEVICE NO. 13)

The SEL Model 84-653A Disc Storage System consists of a disc control unit (DCU) and a disc storage drive. The disc storage system is a random-access bulk storage device with a storage capability of 1,024,000 24-bit words. The disc is subdivided into tracks, heads, and sectors. Each recording surface of the disc is accessed by a movable head which can be moved to any of 100 tracks. Each track contains 16 sectors. Figure 6-9 shows the track and sector layout of a recording surface.

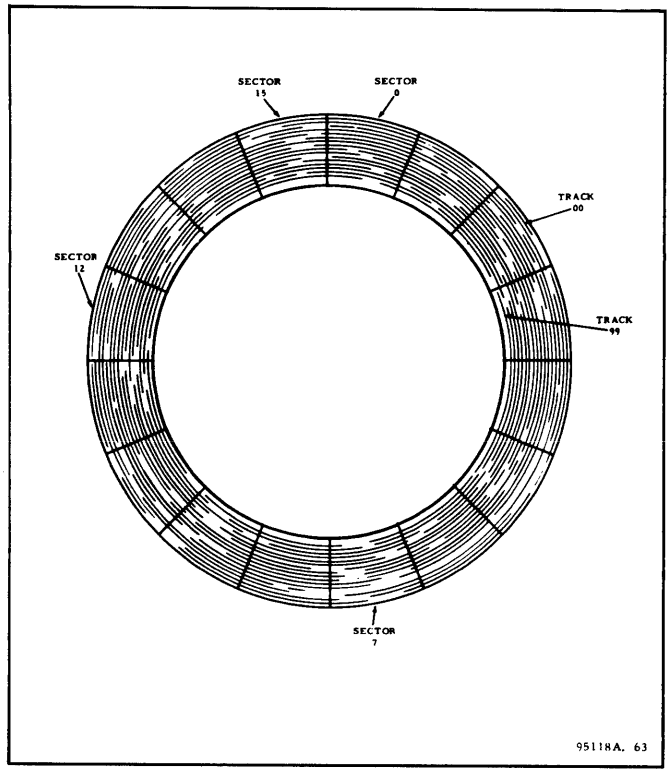


Figure 6-9. Track and Sector Layout

The 10 recording surfaces of the disc pack are addressed by the movable head assembly. Each surface is read/written by an individual head. Figures 6-10 and 6-11 illustrate the head arrangement in relation to the recording surface.

Each sector will store 64 words; thus, each track will store 1,024 words and an entire recording surface will store 102,400 words. Since the head assembly is arranged so that the same track number is addressed on each recording surface, each position of the heads on a track can be viewed as a cylinder (see figure 6-11). Considering a disc pack as 100 cylinders, 10,240 words can be written/read without moving the head assembly. The disc rotates at 2400 rpm. This gives a maximum

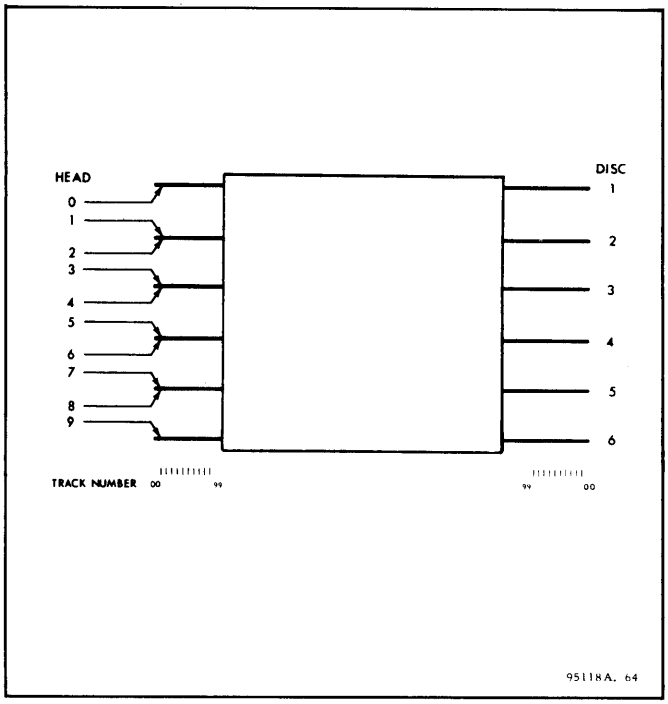


Figure 6-10. Movable Head Arrangement - Recording Surface

latency time of 25 milliseconds. Figure 6-12 shows the time required to move the head n positions. The data transfer rate of the disc storage system is 52,083 Hz, or one word every 19.4 microseconds.

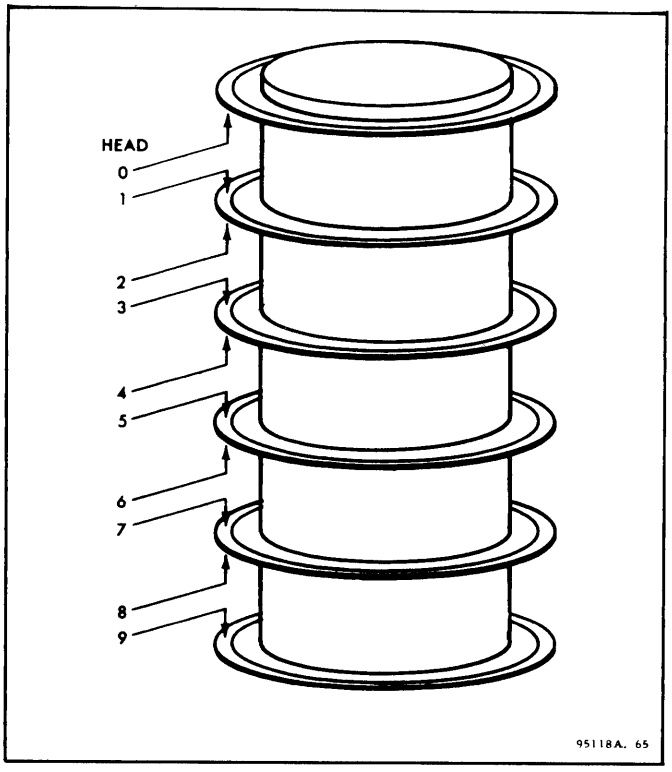


Figure 6-11. Head Position

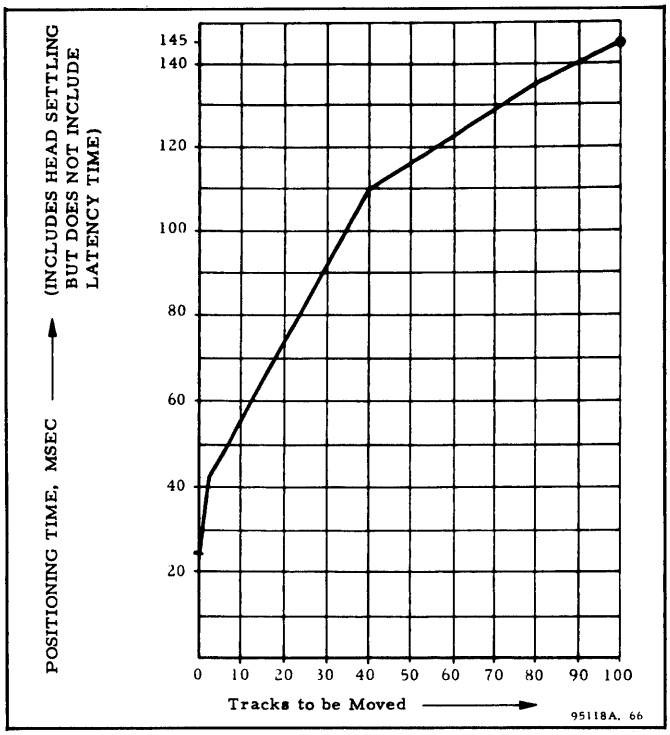


Figure 6-12. Typical Head Positioning Time

OPERATION AND PROGRAMMING

The CEU instruction is used to command the disc control unit. There are two CEU formats for disc commands: disc seek and disc data. The disc seek command is used to position the head assembly to the required track. Refer to Appendix C for CEU second word formats for the disc control unit.

The disc control unit accepts a total of five commands from the computer. These commands define the total range of disc operations and are as follows:

a. Seek Track Zero	Disc Seek Mode
b. Seek n Tracks Forward	
c. Seek n Tracks Reverse	
d. Write Section I, Head J	Disc Data Mode
e. Read Section I, Head J	

The three seek commands enable the 10 physically-connected heads to be positioned to any desired track number. The program may keep track of the current head assembly position and command the head assembly to be moved a specified number (n) of tracks in either direction to position the heads to a new track number. The positioning mechanism is extremely reliable, but an absolute verification of the new head position can be obtained by recording track and sector identification in one or all sectors per track, and subsequently reading a sector containing this identification each time the heads are repositioned. An alternate method of track addressing consists of sending the heads to track 0 after each disc transfer is completed. Use of this technique allows absolute (rather than relative) track addressing, but increases the minimum time between successive disc operations.

The read and write commands enable any sector on the 10 tracks currently under the heads to be read or written. To seek track zero (when the current track is unknown) the following routine is executed:

Skip Mode	Wait Mode
CEU '13	CEU '13, W
DATA '10	DATA '10
BRU *-2	

Once the head is positioned at any track, motion commands specify the number of tracks to be moved and the direction of movement (forward or reverse). For example, assume that the head assembly is at track 50 and the new positions are to be, successively, tracks 55, 71, 38, and 43. To move the heads to the required tracks, the following instructions are executed:

```
CEU     '13, W
DATA    '132        Forward, 5 Tracks
  .
  .
  .
CEU     '13, W
DATA    '200012     Forward, 16 Tracks
  .
  .
  .
  .
CEU     '13, W
DATA    '400031     Reverse, 33 Tracks
  .
  .
  .
CEU     '13, W
DATA    '132        Forward, 5 Tracks
```

Note that, in a disc seek command, bit 20 of the CEU second word is always a ONE.

Head and sector selection are performed by executing the CEU with the disc data format. For example, to read sector 12, head 5, the CEU instruction would be:

```
CEU     '13, W
DATA    '3000121
```

To write sector 7, head 7, the CEU instruction would be:

```
CEU     '13,W
DATA    '1600162
```

The two standard interrupts may be connected/disconnected by the execution of the CEU instruction with the appropriate combination of bits 1, 2, and 3 of the second word. The seek error interrupt occurs when a motion command occurs that cannot be performed; that is, the heads are at track 70 and a forward, 70 tracks command is given. The seek complete interrupt occurs when the heads are at the selected track.

The TEU instruction can be used to test for seek complete, seek error, disc pack on-line, read overflow, write overflow, checksum error, DCU ready, and unit busy. See Appendix C for the TEU second word format.

Data is transferred between the disc and the computer I/O structure one word at a time, in blocks not exceeding 64 words (1 sector).

When using the BTC unit with the disc storage unit, the BTC is initialized with bit 0 of the disc data CEU format. If more than one sector of data is to be read or written, the terminate bit should be set in the word count (WC) location to allow the interrupt processing routine to handle the housekeeping functions (sector and head modification, first word address of buffers, etc.).

When transferring single words to/from the disc (non-BTC applications), the data must be presented to the disc each 19.4 microseconds, or faster. Otherwise, data will be dropped during the transfer.

FIXED HEAD DISC STORAGE—MODEL NO. 84-654A (DEVICE NO. 13)

The fixed head disc storage unit provides random-access bulk storage of output data from any SEL Series 800 Computer. When used in SEL 840 Multiprocessor Computer Systems, the storage capacity is 606,208 24-bit words (maximum). There are up to eight recording surfaces with 64 fixed recording heads per surface. Each surface contains 64 recording tracks with each track divided into 16 sectors. Each sector provides storage for seventy-four, 24-bit data words. Figures 6-13 and 6-14 illustrate the divisions of a recording surface and the positions of the heads relative to each surface.

