FABRI-TEK INC.
COMPUTER SYSTEMS

REFERENCE MANUAL

MP 12

MP 12

REFERENCE MANUAL

Prepared by FABRI-TEK Inc.
5901 South County Road 18
Minneapolis, Minnesota 55436

PHONE: 612/935-8811
TWX: 910-576-2913

FABRI-TEK INC.
COMPUTER SYSTEMS

MP12

FIRST EDITION

FABRI-TEK Publication Number 400-0227-00
Printed in the United States of America

PREFACE

This manual is divided into two sections. Section I describes the organization, features, and operation of the FABRI-TEK MP12. Section II contains descriptions and operating procedures for the standard MP12 software.

SECTION I
TABLE OF CONTENTS

SECTION I INTRODUCTION

ORGANIZATION

FUNCTIONAL DESCRIPTION

OPERATION

INPUT-OUTPUT INTERFACE

ASSEMBLY LANGUAGE

SECTION II
TABLE OF CONTENTS

SECTION II INTRODUCTION

SOURCE EDIT UTILITY

ASSEMBLER

BINARY LOADER

DEBUGGING UTILITY

PROCESSOR DIAGNOSTIC

CROSS-ASSEMBLER

APPENDICES

LIST OF ILLUSTRATIONS

SECTION I INTRODUCTION

The FABRI-TEK MP12 is a general purpose, 12-bit computer designed primarily for OEM applications. The basic system, contained in a 2.0 in. wide, 15.0 in. deep, and 9.5 in. high enclosure, consists of the following elements:

- Parallel-logic processor.
- 4096 x 12 random access magnetic core memory.
- Input-Output interface.
- Hardware interrupt facility.
- Power-fail, Auto-Restart.
- Operating console.

Optional FABRI-TEK MP12 features include up to 2048 words of bipolar read-only memory, +5V power supply module, chassis assembly, wire-wrap assembly, extension memory, and standard peripheral interface cards.

Standard FABRI-TEK MP12 software includes an assembler, loader, source edit utility, debugging utility, diagnostics, and an IBM 360/370 cross-assembler.

PROCESSOR

The FABRI-TEK MP12 processor is mounted on a single 9.5 in. by 15.0 in. printed circuit card that contains all logic circuitry necessary for processor operation and interfacing.

MEMORY

The standard FABRI-TEK MP12 memory is a 4K x 12, coincident current magnetic core memory. The memory is mounted on a single 14.6 in. by 9.25 in. printed circuit card that contains all necessary memory electronics and interface circuitry.

INPUT-OUTPUT INTERFACE

The input-output interface incorporates two input-output channels, a processor input-output (PIO) channel, and a direct memory access (DMA) channel. The PIO channel interfaces with the processor via the data input bus and provides simplex character-oriented data transfer capability, at rates of up to 66,000 words-per-second. The DMA channel interfaces directly with the memory, via the data input bus, and provides high-speed, record-oriented data transfer capability at rates of up to 666,666 words-per-second.

HARDWARE INTERRUPT FACILITY

The FABRI-TEK MP12 hardware interrupt facility permits input-output operations to be scheduled under interrupt control. The interrupt facility may be selectively enabled or disabled under program control. When the interrupt facility is enabled, a device generates a processor interrupt each time it is ready to receive or transmit data. Use of the FABRI-TEK MP12 interrupt facility enables input-output operations to be performed simultaneously with computing.

POWER-FAIL, AUTO-RESTART

The power-fail facility provides a processor interrupt when a power low signal is received from the primary power source. Sufficient time is then available to preserve the contents of the working registers in the memory. The auto-restart feature enables processing to resume, at the point of interruption, when power is restored.

OPERATING CONSOLE

The FABRI-TEK MP12 operating console contains all controls and indicators necessary for the operation of the processor. It features a 12-bit switch register, which may be read under program control, LED display indicators, and toggle switches for entering data and operating the processor.