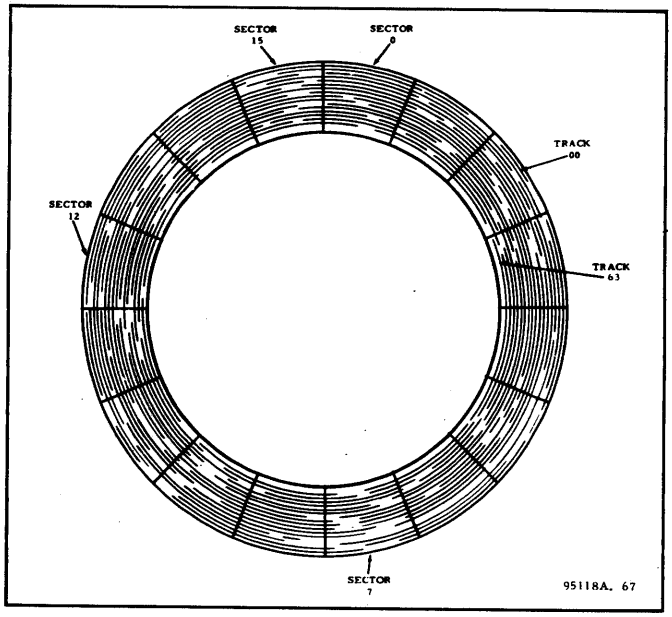


Figure 6-13. Fixed Head Track and Sector Layout

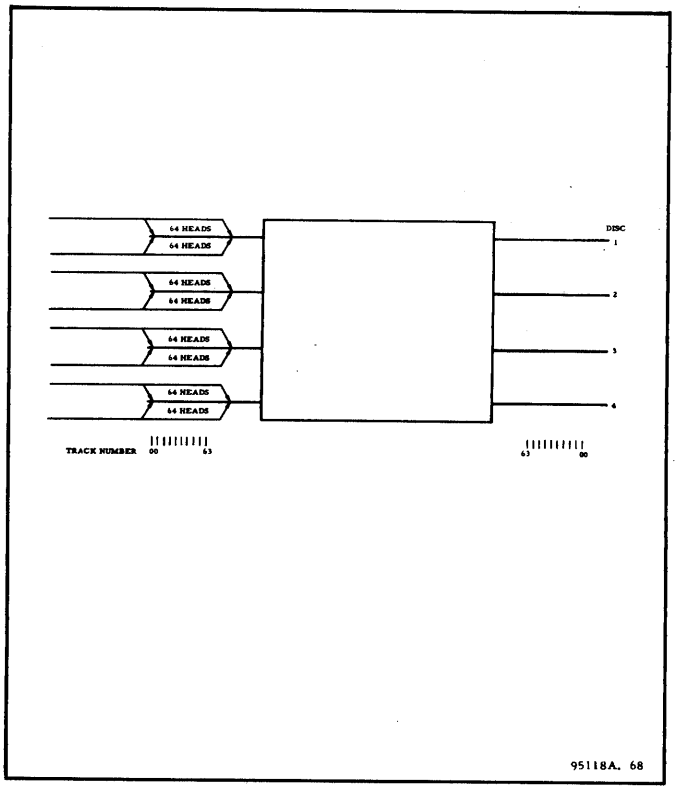


Figure 6-14. Fixed Head Arrangement - Recording Surface

Average access time for data recording or retrieval is 8.3 milliseconds. Maximum access time is 17 milliseconds. Word transfer rate to or from the storage unit is 75 kHz.

The disc control unit regulates the data flow between the processor and the disc storage unit. The disc control unit also performs, under program control, all track and sector selections required to store and retrieve data from specified disc locations. A checksum is generated and written on the disc at the end of each recorded sector. When the data is read, another checksum is generated and compared to the previously-written checksum at the end of the sector. The fixed head disc storage specifications are listed in table 6-15.

OPERATION AND PROGRAMMING

The CEU instruction (or SM-I/OP command operation) is used to set up the fixed head disc storage system. Three separate CEU or SM-I/OP command formats are used when programming the fixed head disc system These formats are:

a. Select track

b. Read

c. Write

The select track format allows any of the 512 (0-511) tracks to be selected for a subsequent read or write command. Once the desired track has been selected, the starting sector is selected by executing a CEU instruction in the read or write format. Both the read and write command formats contain a provision for sequential operation. A read sequential command will cause the data to be read from the selected track, starting at the specified sector and continuing through the remaining sectors and tracks until all the desired data is read. (The actual data transfer will occur via the BTC on a series of AIP or MIP instructions.) A write sequential command will cause the data to be written on the disc on the selected track, starting from the specified sector and continuing sequentially through the remaining sectors and tracks until all the desired data has been transferred to the disc. (The actual data transfer is handled via the BTC or a series of AOP or MOP instructions.)

NOTE

When writing on the disc, the programmer must insure that the available recording surface can accommodate the total data block. Otherwise, a write sequential operation could cause previously recorded information to be destroyed. A hardware means for protecting specific areas on the disc is provided. Refer to the fixed head disc system technical manual.

If the disc storage system is not commanded to read/write sequentially, it will write only on the selected track in the specified sector.

In addition to specifying the specific disc function, each command word format contains provisions for:

a. Initializing the BTC.

b. Connecting/disconnecting the two interrupts to the standard processor interrupt levels.

c. Transferring the BTC current word address to the processor.

Table 6-15. Fixed Head Disc Storage System Specifications

Characteristics	Specifications
Speed	3600 rpm; 16.7 ms/rev. ±S*
Capacity (max.)	
Bits	14,548,992
Words (24 bits)	606,208
Word Rate	75.0 kHz
Bit Rate	1.8 MHz
Access Time	
Average	8.3 milliseconds
Maximum	16.7 milliseconds +S*
Time between Sectors	37.2 microseconds +S*
Number of Tracks	
Per Surface	64
Per Disc	128
Number of Discs	1 to 4
Recording Density	1,000 bits per inch
Interrupts (2)	1. Program error or checksum error 2. Read and write overflow
Error Detection	Checksum
Disc Coating	Nickel-Cobalt magnetic plating

*S = Induction motor slippage; approximately 3 to 4 percent

The fixed head disc system may also be tested, under program control, by the TEU instruction or an I/OP test operation. Seven separate tests may be performed:

a. Skip on no program error

b. Skip on disc on-line

c. Skip on disc read overflow

d. Skip on disc write overflow

e. Skip on no parity error

f. Skip on no disc file area protected

g. Skip on disc controller not busy

These tests operate in the normal TEU or I/OP test manner; that is, if the test condition is satisfied, the next sequential instruction is skipped. If the test condition is not satisfied, the next instruction is executed.

SECTION VII
OPTIONS

GENERAL

A wide variety of optional features is available with 840 Multiprocessor Computer Systems. These options are summarized in Section I under SYSTEM CHARACTERISTICS. These options are discussed elsewhere in this manual; that is, a description of block transfer control (BTC) operation is contained in Section IV - INPUT/OUTPUT. Other options, such as the extended arithmetic unit (EAU), are described in the discussion of the 840MP Computer. This section of the manual describes those options which have not been previously discussed.

PROGRAM PROTECT AND INSTRUCTION TRAP (84-080MP)

The program protect feature permits individual words and areas of memory (data or programs) to be guarded against accidental modification. This protection is accomplished by the use of both a special bit stored with each word in memory and 840MP Computer logic. A special key switch on the 840MP Computer control console enables the program protect feature. When enabled, the protect feature operates as follows:

a. Any instruction having its program protect bit ON (ONE) can operate on any protected or unprotected memory location.

b. Any instruction having its protect bit OFF (ZERO) cannot modify the contents of, or branch into, a protected memory location. If an unprotected (protect bit off) instruction attempts to modify the contents of, or branch into, a protected location, a priority interrupt will occur. The interrupt routine will change the mode to protected, and then determines what to do with the attempted violation. The priority interrupt will also occur if the program counter is advanced from an unprotected instruction address to a protected instruction address.

OPERATION AND PROGRAMMING

The program protect mode is maintained by a protect latch. If the PROGRAM PROTECT MODE key switch on the 840MP Computer console is off, the protect latch is inoperative. The protect latch operates as follows:

a. The protect latch is set on when either the key switch is turned from disable to enable (protect mode on), or a priority interrupt occurs.

b. When the protect latch is on, any operand or instruction can be executed.

c. If the instruction is not protected, the protect latch will turn off.

d. If the protect latch is off, any attempt to execute an instruction with the program protect bit (PPB) on will cause an interrupt. If the protect latch is off, any executed instruction that would change the contents of a memory location which has the program protect bit on will cause an interrupt.

e. Any priority interrupt will turn on the protect latch. Any instruction within the interrupt subroutine which is not protected will turn off the protect latch. If all the instructions are protected, the protect latch will stay on.

The status of the protect latch and the program protect bit are displayed on the console with separate indicators.

The protect bit on (PON) instruction is used to set the protect bit in the specified memory location The protect bit off (POF) instruction is used to turn off the protect bit. Indirect addressing and indexing can be performed with the PON and POF instructions.

When the 840MP Computer is in the protected mode, the PON and POF instructions must have their protect bits on in order to operate. Otherwise, the central processor will perform a NOP (no operation) instruction. The instruction trap portion of the program protect option causes unprotected PON and POF instructions to be trapped and a priority interrupt to be generated.

When the key switch on the central processor console is off, the PON and POF instructions operate normally without regard to their protected or unprotected status. Instruction trap is described in the following paragraphs.

Figure 7-1 illustrates one method of protecting a number of memory locations. In the example, locations 0000 through 0066_8 are to have their protect bit turned on.

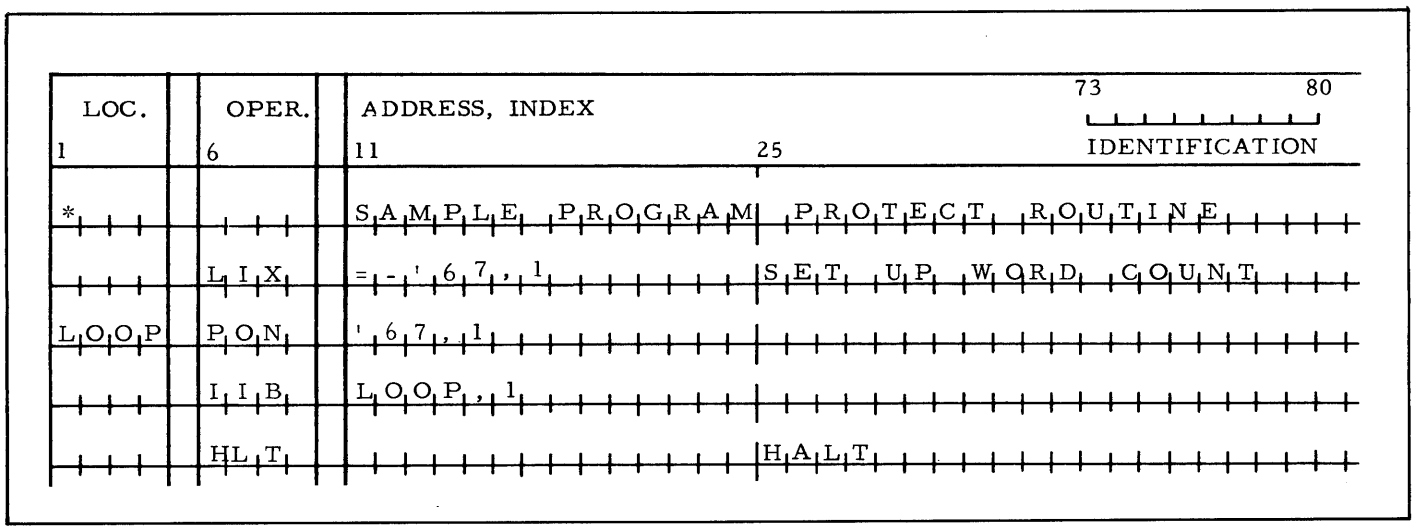


Figure 7-1. Program Protect Sample Program

The instruction trap (include as part of the program protect option) is a hardware provision for preventing the execution of certain instructions. In 840 Multiprocessor Computer Systems, the following instructions are trapped when the instruction trap is enabled:

PON	MOP
POF	MIP
PIE	LBR
PID	TAC
CEU	TEU
AOP	HLT
AIP	PIR

When an attempt to execute one of the above instructions is made, a priority interrupt is generated (providing that interrupt has been previously enabled). The instruction causing the interrupt is then interrogated within the interrupt routine.

The instruction trap is enabled and disabled by PIE and PID instructions, respectively.

STALL ALARM (84-042MP)

The stall alarm is designed to enable a recovery routine to be entered automatically any time a computer "stall" condition occurs. If, after 32 machine cycles (1.75 microseconds per cycle), the computer program address register has not advance, an override interrupt is generated. This interrupt is capable of interrupting an indirect chain, an I/O instruction or even a halt condition. The interrupt routine assigned to this interrupt would interrogate the source of the interrupt to determine its nature.

AUTO START (84-041MP)

The auto start feature provides the capability of the 840MP Computer to return to a run condition automatically in the event that, after being lost, power is restored. An override interrupt is provided which allows return to the regular program or special routines The auto start option is normally used in conjunction with the power fail safe provision.

20 SENSE SWITCHES (84-021MP)

This option includes the wiring and program addressing logic to enable all 24 data-entry program halt sense switches, on the control panel to be used as sense switches.

REAL TIME CLOCK (84-031MP)

The 60 Hz Real Time Clock option generates interrupt signals at the frequency of the ac power.

INPUT/OUTPUT PARITY (84-210MP)

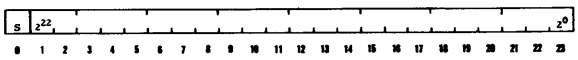
This option is made available for use with special computer interfaces such as serial communication links in which a parity bit is transmitted with each word. This option consists of a means to transfer the parity bit stored in memory with each computer word output transfer, and to check the parity bit accompanying each computer input word transfer

The operation of the I/O Parity unit is controlled by the external interface unit which must send a request for parity checking/transfer with each input/output word request If a parity error is detected for an input transfer, the error signal is transferred to the external unit.

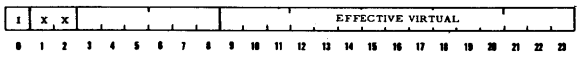
APPENDIX A

SEL 840MP COMPUTER WORD FORMAT

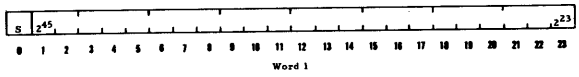
Integer Data

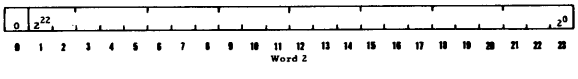


Indirect Address Word

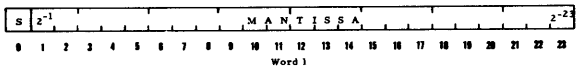


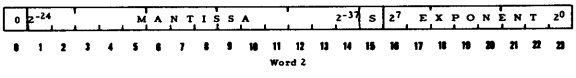
Double-Precision Fixed Point Data



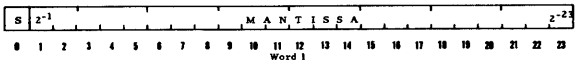


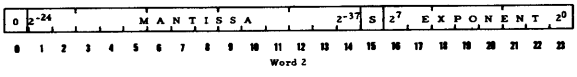
Single-Precision Floating Point Data

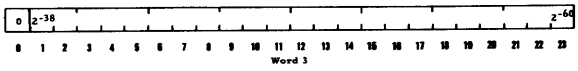




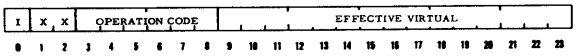
Double-Precision Floating Point Data



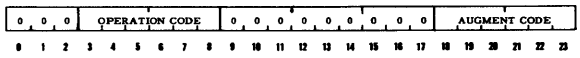




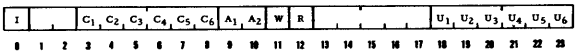
Memory Reference Instruction



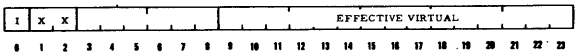
Augmented Instruction



Input/Output Instruction

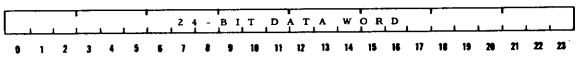


First Word



Second Word
(Address Mode)

OR



OR
Second Word
(Immediate Mode)

95098A. A1

APPENDIX B

SEL PERIPHERAL DEVICE OCTAL CHARACTER CODES

ALPHABETIC CHARACTERS

Character	Teletype ASR-33 & ASR-35	Line Printer (Truncated ASCII)	IBM/BCD	Hollerith Card Code	
				Octal Code	Card Rows
A	301	01	61	4400	12-1
B	302	02	62	4200	12-2
C	303	03	63	4100	12-3
D	304	04	64	4040	12-4
E	305	05	65	4020	12-5
F	306	06	66	4010	12-6
G	307	07	67	4004	12-7
H	310	10	70	4002	12-8
I	311	11	71	4001	12-9
J	312	12	41	2400	11-1
K	313	13	42	2200	11-2
L	314	14	43	2100	11-3
M	315	15	44	2040	11-4
N	316	16	45	2020	11-5
O	317	17	46	2010	11-6
P	320	20	47	2004	11-7
Q	321	21	50	2002	11-8
R	322	22	51	2001	11-9
S	323	23	22	1200	0-2
T	324	24	23	1100	0-3
U	325	25	24	1040	0-4
V	326	26	25	1020	0-5
W	327	27	26	1010	0-6
X	330	30	27	1004	0-7
Y	331	31	30	1002	0-8
Z	332	32	31	1001	0-9

APPENDIX B (CONT'D)

SEL PERIPHERAL DEVICE OCTAL CHARACTER CODES

NUMERIC CHARACTERS

Character	Teletype ASR-33 & ASR-35	Line Printer (Truncated ASCII)	IBM/BCD	Hollerith Card Code	
				Octal Code	Card Rows
0	260	60	12	1000	0
1	261	61	01	0400	1
2	262	62	02	0200	2
3	263	63	03	0100	3
4	264	64	04	0040	4
5	265	65	05	0020	5
6	266	66	06	0010	6
7	267	67	07	0004	7
8	270	70	10	0002	8
9	271	71	11	0001	9

SPECIAL SYMBOLS OR FUNCTIONS

Symbol Or Function	Teletype ASR-33 & ASR-35	Line Printer (Truncated ASCII)	IBM/BCD	Hollerith Card Code	
				Octal Code	Card Rows
@	300	00	57	2006	11-8-7
[	333	33	75	4022	12-8-5
\	334	34	36	1012	0-8-6
]	335	35	55	2024	11-8-5
↑	336	36	32	1202	0-8-2
←	337	37	77	4006	12-8-7
Space	240	40	20	—	—
!	241	41	52	3000	11-0
"	242	42	37	1006	0-8-7
#	243	43	35	1022	0-8-5
$	244	44	53	2102	11-8-3
%	245	45	—	—	—
&	246	46	—	—	—

SEL PERIPHERAL DEVICE OCTAL CHARACTER CODES

SPECIAL SYMBOLS OR FUNCTIONS

Symbol Or Function	Teletype ASR-33 & ASR-35	Line Printer (Truncated ASCII)	IBM/BCD	Hollerith Card Code	
				Octal Code	Card Rows
'	247	47	14	0042	8-4
(	250	50	34	1042	0-8-4
)	251	51	74	4042	12-8-4
*	252	52	54	2042	11-8-4
+	253	53	60	4000	12
,	254	54	33	1102	0-8-3
_	255	55	40	2000	11
.	256	56	73	4102	12-8-3
/	257	57	21	1400	0-1
:	272	72	15	0022	8-5
;	273	73	56	2012	11-8-6
<	274	74	76	4012	12-8-6
=	275	75	13	0102	8-3
>	276	76	16	0012	8-6
?	277	77	72	5000	12-0
Carriage Return	215				
Line Feed	212				
Bell	207				
Delete	377				

95118A. B3

APPENDIX C
840MP COMPUTER CEU WORD FORMAT

CEU SECOND WORD FORMAT

	0	1	2*	3*	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
MAGNETIC TAPE FORMAT 0	0	PRIORITY INTERRUPT CONNECT = 1 DISCONNECT = 0	WORD TRANSFER READY INTERRUPT	END OF RECORD INTERRUPT	0	REWIND	ERASE FOUR INCHES OF TAPE	BCD = 1 BINARY = 0									DENSITY**			TAPE TRANSPORT		CURRENT WORD ADDRESS IN	CHARACTERS** PER WORD	
MAGNETIC TAPE FORMAT 1	BTC INITIALIZE		WORD TRANSFER READY INTERRUPT	END OF RECORD INTERRUPT	1	WRITE RECORD	WRITE END OF FILE	READ RECORD									ADVANCE RECORD	ADVANCE END OF FILE	BACKSPACE RECORD	BACKSPACE END OF FILE	CORRECT CRC ERROR (9-TRACK OPTION)			
ASR-33/35	0		IN	OUT	READER MODE	KEYBOARD MODE	MODE CLEAR															0		
PAPER TAPE READER AND PUNCH	0		IN	OUT	PUNCH POWER ON	PUNCH POWER OFF	READER ENABLE	READER DISABLE														0		
CARD READER PUNCH	BTC INITIALIZE		IN	OUT	READER FEED CARD & READ BINARY	READER FEED CARD & READ BCD (BURROUGHS ONLY)	READER STACKER OFFSET	FEED CARD (PUNCH)									EJECT CARD (PUNCH)	PUNCH STACKER OFFSET				CURRENT WORD ADDRESS IN		
X-Y PLOTTER	0		PROCESS COMPLETE		PEN DOWN	PEN UP	DRUM DOWN	DRUM UP									CARRIAGE LEFT	CARRIAGE RIGHT				0		
LINE PRINTER	BTC INITIALIZE		END OF PRINT	BUFFER NOT BUSY	ADVANCE TO FORMAT N ***	ADVANCE 1 LINE	TOP OF FORM	PRINT —OR— ***									CLEAR BUFFER —OR— ***	FILL BUFFER —OR— ***				CURRENT WORD ADDRESS IN		
MOVEABLE HEAD DISC SEEK			SEEK ERROR	SEEK COMPLETE		NO. OF TRACKS TO BE MOVED											NO. OF TRACKS TO BE MOVED				1		FORWARD ****	REVERSE ****
						64	32	16									8	4	2	1				
MOVEABLE HEAD DISC DATA			SEEK ERROR	SEEK COMPLETE		SECTOR NUMBER											HEAD NUMBER				0		WRITE	READ
					8	4	2	1									8	4	2	1				
FIXED HEAD DISC SELECT TRACK			CHECKSUM ERROR OR PROGRAM ERROR	READ OVERFLOW OR PROGRAM ERROR		TRACK NUMBER											TRACK NUMBER						1	1
					256	128	64	32									16	8	4	2	1			
FIXED HEAD DISC READ			CHECKSUM ERROR OR PROGRAM ERROR	READ OVERFLOW OR WRITE OVERFLOW	READ SEQUENTIAL													STARTING SECTOR					0	READ 1
																		8	4	2	1			
FIXED HEAD DISC WRITE			CHECKSUM ERROR OR PROGRAM ERROR	READ OVERFLOW OR WRITE OVERFLOW	WRITE SEQUENTIAL													STARTING SECTOR					WRITE 1	0
																		8	4	2	1			
CRT			OVERFLOW	STOP	DISPLAY ON	DISPLAY OFF																		

*INTERUPT LEVELS:

BIT 2 — LEVEL 14, GROUP 0
BIT 3 — LEVEL 15, GROUP 0

**MAGNETIC TAPE DENSITY & CHARACTERS PER WORD:

BITS 16 & 17	BITS 22 & 23
00 = 200 BPI	01 = 1 CHAR/WORD
01 = 556 BPI	10 = 2 CHAR/WORD
10 = 800 BPI	11 = 3 CHAR/WORD

*** WHEN A ONE IS PRESENT IN BIT 4, ADVANCE TO THE CHANNEL NUMBER (EXPRESSED IN OCTAL) REPRESENTED BY THE CODE IN BIT POSITIONS 7, 16 AND 17.

**** TO SEEK TRACK 00, BOTH BITS (22 & 23) MUST BE ZERO.

APPENDIX C

840MP COMPUTER TEU WORD FORMAT

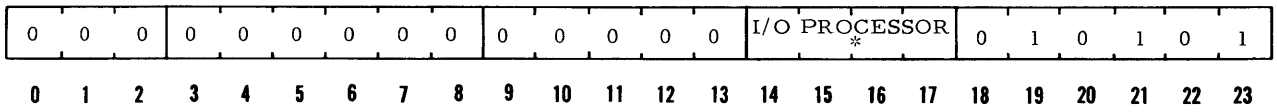
TEU SECOND WORD FORMAT

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
MAGNETIC TAPE	SKIP IF NOT BUSY	SKIP IF NO END OF FILE	SKIP IF NO OVERFLOW	SKIP IF AT LOAD POINT	SKIP ON END OF RECORD INTERRUPT	SKIP IF NO PARITY ERROR	SKIP IF WRITE RING IN	SKIP IF NO END OF TAPE									SKIP IF REWINDING	SKIP IF NO DATA ERROR (9-TRACK ONLY)						
LINE PRINTER					SKIP IF NOT BUSY		SKIP IF NO PARITY ERROR										SKIP IF NO BOTTOM OF FORM	SKIP IF PRINTER OPERABLE						
CARD READER/ PUNCH																		SKIP IF NO I/O PUNCH ERROR						
MOVEABLE HEAD DISC					SKIP IF SEEK COMPLETE	SKIP IF NO SEEK ERROR	SKIP IF BEGINNING OF DISC	SKIP IF BEGINNING OF SECTOR									SKIP IF PACK ON LINE	SKIP IF NO READ OVERFLOW	SKIP IF NO WRITE OVERFLOW	SKIP IF NO CHECKSUM ERROR	SKIP IF FILE UNSAFE	SKIP IF DCU READY	SKIP IF NOT BUSY	
FIXED HEAD DISC	SKIP IF NO PROGRAM ERROR	SKIP IF DISC ON LINE	SKIP IF NO DISC READ OVERFLOW	SKIP IF NO DISC WRITE OVERFLOW	SKIP IF NO CHECKSUM ERROR	SKIP IF NO DISC FILE AREA PROTECTED	SKIP IF CONTROLLER NOT BUSY																	
INTERVAL TIMER																	SKIP ON ZERO COUNT INTERRUPT DISABLE							

APPENDIX D

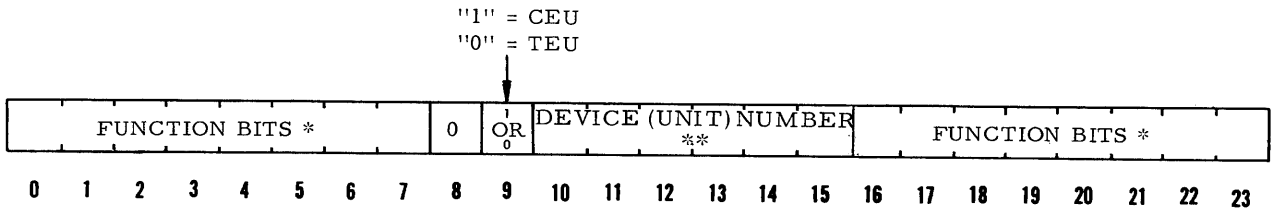
840MP COMPUTER TAC INSTRUCTION WORD FORMAT

TAC INSTRUCTION WORD FORMAT



* UNITARY CODE FOR SM-I/OP NUMBER

SHARED MEMORY I/O PROCESSOR CEU/TEU INSTRUCTION WORD
(DEDICATED LOCATION)



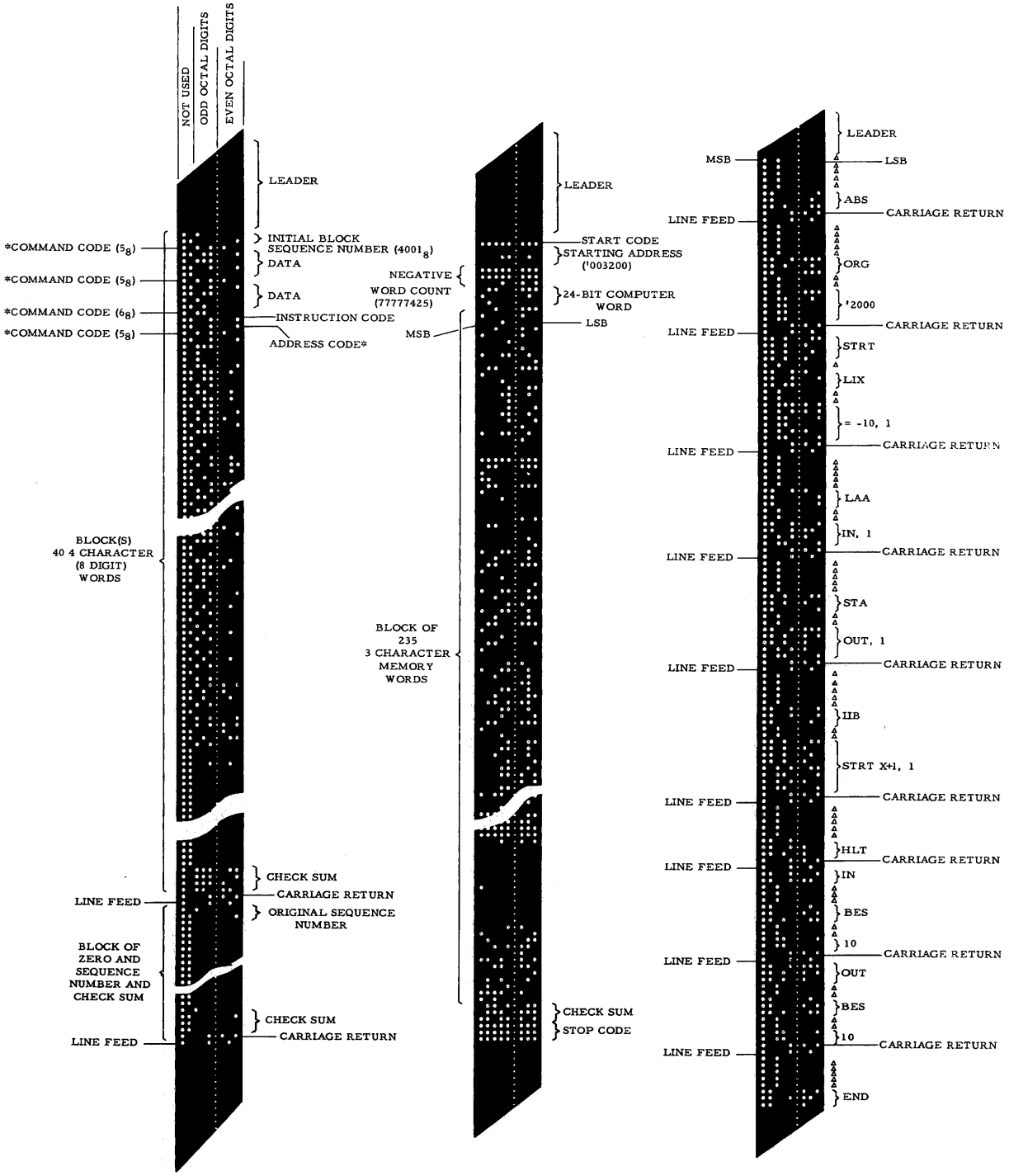
*FUNCTION BITS ARE IDENTICAL TO THOSE SHOWN IN CEU AND TEU WORD FORMATS IN APPENDIX C

**BINARY, CODING FOR DEVICE NUMBER

95098A. D1

APPENDIX E

SEL 840 PAPER TAPE FORMATS



SEL 840 ASSEMBLER AND COMPILER OBJECT PROGRAM OUTPUT TAPE. MUST BE LOADED WITH THE SEL MNEMBLER LOADER PROGRAM. *SEE NEXT PAGE FOR WORD FORMAT DESCRIPTIONS

SEL 840 DEBUG DUMP TAPE. MUST BE LOADED WITH DEBUG LOADER PROGRAM.

ASC II CODE ASSEMBLER SOURCE INPUT IN CARD FORMAT IMAGE. MUST BE LOADED BY ASSEMBLER.

95098A.E1

APPENDIX F

NUMERICAL INFORMATION

NUMERICAL OCTAL TO DECIMAL CONVERSION

OCTAL MULTIPLICATION

M•N or N•M

M \ N	1	2	3	4	5	6	7	10
1	1	2	3	4	5	6	7	10
2	2	4	6	10	12	14	16	20
3	3	6	11	14	17	22	25	30
4	4	10	14	20	24	30	34	40
5	5	12	17	24	31	36	43	50
6	6	14	22	30	36	44	52	60
7	7	16	25	34	43	52	61	70
10	10	20	30	40	50	60	70	100

OCTAL ADDITION

M+N or N+M

M \ N	1	2	3	4	5	6	7	10
1	2	3	4	5	6	7	10	11
2	3	4	5	6	7	10	11	12
3	4	5	6	7	10	11	12	13
4	5	6	7	10	11	12	13	14
5	6	7	10	11	12	13	14	15
6	7	10	11	12	13	14	15	16
7	10	11	12	13	14	15	16	17
10	11	12	13	14	15	16	17	20

$8^0 = 1$
$8^1 = 8$
$8^2 = 64$
$8^3 = 512$
$8^4 = 4096$
$8^5 = 32768$

95118A.F1

APPENDIX F (CONT'D)

NUMERICAL INFORMATION

TABLE OF POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 627 776	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5
2 199 023 255 552	41	0.000 000 000 000 454 747 350 886 464 118 957 519 531 25
4 398 046 511 104	42	0.000 000 000 000 227 373 675 443 232 059 478 759 765 625
8 796 093 022 208	43	0.000 000 000 000 113 686 837 721 616 029 739 379 882 812 5
17 592 186 044 416	44	0.000 000 000 000 056 843 418 860 808 014 869 689 941 406 25
35 184 372 088 832	45	0.000 000 000 000 028 421 709 430 404 007 434 844 970 703 125
70 368 744 177 664	46	0.000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5
140 737 488 355 328	47	0.000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25
281 474 976 710 656	48	0.000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625

95118A. F2

APPENDIX F (CONT'D)

NUMERICAL INFORMATION

OCTAL-DECIMAL INTEGER CONVERSION TABLE

0000 to 0777 (Octal)	0000 to 0511 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0004	0005	0006	0007
0010	0008	0009	0010	0011	0012	0013	0014	0015
0020	0016	0017	0018	0019	0020	0021	0022	0023
0030	0024	0025	0026	0027	0028	0029	0030	0031
0040	0032	0033	0034	0035	0036	0037	0038	0039
0050	0040	0041	0042	0043	0044	0045	0046	0047
0060	0048	0049	0050	0051	0052	0053	0054	0055
0070	0056	0057	0058	0059	0060	0061	0062	0063
0100	0064	0065	0066	0067	0068	0069	0070	0071
0110	0072	0073	0074	0075	0076	0077	0078	0079
0120	0080	0081	0082	0083	0084	0085	0086	0087
0130	0088	0089	0090	0091	0092	0093	0094	0095
0140	0096	0097	0098	0099	0100	0101	0102	0103
0150	0104	0105	0106	0107	0108	0109	0110	0111
0160	0112	0113	0114	0115	0116	0117	0118	0119
0170	0120	0121	0122	0123	0124	0125	0126	0127
0200	0128	0129	0130	0131	0132	0133	0134	0135
0210	0136	0137	0138	0139	0140	0141	0142	0143
0220	0144	0145	0146	0147	0148	0149	0150	0151
0230	0152	0153	0154	0155	0156	0157	0158	0159
0240	0160	0161	0162	0163	0164	0165	0166	0167
0250	0168	0169	0170	0171	0172	0173	0174	0175
0260	0176	0177	0178	0179	0180	0181	0182	0183
0270	0184	0185	0186	0187	0188	0189	0190	0191
0300	0192	0193	0194	0195	0196	0197	0198	0199
0310	0200	0201	0202	0203	0204	0205	0206	0207
0320	0208	0209	0210	0211	0212	0213	0214	0215
0330	0216	0217	0218	0219	0220	0221	0222	0223
0340	0224	0225	0226	0227	0228	0229	0230	0231
0350	0232	0233	0234	0235	0236	0237	0238	0239
0360	0240	0241	0242	0243	0244	0245	0246	0247
0370	0248	0249	0250	0251	0252	0253	0254	0255

	0	1	2	3	4	5	6	7
0400	0256	0257	0258	0259	0260	0261	0262	0263
0410	0264	0265	0266	0267	0268	0269	0270	0271
0420	0272	0273	0274	0275	0276	0277	0278	0279
0430	0280	0281	0282	0283	0284	0285	0286	0287
0440	0288	0289	0290	0291	0292	0293	0294	0295
0450	0296	0297	0298	0299	0300	0301	0302	0303
0460	0304	0305	0306	0307	0308	0309	0310	0311
0470	0312	0313	0314	0315	0316	0317	0318	0319
0500	0320	0321	0322	0323	0324	0325	0326	0327
0510	0328	0329	0330	0331	0332	0333	0334	0335
0520	0336	0337	0338	0339	0340	0341	0342	0343
0530	0344	0345	0346	0347	0348	0349	0350	0351
0540	0352	0353	0354	0355	0356	0357	0358	0359
0550	0360	0361	0362	0363	0364	0365	0366	0367
0560	0368	0369	0370	0371	0372	0373	0374	0375
0570	0376	0377	0378	0379	0380	0381	0382	0383
0600	0384	0385	0386	0387	0388	0389	0390	0391
0610	0392	0393	0394	0395	0396	0397	0398	0399
0620	0400	0401	0402	0403	0404	0405	0406	0407
0630	0408	0409	0410	0411	0412	0413	0414	0415
0640	0416	0417	0418	0419	0420	0421	0422	0423
0650	0424	0425	0426	0427	0428	0429	0430	0431
0660	0432	0433	0434	0435	0436	0437	0438	0439
0670	0440	0441	0442	0443	0444	0445	0446	0447
0700	0448	0449	0450	0451	0452	0453	0454	0455
0710	0456	0457	0458	0459	0460	0461	0462	0463
0720	0464	0465	0466	0467	0468	0469	0470	0471
0730	0472	0473	0474	0475	0476	0477	0478	0479
0740	0480	0481	0482	0483	0484	0485	0486	0487
0750	0488	0489	0490	0491	0492	0493	0494	0495
0760	0496	0497	0498	0499	0500	0501	0502	0503
0770	0504	0505	0506	0507	0508	0509	0510	0511

1000 to 1777 (Octal)	0512 to 1023 (Decimal)

	0	1	2	3	4	5	6	7
1000	0512	0513	0514	0515	0516	0517	0518	0519
1010	0520	0521	0522	0523	0524	0525	0526	0527
1020	0528	0529	0530	0531	0532	0533	0534	0535
1030	0536	0537	0538	0539	0540	0541	0542	0543
1040	0544	0545	0546	0547	0548	0549	0550	0551
1050	0552	0553	0554	0555	0556	0557	0558	0559
1060	0560	0561	0562	0563	0564	0565	0566	0567
1070	0568	0569	0570	0571	0572	0573	0574	0575
1100	0576	0577	0578	0579	0580	0581	0582	0583
1110	0584	0585	0586	0587	0588	0589	0590	0591
1120	0592	0593	0594	0595	0596	0597	0598	0599
1130	0600	0601	0602	0603	0604	0605	0606	0607
1140	0608	0609	0610	0611	0612	0613	0614	0615
1150	0616	0617	0618	0619	0620	0621	0622	0623
1160	0624	0625	0626	0627	0628	0629	0630	0631
1170	0632	0633	0634	0635	0636	0637	0638	0639
1200	0640	0641	0642	0643	0644	0645	0646	0647
1210	0648	0649	0650	0651	0652	0653	0654	0655
1220	0656	0657	0658	0659	0660	0661	0662	0663
1230	0664	0665	0666	0667	0668	0669	0670	0671
1240	0672	0673	0674	0675	0676	0677	0678	0679
1250	0680	0681	0682	0683	0684	0685	0686	0687
1260	0688	0689	0690	0691	0692	0693	0694	0695
1270	0696	0697	0698	0699	0700	0701	0702	0703
1300	0704	0705	0706	0707	0708	0709	0710	0711
1310	0712	0713	0714	0715	0716	0717	0718	0719
1320	0720	0721	0722	0723	0724	0725	0726	0727
1330	0728	0729	0730	0731	0732	0733	0734	0735
1340	0736	0737	0738	0739	0740	0741	0742	0743
1350	0744	0745	0746	0747	0748	0749	0750	0751
1360	0752	0753	0754	0755	0756	0757	0758	0759
1370	0760	0761	0762	0763	0764	0765	0766	0767

	0	1	2	3	4	5	6	7
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865	0866	0867	0868	0869	0870	0871
1550	0872	0873	0874	0875	0876	0877	0878	0879
1560	0880	0881	0882	0883	0884	0885	0886	0887
1570	0888	0889	0890	0891	0892	0893	0894	0895
1600	0896	0897	0898	0899	0900	0901	0902	0903
1610	0904	0905	0906	0907	0908	0909	0910	0911
1620	0912	0913	0914	0915	0916	0917	0918	0919
1630	0920	0921	0922	0923	0924	0925	0926	0927
1640	0928	0929	0930	0931	0932	0933	0934	0935
1650	0936	0937	0938	0939	0940	0941	0942	0943
1660	0944	0945	0946	0947	0948	0949	0950	0951
1670	0952	0953	0954	0955	0956	0957	0958	0959
1700	0960	0961	0962	0963	0964	0965	0966	0967
1710	0968	0969	0970	0971	0972	0973	0974	0975
1720	0976	0977	0978	0979	0980	0981	0982	0983
1730	0984	0985	0986	0987	0988	0989	0990	0991
1740	0992	0993	0994	0995	0996	0997	0998	0999
1750	1000	1001	1002	1003	1004	1005	1006	1007
1760	1008	1009	1010	1011	1012	1013	1014	1015
1770	1016	1017	1018	1019	1020	1021	1022	1023

95118A. F3

APPENDIX F (CONT'D)

NUMERICAL INFORMATION

OCTAL-DECIMAL INTEGER CONVERSION TABLE

	0	1	2	3	4	5	6	7
2000	1024	1025	1026	1027	1028	1029	1030	1031
2010	1032	1033	1034	1035	1036	1037	1038	1039
2020	1040	1041	1042	1043	1044	1045	1046	1047
2030	1048	1049	1050	1051	1052	1053	1054	1055
2040	1056	1057	1058	1059	1060	1061	1062	1063
2050	1064	1065	1066	1067	1068	1069	1070	1071
2060	1072	1073	1074	1075	1076	1077	1078	1079
2070	1080	1081	1082	1083	1084	1085	1086	1087
2100	1088	1089	1090	1091	1092	1093	1094	1095
2110	1096	1097	1098	1099	1100	1101	1102	1103
2120	1104	1105	1106	1107	1108	1109	1110	1111
2130	1112	1113	1114	1115	1116	1117	1118	1119
2140	1120	1121	1122	1123	1124	1125	1126	1127
2150	1128	1129	1130	1131	1132	1133	1134	1135
2160	1136	1137	1138	1139	1140	1141	1142	1143
2170	1144	1145	1146	1147	1148	1149	1150	1151
2200	1152	1153	1154	1155	1156	1157	1158	1159
2210	1160	1161	1162	1163	1164	1165	1166	1167
2220	1168	1169	1170	1171	1172	1173	1174	1175
2230	1176	1177	1178	1179	1180	1181	1182	1183
2240	1184	1185	1186	1187	1188	1189	1190	1191
2250	1192	1193	1194	1195	1196	1197	1198	1199
2260	1200	1201	1202	1203	1204	1205	1206	1207
2270	1208	1209	1210	1211	1212	1213	1214	1215
2300	1216	1217	1218	1219	1220	1221	1222	1223
2310	1224	1225	1226	1227	1228	1229	1230	1231
2320	1232	1233	1234	1235	1236	1237	1238	1239
2330	1240	1241	1242	1243	1244	1245	1246	1247
2340	1248	1249	1250	1251	1252	1253	1254	1255
2350	1256	1257	1258	1259	1260	1261	1262	1263
2360	1264	1265	1266	1267	1268	1269	1270	1271
2370	1272	1273	1274	1275	1276	1277	1278	1279

	0	1	2	3	4	5	6	7
2400	1280	1281	1282	1283	1284	1285	1286	1287
2410	1288	1289	1290	1291	1292	1293	1294	1295
2420	1296	1297	1298	1299	1300	1301	1302	1303
2430	1304	1305	1306	1307	1308	1309	1310	1311
2440	1312	1313	1314	1315	1316	1317	1318	1319
2450	1320	1321	1322	1323	1324	1325	1326	1327
2460	1328	1329	1330	1331	1332	1333	1334	1335
2470	1336	1337	1338	1339	1340	1341	1342	1343
2500	1344	1345	1346	1347	1348	1349	1350	1351
2510	1352	1353	1354	1355	1356	1357	1358	1359
2520	1360	1361	1362	1363	1364	1365	1366	1367
2530	1368	1369	1370	1371	1372	1373	1374	1375
2540	1376	1377	1378	1379	1380	1381	1382	1383
2550	1384	1385	1386	1387	1388	1389	1390	1391
2560	1392	1393	1394	1395	1396	1397	1398	1399
2570	1400	1401	1402	1403	1404	1405	1406	1407
2600	1408	1409	1410	1411	1412	1413	1414	1415
2610	1416	1417	1418	1419	1420	1421	1422	1423
2620	1424	1425	1426	1427	1428	1429	1430	1431
2630	1432	1433	1434	1435	1436	1437	1438	1439
2640	1440	1441	1442	1443	1444	1445	1446	1447
2650	1448	1449	1450	1451	1452	1453	1454	1455
2660	1456	1457	1458	1459	1460	1461	1462	1463
2670	1464	1465	1466	1467	1468	1469	1470	1471
2700	1472	1473	1474	1475	1476	1477	1478	1479
2710	1480	1481	1482	1483	1484	1485	1486	1487
2720	1488	1489	1490	1491	1492	1493	1494	1495
2730	1496	1497	1498	1499	1500	1501	1502	1503
2740	1504	1505	1506	1507	1508	1509	1510	1511
2750	1512	1513	1514	1515	1516	1517	1518	1519
2760	1520	1521	1522	1523	1524	1525	1526	1527
2770	1528	1529	1530	1531	1532	1533	1534	1535

Octal	Decimal
2000 to 2777	1024 to 1535

(Octal) (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
3000	1536	1537	1538	1539	1540	1541	1542	1543
3010	1544	1545	1546	1547	1548	1549	1550	1551
3020	1552	1553	1554	1555	1556	1557	1558	1559
3030	1560	1561	1562	1563	1564	1565	1566	1567
3040	1568	1569	1570	1571	1572	1573	1574	1575
3050	1576	1577	1578	1579	1580	1581	1582	1583
3060	1584	1585	1586	1587	1588	1589	1590	1591
3070	1592	1593	1594	1595	1596	1597	1598	1599
3100	1600	1601	1602	1603	1604	1605	1606	1607
3110	1608	1609	1610	1611	1612	1613	1614	1615
3120	1616	1617	1618	1619	1620	1621	1622	1623
3130	1624	1625	1626	1627	1628	1629	1630	1631
3140	1632	1633	1634	1635	1636	1637	1638	1639
3150	1640	1641	1642	1643	1644	1645	1646	1647
3160	1648	1649	1650	1651	1652	1653	1654	1655
3170	1656	1657	1658	1659	1660	1661	1662	1663
3200	1664	1665	1666	1667	1668	1669	1670	1671
3210	1672	1673	1674	1675	1676	1677	1678	1679
3220	1680	1681	1682	1683	1684	1685	1686	1687
3230	1688	1689	1690	1691	1692	1693	1694	1695
3240	1696	1697	1698	1699	1700	1701	1702	1703
3250	1704	1705	1706	1707	1708	1709	1710	1711
3260	1712	1713	1714	1715	1716	1717	1718	1719
3270	1720	1721	1722	1723	1724	1725	1726	1727
3300	1728	1729	1730	1731	1732	1733	1734	1735
3310	1736	1737	1738	1739	1740	1741	1742	1743
3320	1744	1745	1746	1747	1748	1749	1750	1751
3330	1752	1753	1754	1755	1756	1757	1758	1759
3340	1760	1761	1762	1763	1764	1765	1766	1767
3350	1768	1769	1770	1771	1772	1773	1774	1775
3360	1776	1777	1778	1779	1780	1781	1782	1783
3370	1784	1785	1786	1787	1788	1789	1790	1791

	0	1	2	3	4	5	6	7
3400	1792	1793	1794	1795	1796	1797	1798	1799
3410	1800	1801	1802	1803	1804	1805	1806	1807
3420	1808	1809	1810	1811	1812	1813	1814	1815
3430	1816	1817	1818	1819	1820	1821	1822	1823
3440	1824	1825	1826	1827	1828	1829	1830	1831
3450	1832	1833	1834	1835	1836	1837	1838	1839
3460	1840	1841	1842	1843	1844	1845	1846	1847
3470	1848	1849	1850	1851	1852	1853	1854	1855
3500	1856	1857	1858	1859	1860	1861	1862	1863
3510	1864	1865	1866	1867	1868	1869	1870	1871
3520	1872	1873	1874	1875	1876	1877	1878	1879
3530	1880	1881	1882	1883	1884	1885	1886	1887
3540	1888	1889	1890	1891	1892	1893	1894	1895
3550	1896	1897	1898	1899	1900	1901	1902	1903
3560	1904	1905	1906	1907	1908	1909	1910	1911
3570	1912	1913	1914	1915	1916	1917	1918	1919
3600	1920	1921	1922	1923	1924	1925	1926	1927
3610	1928	1929	1930	1931	1932	1933	1934	1935
3620	1936	1937	1938	1939	1940	1941	1942	1943
3630	1944	1945	1946	1947	1948	1949	1950	1951
3640	1952	1953	1954	1955	1956	1957	1958	1959
3650	1960	1961	1962	1963	1964	1965	1966	1967
3660	1968	1969	1970	1971	1972	1973	1974	1975
3670	1976	1977	1978	1979	1980	1981	1982	1983
3700	1984	1985	1986	1987	1988	1989	1990	1991
3710	1992	1993	1994	1995	1996	1997	1998	1999
3720	2000	2001	2002	2003	2004	2005	2006	2007
3730	2008	2009	2010	2011	2012	2013	2014	2015
3740	2016	2017	2018	2019	2020	2021	2022	2023
3750	2024	2025	2026	2027	2028	2029	2030	2031
3760	2032	2033	2034	2035	2036	2037	2038	2039
3770	2040	2041	2042	2043	2044	2045	2046	2047

Octal	Decimal
3000 to 3777	1536 to 2047

(Octal) (Decimal)

95118A. F4

APPENDIX F (CONT'D)

NUMERICAL INFORMATION

OCTAL-DECIMAL INTEGER CONVERSION TABLE

4000 to 4777 (Octal)	2048 to 2559 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
4000	2048	2049	2050	2051	2052	2053	2054	2055
4010	2056	2057	2058	2059	2060	2061	2062	2063
4020	2064	2065	2066	2067	2068	2069	2070	2071
4030	2072	2073	2074	2075	2076	2077	2078	2079
4040	2080	2081	2082	2083	2084	2085	2086	2087
4050	2088	2089	2090	2091	2092	2093	2094	2095
4060	2096	2097	2098	2099	2100	2101	2102	2103
4070	2104	2105	2106	2107	2108	2109	2110	2111
4100	2112	2113	2114	2115	2116	2117	2118	2119
4110	2120	2121	2122	2123	2124	2125	2126	2127
4120	2128	2129	2130	2131	2132	2133	2134	2135
4130	2136	2137	2138	2139	2140	2141	2142	2143
4140	2144	2145	2146	2147	2148	2149	2150	2151
4150	2152	2153	2154	2155	2156	2157	2158	2159
4160	2160	2161	2162	2163	2164	2165	2166	2167
4170	2168	2169	2170	2171	2172	2173	2174	2175
4200	2176	2177	2178	2179	2180	2181	2182	2183
4210	2184	2185	2186	2187	2188	2189	2190	2191
4220	2192	2193	2194	2195	2196	2197	2198	2199
4230	2200	2201	2202	2203	2204	2205	2206	2207
4240	2208	2209	2210	2211	2212	2213	2214	2215
4250	2216	2217	2218	2219	2220	2221	2222	2223
4260	2224	2225	2226	2227	2228	2229	2230	2231
4270	2232	2233	2234	2235	2236	2237	2238	2239
4300	2240	2241	2242	2243	2244	2245	2246	2247
4310	2248	2249	2250	2251	2252	2253	2254	2255
4320	2256	2257	2258	2259	2260	2261	2262	2263
4330	2264	2265	2266	2267	2268	2269	2270	2271
4340	2272	2273	2274	2275	2276	2277	2278	2279
4350	2280	2281	2282	2283	2284	2285	2286	2287
4360	2288	2289	2290	2291	2292	2293	2294	2295
4370	2296	2297	2298	2299	2300	2301	2302	2303

	0	1	2	3	4	5	6	7
4400	2304	2305	2306	2307	2308	2309	2310	2311
4410	2312	2313	2314	2315	2316	2317	2318	2319
4420	2320	2321	2322	2323	2324	2325	2326	2327
4430	2328	2329	2330	2331	2332	2333	2334	2335
4440	2336	2337	2338	2339	2340	2341	2342	2343
4450	2344	2345	2346	2347	2348	2349	2350	2351
4460	2352	2353	2354	2355	2356	2357	2358	2359
4470	2360	2361	2362	2363	2364	2365	2366	2367
4500	2368	2369	2370	2371	2372	2373	2374	2375
4510	2376	2377	2378	2379	2380	2381	2382	2383
4520	2384	2385	2386	2387	2388	2389	2390	2391
4530	2392	2393	2394	2395	2396	2397	2398	2399
4540	2400	2401	2402	2403	2404	2405	2406	2407
4550	2408	2409	2410	2411	2412	2413	2414	2415
4560	2416	2417	2418	2419	2420	2421	2422	2423
4570	2424	2425	2426	2427	2428	2429	2430	2431
4600	2432	2433	2434	2435	2436	2437	2438	2439
4610	2440	2441	2442	2443	2444	2445	2446	2447
4620	2448	2449	2450	2451	2452	2453	2454	2455
4630	2456	2457	2458	2459	2460	2461	2462	2463
4640	2464	2465	2466	2467	2468	2469	2470	2471
4650	2472	2473	2474	2475	2476	2477	2478	2479
4660	2480	2481	2482	2483	2484	2485	2486	2487
4670	2488	2489	2490	2491	2492	2493	2494	2495
4700	2496	2497	2498	2499	2500	2501	2502	2503
4710	2504	2505	2506	2507	2508	2509	2510	2511
4720	2512	2513	2514	2515	2516	2517	2518	2519
4730	2520	2521	2522	2523	2524	2525	2526	2527
4740	2528	2529	2530	2531	2532	2533	2534	2535
4750	2536	2537	2538	2539	2540	2541	2542	2543
4760	2544	2545	2546	2547	2548	2549	2550	2551
4770	2552	2553	2554	2555	2556	2557	2558	2559

5000 to 5777 (Octal)	2560 to 3071 (Decimal)

	0	1	2	3	4	5	6	7
5000	2560	2561	2562	2563	2564	2565	2566	2567
5010	2568	2569	2570	2571	2572	2573	2574	2575
5020	2576	2577	2578	2579	2580	2581	2582	2583
5030	2584	2585	2586	2587	2588	2589	2590	2591
5040	2592	2593	2594	2595	2596	2597	2598	2599
5050	2600	2601	2602	2603	2604	2605	2606	2607
5060	2608	2609	2610	2611	2612	2613	2614	2615
5070	2616	2617	2618	2619	2620	2621	2622	2623
5100	2624	2625	2626	2627	2628	2629	2630	2631
5110	2632	2633	2634	2635	2636	2637	2638	2639
5120	2640	2641	2642	2643	2644	2645	2646	2647
5130	2648	2649	2650	2651	2652	2653	2654	2655
5140	2656	2657	2658	2659	2660	2661	2662	2663
5150	2664	2665	2666	2667	2668	2669	2670	2671
5160	2672	2673	2674	2675	2676	2677	2678	2679
5170	2680	2681	2682	2683	2684	2685	2686	2687
5200	2688	2689	2690	2691	2692	2693	2694	2695
5210	2696	2697	2698	2699	2700	2701	2702	2703
5220	2704	2705	2706	2707	2708	2709	2710	2711
5230	2712	2713	2714	2715	2716	2717	2718	2719
5240	2720	2721	2722	2723	2724	2725	2726	2727
5250	2728	2729	2730	2731	2732	2733	2734	2735
5260	2736	2737	2738	2739	2740	2741	2742	2743
5270	2744	2745	2746	2747	2748	2749	2750	2751
5300	2752	2753	2754	2755	2756	2757	2758	2759
5310	2760	2761	2762	2763	2764	2765	2766	2767
5320	2768	2769	2770	2771	2772	2773	2774	2775
5330	2776	2777	2778	2779	2780	2781	2782	2783
5340	2784	2785	2786	2787	2788	2789	2790	2791
5350	2792	2793	2794	2795	2796	2797	2798	2799
5360	2800	2801	2802	2803	2804	2805	2806	2807
5370	2808	2809	2810	2811	2812	2813	2814	2815

	0	1	2	3	4	5	6	7
5400	2816	2817	2818	2819	2820	2821	2822	2823
5410	2824	2825	2826	2827	2828	2829	2830	2831
5420	2832	2833	2834	2835	2836	2837	2838	2839
5430	2840	2841	2842	2843	2844	2845	2846	2847
5440	2848	2849	2850	2851	2852	2853	2854	2855
5450	2856	2857	2858	2859	2860	2861	2862	2863
5460	2864	2865	2866	2867	2868	2869	2870	2871
5470	2872	2873	2874	2875	2876	2877	2878	2879
5500	2880	2881	2882	2883	2884	2885	2886	2887
5510	2888	2889	2890	2891	2892	2893	2894	2895
5520	2896	2897	2898	2899	2900	2901	2902	2903
5530	2904	2905	2906	2907	2908	2909	2910	2911
5540	2912	2913	2914	2915	2916	2917	2918	2919
5550	2920	2921	2922	2923	2924	2925	2926	2927
5560	2928	2929	2930	2931	2932	2933	2934	2935
5570	2936	2937	2938	2939	2940	2941	2942	2943
5600	2944	2945	2946	2947	2948	2949	2950	2951
5610	2952	2953	2954	2955	2956	2957	2958	2959
5620	2960	2961	2962	2963	2964	2965	2966	2967
5630	2968	2969	2970	2971	2972	2973	2974	2975
5640	2976	2977	2978	2979	2980	2981	2982	2983
5650	2984	2985	2986	2987	2988	2989	2990	2991
5660	2992	2993	2994	2995	2996	2997	2998	2999
5670	3000	3001	3002	3003	3004	3005	3006	3007
5700	3008	3009	3010	3011	3012	3013	3014	3015
5710	3016	3017	3018	3019	3020	3021	3022	3023
5720	3024	3025	3026	3027	3028	3029	3030	3031
5730	3032	3033	3034	3035	3036	3037	3038	3039
5740	3040	3041	3042	3043	3044	3045	3046	3047
5750	3048	3049	3050	3051	3052	3053	3054	3055
5760	3056	3057	3058	3059	3060	3061	3062	3063
5770	3064	3065	3066	3067	3068	3069	3070	3071

95118A. F5

APPENDIX F (CONT'D)

NUMERICAL INFORMATION

OCTAL-DECIMAL INTEGER CONVERSION TABLE

	0	1	2	3	4	5	6	7
6000	3072	3073	3074	3075	3076	3077	3078	3079
6010	3080	3081	3082	3083	3084	3085	3086	3087
6020	3088	3089	3090	3091	3092	3093	3094	3095
6030	3096	3097	3098	3099	3100	3101	3102	3103
6040	3104	3105	3106	3107	3108	3109	3110	3111
6050	3112	3113	3114	3115	3116	3117	3118	3119
6060	3120	3121	3122	3123	3124	3125	3126	3127
6070	3128	3129	3130	3131	3132	3133	3134	3135
6100	3136	3137	3138	3139	3140	3141	3142	3143
6110	3144	3145	3146	3147	3148	3149	3150	3151
6120	3152	3153	3154	3155	3156	3157	3158	3159
6130	3160	3161	3162	3163	3164	3165	3166	3167
6140	3168	3169	3170	3171	3172	3173	3174	3175
6150	3176	3177	3178	3179	3180	3181	3182	3183
6160	3184	3185	3186	3187	3188	3189	3190	3191
6170	3192	3193	3194	3195	3196	3197	3198	3199
6200	3200	3201	3202	3203	3204	3205	3206	3207
6210	3208	3209	3210	3211	3212	3213	3214	3215
6220	3216	3217	3218	3219	3220	3221	3222	3223
6230	3224	3225	3226	3227	3228	3229	3230	3231
6240	3232	3233	3234	3235	3236	3237	3238	3239
6250	3240	3241	3242	3243	3244	3245	3246	3247
6260	3248	3249	3250	3251	3252	3253	3254	3255
6270	3256	3257	3258	3259	3260	3261	3262	3263
6300	3264	3265	3266	3267	3268	3269	3270	3271
6310	3272	3273	3274	3275	3276	3277	3278	3279
6320	3280	3281	3282	3283	3284	3285	3286	3287
6330	3288	3289	3290	3291	3292	3293	3294	3295
6340	3296	3297	3298	3299	3300	3301	3302	3303
6350	3304	3305	3306	3307	3308	3309	3310	3311
6360	3312	3313	3314	3315	3316	3317	3318	3319
6370	3320	3321	3322	3323	3324	3325	3326	3327

	0	1	2	3	4	5	6	7
6400	3328	3329	3330	3331	3332	3333	3334	3335
6410	3336	3337	3338	3339	3340	3341	3342	3343
6420	3344	3345	3346	3347	3348	3349	3350	3351
6430	3352	3353	3354	3355	3356	3357	3358	3359
6440	3360	3361	3362	3363	3364	3365	3366	3367
6450	3368	3369	3370	3371	3372	3373	3374	3375
6460	3376	3377	3378	3379	3380	3381	3382	3383
6470	3384	3385	3386	3387	3388	3389	3390	3391
6500	3392	3393	3394	3395	3396	3397	3398	3399
6510	3400	3401	3402	3403	3404	3405	3406	3407
6520	3408	3409	3410	3411	3412	3413	3414	3415
6530	3416	3417	3418	3419	3420	3421	3422	3423
6540	3424	3425	3426	3427	3428	3429	3430	3431
6550	3432	3433	3434	3435	3436	3437	3438	3439
6560	3440	3441	3442	3443	3444	3445	3446	3447
6570	3448	3449	3450	3451	3452	3453	3454	3455
6600	3456	3457	3458	3459	3460	3461	3462	3463
6610	3464	3465	3466	3467	3468	3469	3470	3471
6620	3472	3473	3474	3475	3476	3477	3478	3479
6630	3480	3481	3482	3483	3484	3485	3486	3487
6640	3488	3489	3490	3491	3492	3493	3494	3495
6650	3496	3497	3498	3499	3500	3501	3502	3503
6660	3504	3505	3506	3507	3508	3509	3510	3511
6670	3512	3513	3514	3515	3516	3517	3518	3519
6700	3520	3521	3522	3523	3524	3525	3526	3527
6710	3528	3529	3530	3531	3532	3533	3534	3535
6720	3536	3537	3538	3539	3540	3541	3542	3543
6730	3544	3545	3546	3547	3548	3549	3550	3551
6740	3552	3553	3554	3555	3556	3557	3558	3559
6750	3560	3561	3562	3563	3564	3565	3566	3567
6760	3568	3569	3570	3571	3572	3573	3574	3575
6770	3576	3577	3578	3579	3580	3581	3582	3583

6000	3072
to	to
6777	3583
(Octal)	(Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
7000	3584	3585	3586	3587	3588	3589	3590	3591
7010	3592	3593	3594	3595	3596	3597	3598	3599
7020	3600	3601	3602	3603	3604	3605	3606	3607
7030	3608	3609	3610	3611	3612	3613	3614	3615
7040	3616	3617	3618	3619	3620	3621	3622	3623
7050	3624	3625	3626	3627	3628	3629	3630	3631
7060	3632	3633	3634	3635	3636	3637	3638	3639
7070	3640	3641	3642	3643	3644	3645	3646	3647
7100	3648	3649	3650	3651	3652	3653	3654	3655
7110	3656	3657	3658	3659	3660	3661	3662	3663
7120	3664	3665	3666	3667	3668	3669	3670	3671
7130	3672	3673	3674	3675	3676	3677	3678	3679
7140	3680	3681	3682	3683	3684	3685	3686	3687
7150	3688	3689	3690	3691	3692	3693	3694	3695
7160	3696	3697	3698	3699	3700	3701	3702	3703
7170	3704	3705	3706	3707	3708	3709	3710	3711
7200	3712	3713	3714	3715	3716	3717	3718	3719
7210	3720	3721	3722	3723	3724	3725	3726	3727
7220	3728	3729	3730	3731	3732	3733	3734	3735
7230	3736	3737	3738	3739	3740	3741	3742	3743
7240	3744	3745	3746	3747	3748	3749	3750	3751
7250	3752	3753	3754	3755	3756	3757	3758	3759
7260	3760	3761	3762	3763	3764	3765	3766	3767
7270	3768	3769	3770	3771	3772	3773	3774	3775
7300	3776	3777	3778	3779	3780	3781	3782	3783
7310	3784	3785	3786	3787	3788	3789	3790	3791
7320	3792	3793	3794	3795	3796	3797	3798	3799
7330	3800	3801	3802	3803	3804	3805	3806	3807
7340	3808	3809	3810	3811	3812	3813	3814	3815
7350	3816	3817	3818	3819	3820	3821	3822	3823
7360	3824	3825	3826	3827	3828	3829	3830	3831
7370	3832	3833	3834	3835	3836	3837	3838	3839

	0	1	2	3	4	5	6	7
7400	3840	3841	3842	3843	3844	3845	3846	3847
7410	3848	3849	3850	3851	3852	3853	3854	3855
7420	3856	3857	3858	3859	3860	3861	3862	3863
7430	3864	3865	3866	3867	3868	3869	3870	3871
7440	3872	3873	3874	3875	3876	3877	3878	3879
7450	3880	3881	3882	3883	3884	3885	3886	3887
7460	3888	3889	3890	3891	3892	3893	3894	3895
7470	3896	3897	3898	3899	3900	3901	3902	3903
7500	3904	3905	3906	3907	3908	3909	3910	3911
7510	3912	3913	3914	3915	3916	3917	3918	3919
7520	3920	3921	3922	3923	3924	3925	3926	3927
7530	3928	3929	3930	3931	3932	3933	3934	3935
7540	3936	3937	3938	3939	3940	3941	3942	3943
7550	3944	3945	3946	3947	3948	3949	3950	3951
7560	3952	3953	3954	3955	3956	3957	3958	3959
7570	3960	3961	3962	3963	3964	3965	3966	3967
7600	3968	3969	3970	3971	3972	3973	3974	3975
7610	3976	3977	3978	3979	3980	3981	3982	3983
7620	3984	3985	3986	3987	3988	3989	3990	3991
7630	3992	3993	3994	3995	3996	3997	3998	3999
7640	4000	4001	4002	4003	4004	4005	4006	4007
7650	4008	4009	4010	4011	4012	4013	4014	4015
7660	4016	4017	4018	4019	4020	4021	1022	4023
7670	4024	4025	4026	4027	4028	4029	4030	4031
7700	4032	4033	4034	4035	4036	4037	4038	4039
7710	4040	4041	4042	4043	4044	4045	4046	4047
7720	4048	4049	4050	4051	4052	4053	4054	4055
7730	4056	4057	4058	4059	4060	4061	4062	4063
7740	4064	4065	4066	4067	4068	4069	4070	4071
7750	4072	4073	4074	4075	4076	4077	4078	4079
7760	4080	4081	4082	4083	4084	4085	4086	4087
7770	4088	4089	4090	4091	4092	4093	4094	4095

7000	3584
to	to
7777	4095
(Octal)	(Decimal)

APPENDIX F (CONT'D)

NUMERICAL INFORMATION

OCTAL-DECIMAL FRACTION CONVERSION TABLE

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000	.000000	.100	.125000	.200	.250000	.300	.375000
.001	.001953	.101	.126953	.201	.251953	.301	.376953
.002	.003906	.102	.128906	.202	.253906	.302	.378906
.003	.005859	.103	.130859	.203	.255859	.303	.380859
.004	.007812	.104	.132812	.204	.257812	.304	.382812
.005	.009765	.105	.134765	.205	.259765	.305	.384765
.006	.011718	.106	.136718	.206	.261718	.306	.386718
.007	.013671	.107	.138671	.207	.263671	.307	.388671
.010	.015625	.110	.140625	.210	.265625	.310	.390625
.011	.017578	.111	.142578	.211	.267578	.311	.392578
.012	.019531	.112	.144531	.212	.269531	.312	.394531
.013	.021484	.113	.146484	.213	.271484	.313	.396484
.014	.023437	.114	.148437	.214	.273437	.314	.398437
.015	.025390	.115	.150390	.215	.275390	.315	.400390
.016	.027343	.116	.152343	.216	.277343	.316	.402343
.017	.029296	.117	.154296	.217	.279296	.317	.404296
.020	.031250	.120	.156250	.220	.281250	.320	.406250
.021	.033203	.121	.158203	.221	.283203	.321	.408203
.022	.035156	.122	.160156	.222	.285156	.322	.410156
.023	.037109	.123	.162109	.223	.287109	.323	.412109
.024	.039062	.124	.164062	.224	.289062	.324	.414062
.025	.041015	.125	.166015	.225	.291015	.325	.416015
.026	.042968	.126	.167968	.226	.292968	.326	.417968
.027	.044921	.127	.169921	.227	.294921	.327	.419921
.030	.046875	.130	.171875	.230	.296875	.330	.421875
.031	.048828	.131	.173828	.231	.298828	.331	.423828
.032	.050781	.132	.175781	.232	.300781	.332	.425781
.033	.052734	.133	.177734	.233	.302734	.333	.427734
.034	.054687	.134	.179687	.234	.304687	.334	.429687
.035	.056640	.135	.181640	.235	.306640	.335	.431640
.036	.058593	.136	.183593	.236	.308593	.336	.433593
.037	.060546	.137	.185546	.237	.310546	.337	.435546
.040	.062500	.140	.187500	.240	.312500	.340	.437500
.041	.064453	.141	.189453	.241	.314453	.341	.439453
.042	.066406	.142	.191406	.242	.316406	.342	.441406
.043	.068359	.143	.193359	.243	.318359	.343	.443359
.044	.070312	.144	.195312	.244	.320312	.344	.445312
.045	.072265	.145	.197265	.245	.322265	.345	.447265
.046	.074218	.146	.199218	.246	.324218	.346	.449218
.047	.076171	.147	.201171	.247	.326171	.347	.451171
.050	.078125	.150	.203125	.250	.328125	.350	.453125
.051	.080078	.151	.205078	.251	.330078	.351	.455078
.052	.082031	.152	.207031	.252	.332031	.352	.457031
.053	.083984	.153	.208984	.253	.333984	.353	.458984
.054	.085937	.154	.210937	.254	.335937	.354	.460937
.055	.087890	.155	.212890	.255	.337890	.355	.462890
.056	.089843	.156	.214843	.256	.339843	.356	.464843
.057	.091796	.157	.216796	.257	.341796	.357	.466796
.060	.093750	.160	.218750	.260	.343750	.360	.468750
.061	.095703	.161	.220703	.261	.345703	.361	.470703
.062	.097656	.162	.222656	.262	.347656	.362	.472656
.063	.099609	.163	.224609	.263	.349609	.363	.474609
.064	.101562	.164	.226562	.264	.351562	.364	.476562
.065	.103515	.165	.228515	.265	.353515	.365	.478515
.066	.105468	.166	.230468	.266	.355468	.366	.480468
.067	.107421	.167	.232421	.267	.357421	.367	.482421
.070	.109375	.170	.234375	.270	.359375	.370	.484375
.071	.111328	.171	.236328	.271	.361328	.371	.486328
.072	.113281	.172	.238281	.272	.363281	.372	.488281
.073	.115234	.173	.240234	.273	.365234	.373	.490234
.074	.117187	.174	.242187	.274	.367187	.374	.492187
.075	.119140	.175	.244140	.275	.369140	.375	.494140
.076	.121093	.176	.246093	.276	.371093	.376	.496093
.077	.123046	.177	.248046	.277	.373046	.377	.498046

APPENDIX F (CONT'D)

NUMERICAL INFORMATION

OCTAL-DECIMAL FRACTION CONVERSION TABLE

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000000	.000000	.000100	.000244	.000200	.000488	.000300	.000732
.000001	.000003	.000101	.000247	.000201	.000492	.000301	.000736
.000002	.000007	.000102	.000251	.000202	.000495	.000302	.000740
.000003	.000011	.000103	.000255	.000203	.000499	.000303	.000743
.000004	.000015	.000104	.000259	.000204	.000503	.000304	.000747
.000005	.000019	.000105	.000263	.000205	.000507	.000305	.000751
.000006	.000022	.000106	.000267	.000206	.000511	.000306	.000755
.000007	.000026	.000107	.000270	.000207	.000514	.000307	.000759
.000010	.000030	.000110	.000274	.000210	.000518	.000310	.000762
.000011	.000034	.000111	.000278	.000211	.000522	.000311	.000766
.000012	.000038	.000112	.000282	.000212	.000526	.000312	.000770
.000013	.000041	.000113	.000286	.000213	.000530	.000313	.000774
.000014	.000045	.000114	.000289	.000214	.000534	.000314	.000778
.000015	.000049	.000115	.000293	.000215	.000537	.000315	.000782
.000016	.000053	.000116	.000297	.000216	.000541	.000316	.000785
.000017	.000057	.000117	.000301	.000217	.000545	.000317	.000789
.000020	.000061	.000120	.000305	.000220	.000549	.000320	.000793
.000021	.000064	.000121	.000308	.000221	.000553	.000321	.000797
.000022	.000068	.000122	.000312	.000222	.000556	.000322	.000801
.000023	.000072	.000123	.000316	.000223	.000560	.000323	.000805
.000024	.000076	.000124	.000320	.000224	.000564	.000324	.000808
.000025	.000080	.000125	.000324	.000225	.000568	.000325	.000812
.000026	.000083	.000126	.000328	.000226	.000572	.000326	.000816
.000027	.000087	.000127	.000331	.000227	.000576	.000327	.000820
.000030	.000091	.000130	.000335	.000230	.000579	.000330	.000823
.000031	.000095	.000131	.000339	.000231	.000583	.000331	.000827
.000032	.000099	.000132	.000343	.000232	.000587	.000332	.000831
.000033	.000102	.000133	.000347	.000233	.000591	.000333	.000835
.000034	.000106	.000134	.000350	.000234	.000595	.000334	.000839
.000035	.000110	.000135	.000354	.000235	.000598	.000335	.000843
.000036	.000114	.000136	.000358	.000236	.000602	.000336	.000846
.000037	.000118	.000137	.000362	.000237	.000606	.000337	.000850
.000040	.000122	.000140	.000366	.000240	.000610	.000340	.000854
.000041	.000125	.000141	.000370	.000241	.000614	.000341	.000858
.000042	.000129	.000142	.000373	.000242	.000617	.000342	.000862
.000043	.000133	.000143	.000377	.000243	.000621	.000343	.000865
.000044	.000137	.000144	.000381	.000244	.000625	.000344	.000869
.000045	.000141	.000145	.000385	.000245	.000629	.000345	.000873
.000046	.000144	.000146	.000389	.000246	.000633	.000346	.000877
.000047	.000148	.000147	.000392	.000247	.000637	.000347	.000881
.000050	.000152	.000150	.000396	.000250	.000640	.000350	.000885
.000051	.000156	.000151	.000400	.000251	.000644	.000351	.000888
.000052	.000160	.000152	.000404	.000252	.000648	.000352	.000892
.000053	.000164	.000153	.000408	.000253	.000652	.000353	.000896
.000054	.000167	.000154	.000411	.000254	.000656	.000354	.000900
.000055	.000171	.000155	.000415	.000255	.000659	.000355	.000904
.000056	.000175	.000156	.000419	.000256	.000663	.000356	.000907
.000057	.000179	.000157	.000423	.000257	.000667	.000357	.000911
.000060	.000183	.000160	.000427	.000260	.000671	.000360	.000915
.000061	.000186	.000161	.000431	.000261	.000675	.000361	.000919
.000062	.000190	.000162	.000434	.000262	.000679	.000362	.000923
.000063	.000194	.000163	.000438	.000263	.000682	.000363	.000926
.000064	.000198	.000164	.000442	.000264	.000686	.000364	.000930
.000065	.000202	.000165	.000446	.000265	.000690	.000365	.000934
.000066	.000205	.000166	.000450	.000266	.000694	.000366	.000938
.000067	.000209	.000167	.000453	.000267	.000698	.000367	.000942
.000070	.000213	.000170	.000457	.000270	.000701	.000370	.000946
.000071	.000217	.000171	.000461	.000271	.000705	.000371	.000949
.000072	.000221	.000172	.000465	.000272	.000709	.000372	.000953
.000073	.000225	.000173	.000469	.000273	.000713	.000373	.000957
.000074	.000228	.000174	.000473	.000274	.000717	.000374	.000961
.000075	.000232	.000175	.000476	.000275	.000720	.000375	.000965
.000076	.000236	.000176	.000480	.000276	.000724	.000376	.000968
.000077	.000240	.000177	.000484	.000277	.000728	.000377	.000972

APPENDIX F (CONT'D)

NUMERICAL INFORMATION

OCTAL-DECIMAL FRACTION CONVERSION TABLE

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000400	.000976	.000500	.001220	.000600	.001464	.000700	.001708
.000401	.000980	.000501	.001224	.000601	.001468	.000701	.001712
.000402	.000984	.000502	.001228	.000602	.001472	.000702	.001716
.000403	.000988	.000503	.001232	.000603	.001476	.000703	.001720
.000404	.000991	.000504	.001235	.000604	.001480	.000704	.001724
.000405	.000995	.000505	.001239	.000605	.001483	.000705	.001728
.000406	.000999	.000506	.001243	.000606	.001487	.000706	.001731
.000407	.001003	.000507	.001247	.000607	.001491	.000707	.001735
.000410	.001007	.000510	.001251	.000610	.001495	.000710	.001739
.000411	.001010	.000511	.001255	.000611	.001499	.000711	.001743
.000412	.001014	.000512	.001258	.000612	.001502	.000712	.001747
.000413	.001018	.000513	.001262	.000613	.001506	.000713	.001750
.000414	.001022	.000514	.001266	.000614	.001510	.000714	.001754
.000415	.001026	.000515	.001270	.000615	.001514	.000715	.001758
.000416	.001029	.000516	.001274	.000616	.001518	.000716	.001762
.000417	.001033	.000517	.001277	.000617	.001522	.000717	.001766
.000420	.001037	.000520	.001281	.000620	.001525	.000720	.001770
.000421	.001041	.000521	.001285	.000621	.001529	.000721	.001773
.000422	.001045	.000522	.001289	.000622	.001533	.000722	.001777
.000423	.001049	.000523	.001293	.000623	.001537	.000723	.001781
.000424	.001052	.000524	.001296	.000624	.001541	.000724	.001785
.000425	.001056	.000525	.001300	.000625	.001544	.000725	.001789
.000426	.001060	.000526	.001304	.000626	.001548	.000726	.001792
.000427	.001064	.000527	.001308	.000627	.001552	.000727	.001796
.000430	.001068	.000530	.001312	.000630	.001556	.000730	.001800
.000431	.001071	.000531	.001316	.000631	.001560	.000731	.001804
.000432	.001075	.000532	.001319	.000632	.001564	.000732	.001808
.000433	.001079	.000533	.001323	.000633	.001567	.000733	.001811
.000434	.001083	.000534	.001327	.000634	.001571	.000734	.001815
.000435	.001087	.000535	.001331	.000635	.001575	.000735	.001819
.000436	.001091	.000536	.001335	.000636	.001579	.000736	.001823
.000437	.001094	.000537	.001338	.000637	.001583	.000737	.001827
.000440	.001098	.000540	.001342	.000640	.001586	.000740	.001831
.000441	.001102	.000541	.001346	.000641	.001590	.000741	.001834
.000442	.001106	.000542	.001350	.000642	.001594	.000742	.001838
.000443	.001110	.000543	.001354	.000643	.001598	.000743	.001842
.000444	.001113	.000544	.001358	.000644	.001602	.000744	.001846
.000445	.001117	.000545	.001361	.000645	.001605	.000745	.001850
.000446	.001121	.000546	.001365	.000646	.001609	.000746	.001853
.000447	.001125	.000547	.001369	.000647	.001613	.000747	.001857
.000450	.001129	.000550	.001373	.000650	.001617	.000750	.001861
.000451	.001132	.000551	.001377	.000651	.001621	.000751	.001865
.000452	.001136	.000552	.001380	.000652	.001625	.000752	.001869
.000453	.001140	.000553	.001384	.000653	.001628	.000753	.001873
.000454	.001144	.000554	.001388	.000654	.001632	.000754	.001876
.000455	.001148	.000555	.001392	.000655	.001636	.000755	.001880
.000456	.001152	.000556	.001396	.000656	.001640	.000756	.001884
.000457	.001155	.000557	.001399	.000657	.001644	.000757	.001888
.000460	.001159	.000560	.001403	.000660	.001647	.000760	.001892
.000461	.001163	.000561	.001407	.000661	.001651	.000761	.001895
.000462	.001167	.000562	.001411	.000662	.001655	.000762	.001899
.000463	.001171	.000563	.001415	.000663	.001659	.000763	.001903
.000464	.001174	.000564	.001419	.000664	.001663	.000764	.001907
.000465	.001178	.000565	.001422	.000665	.001667	.000765	.001911
.000466	.001182	.000566	.001426	.000666	.001670	.000766	.001914
.000467	.001186	.000567	.001430	.000667	.001674	.000767	.001918
.000470	.001190	.000570	.001434	.000670	.001678	.000770	.001922
.000471	.001194	.000571	.001438	.000671	.001682	.000771	.001926
.000472	.001197	.000572	.001441	.000672	.001686	.000772	.001930
.000473	.001201	.000573	.001445	.000673	.001689	.000773	.001934
.000474	.001205	.000574	.001449	.000674	.001693	.000774	.001937
.000475	.001209	.000575	.001453	.000675	.001697	.000775	.001941
.000476	.001213	.000576	.001457	.000676	.001701	.000776	.001945
.000477	.001216	.000577	.001461	.000677	.001705	.000777	.001949

APPENDIX F (CONT'D)

NUMERICAL INFORMATION

MATHEMATICAL INFORMATION

POWERS OF TEN ($10^{\pm n}$) IN OCTAL

10^n	n	10^{-n}
1	0	1. 000 000 000 000 000 000 00
12	1	0. 063 146 314 631 463 146 31
144	2	0. 005 075 341 217 270 243 66
1 750	3	0. 000 406 111 564 570 651 77
23 420	4	0. 000 032 155 613 530 704 15
303 240	5	0. 000 002 476 132 610 706 64
3 641 100	6	0. 000 000 206 157 364 055 37
46 113 200	7	0. 000 000 015 327 745 152 75
575 360 400	8	0. 000 000 001 257 143 561 06
7 346 545 000	9	0. 000 000 000 104 560 276 41
112 402 762 000	10	0. 000 000 000 006 676 337 66
1 351 035 564 000	11	0. 000 000 000 000 537 657 77
16 432 451 210 000	12	0. 000 000 000 000 043 136 32
221 411 634 520 000	13	0. 000 000 000 000 003 411 35
2 657 142 036 440 000	14	0. 000 000 000 000 000 264 11
34 327 724 461 500 000	15	0. 000 000 000 000 000 022 01
434 157 115 760 200 000	16	0. 000 000 000 000 000 001 63
5 432 127 413 542 400 000	17	0. 000 000 000 000 000 000 014
67 405 553 164 731 000 000	18	0. 000 000 000 000 000 000 001

VARIOUS CONSTANTS IN OCTAL NOTATION

π = 3.11037552421

$\pi/2$ = 1.04417665210

$1/\pi$ = 0.24276301556

$\sqrt{\pi}$ = 1.61337611067

$\ln \pi$ = 1.11206404435

$\sqrt{10}$ = 3.12305407267

e = 2.55760521305

$1/e$ = 0.27426530661

$\sqrt{e}$ = 1.51411230704

$\log_{10} e$ = 0.33626754251

$\sqrt{2}$ = 1.32404746320

ln 10 = 2.23273067355

APPENDIX G

SEL 840MP COMPUTER INSTRUCTION LIST SUMMARY

CLASS	MNE-MONIC	OP CODE	FUNCTION	PAGE
Load/ Store	LAA	01	Load A-Acc.	2-5
	LBA	02	Load B-Acc.	2-8
	STA	03	Store A-Acc.	2-8
	STB	04	Store B-Acc.	2-8
	LIX	32	Load Index Register	2-8
	STI	33	Store Index Register	2-8
	LBR	37	Load Bank Address Register	2-9
	SBR	40	Store Bank Address Register	2-9
	IAM	44	Interchange A-Acc. and Memory	2-9
	LCS	57	Load Control Switches	2-9
Arithmetic	AMA	05	Add Memory to A-Acc.	2-10
	SMA	06	Subtract Memory from A-Acc.	2-10
	MPX	07	Multiply	2-10
	DIV	10	Divide	2-11
	CNS	20	Convert Number System	2-11
	AAM	31	Add A-Acc. to Memory	2-11
	NEG	56	Negate A-Acc.	2-12
	RNA	60	Round A-Acc.	2-12
	AMX	61	Add Memory to Index Register	2-12
	CSB	00-07	Copy Sign of B-Acc.	2-12
Branch/ Skip Instruction	BRU	11	Unconditional Branch	2-13
	SPB	12	Store, Place and Branch	2-13
	IMS	14	Increment Memory and Skip	2-13
	CMA	15	Compare Memory and A-Acc.	2-13
	BAZ	22	Branch if A-Acc. zero	2-14
	BAN	23	Branch if A-Acc. Negative	2-14
	BAP	24	Branch if A-Acc. Positive	2-14
	BOF	25	Branch on Overflow	2-14
	IIB	34	Increment Index and Branch	2-14
	SMP	35	Skip if Memory Positive	2-14
	PIR	36	Priority Interrupt Return	2-14
	SNS	134	Sense Numbered Switch	2-15
	SAS	00-21	Skip on A-Acc. Sign	2-15
Logical	MEA	26	Memory Exclusive OR A-Acc.	2-15
	MAA	27	Memory AND A-Acc.	2-16
	MOA	30	Memory OR A-Acc.	2-16
	TAI	00-01	Transfer A-Acc. to Index Register	2-16
	TBI	00-02	Transfer B-Acc. to Index Register	2-16
	CLA	00-03	Clear A-Acc.	2-16
	TBA	00-04	Transfer B-Acc. to A-Acc.	2-17
	TAB	00-05	Transfer A-Acc. to B-Acc.	2-17
	IAB	00-06	Interchange A- and B-Acc.	2-17
	ASC	00-20	Complement A-Acc. Sign	2-17

CLASS	MNE-MONIC	OP CODE	FUNCTION	PAGE
Shift	RSA	00-10	Right Shift A-Acc. Arithmetic	2-17
	LSA	00-11	Left Shift A-Acc. Arithmetic	2-18
	FRA	00-12	Full-Right Arithmetic Shift	2-18
	FLA	00-13	Full-Left Arithmetic Shift	2-18
	FRL	00-14	Full-Left Rotate Accumulators	2-18
	RSL	00-15	Right Shift A-Acc. Logical	2-19
	LSL	00-16	Left-Shift A-Acc. Logical	2-19
	FLL	00-17	Full-Left Shift, Logical	2-19
	NOR	00-32	Normalize Acc.	2-19
Control	EXU	16	Execute Instruction at Location	2-20
	PID	0.43	Priority Interrupt Disable	2-20
	PIE	1.43	Priority Interrupt Enable	2-21
	HLT	00-00	Halt	2-21
	NOP	00-22	No Operation	2-21
	EXI	00-24	External Interrupt	2-21
	TAC	00-25	Test and Activate	2-21
Input/ Output	CEU	130	Command External Unit	2-24
	TEU	132	Test External Unit	2-25
	AOP	170	A-Acc. Output to Peripheral Unit	2-25
	AIP	172	A-Acc. Input from Peripheral Unit	2-26
	MOP	174	Memory Word Output to Peripheral Unit	2-26
	MIP	176	Memory Input from Peripheral Unit	2-26
EAU	ELB	41	Extended Load EB-Acc.	2-27
	EAD	45	Extended Add	2-27
	ESU	46	Extended Subtract	2-27
	EMU	47	Extended Multiply	2-28
	EDV	50	Extended Divide	2-28
	ELN	51	Extended Load Negative	2-29
	ELO	52	Extended Load	2-29
	EST	53	Extended Store	2-29
	ECA	21-15	Clear EA-Acc.	2-30
	ECB	21-16	Clear EB-Acc.	2-30
	ESR	00-23	Extended Skip if Ready	2-30
	ENO	21-00	Extended Normalize	2-30
	EIA	21-01	Interchange EA-Acc. and EB-Acc.	2-31
	EBA	21-02	Transfer EB-Acc. to EA-Acc.	2-31
	EAB	21-03	Transfer EA-Acc. to EB-Acc.	2-31
	EUN	21-04	Ext. Unnormalize	2-31
	ESZ	21-05	Ext. Skip if Zero	2-32
	EPS	21-06	Ext. Skip if Positive	2-32
	ESO	21-10	Ext. Skip on Overflow	2-32
	EDP	21-12	Ext. Double Precision Fixed Pt.	2-32
	EFP	21-14	Ext. Floating Pt.	2-32
	ESD	21-17	Ext. Skip if Fixed Pt. Double Precision	2-32
	ESN	21-07	Ext. Skip if Negative	2-32
Program Protect	PON	62	Protect Bit On	2-33
	POF	63	Protect Bit Off	2-33