READ-ONLY MEMORY

Up to 2048 words of optional bipolar ROM is available in 512 word increments. A special feature of the FABRI-TEK MP12 permits efficient use of ROM for subroutine programming. An installed ROM covers a corresponding area of core memory and contains a ''window'' into the core memory at each ROM location containing a zero data word. A read or write operation into a zero ROM location will cause an operand to be read from, or written into, the corresponding core memory location. In particular, subroutines may be programmed into the ROM based on the fact that return addresses are automatically stored in, and retrieved from core memory when the first location of a ROM subroutine contains a zero word.

The combinations of power-fail, auto-restart, and read-only memory enable the FABRI-TEK MP12 to function effectively in unattended operating environments.

POWER SUPPLY

The optional power supply provides a regulated DC source of 5 volts at 20 amps to operate the processor and peripheral interface logic. The power supply also includes circuitry to detect a low AC input condition and to provide a power low interrupt signal to the processor. An additional feature of the power supply is a line frequency clock interrupt circuit that provides an interrupt signal to the processor at 16 2/3 ms intervals. The power supply is designed to permit the regulator section to operate with 12-volt battery input; this feature permits operation in a mobile vehicle. The power supply is housed in a double width enclosure (2 X processor enclosure size) that plugs into the chassis assembly without any wiring other than the AC line cord.

CHASSIS ASSEMBLY

The optional chassis assembly mounts in a standard 19 inch rack and provides convenient mounting locations for the processor, power supply, and up to 15 peripheral interface cards. The chassis assembly is designed using a printed circuit backplane for all interconnecting wiring.

WIRE-WRAP ASSEMBLY

Input-output signals, available at the FABRI-TEK MP12 input-output interface connector, are arranged to permit the use of a printed circuit backplane for interconnecting signals between the processor and external input-output device controllers. An optional wire-wrap assembly, having the same physical dimensions as the processor enclosure, accommodates a printed circuit card suitable for mounting up to 140 14-pin, 16-pin, or 24-pin packages. This card connects directly to the printed circuit backplane and may be used to host peripheral interface logic.

EXTENSION MEMORY

The optional MP12 extension memory consists of a 4K x 12 core memory, mounted in a 2.0'' x 15.0'' x 9.5'' enclosure that plugs into the chassis assembly. The extension memory is accessable for word-level read or write operations and is addressed as an input-output device. The access mechanism is such that the memory may be utilized either as a randomly addressable data storage module or a sequentially addressable stack. An additional feature is that each extension memory may be shared by a pair of MP12 processors. This feature permits simultaneous access to common data for multiprocessor configurations. There is no practical limitation to the number of extension memories which may be attached.

PERIPHERAL INTERFACES

A number of optional interface cards are available for a selected set of standard peripherals. The following interfaces are implemented as printed circuit cards that plug into the chassis assembly.

TELETYPE

PIO controller located inside the processor enclosure for ASR 33 teletype. Full-duplex, character-serial operation. Requires teletype modification kit option.

KEYBOARD/PRINTER

PIO controller for 300 BAUD or 2400 BAUD terminal. Full-duplex, character-serial operation. Complete with cables for Texas Instruments Model 733 or equivalent.

HIGH SPEED PAPER TAPE READER INTERFACE

PIO controller for 300 CPS paper tape reader. Complete with cables for Remex Model RRF6300.

HIGH SPEED PAPER TAPE READER/PUNCH INTERFACE

PIO controller for combination 300 CPS paper tape reader and 75 CPS paper tape punch. Complete with cables for Remex Model RAF6375.

STANDARD 80 COLUMN PUNCHED CARD READER INTERFACE

PIO controller for 300 CPM punched card reader with Hollerith to ASCII conversion. Complete with cables for Data Products Model 8330.

HIGH SPEED LINE PRINTER INTERFACE

PIO controller for 300 or 600 LPM 132 column line printer. Complete with cables for Data Products Model 2230.

ASYNCHRONOUS COMMUNCIATIONS INTERFACE

PIO controller for Bell System Data Set Models 103A, 103F, 202C, 202D, or equivalent. Operates with 6, 8, or 12-bit characters at rates of 110, 150, 300, 600, 1200, 1800, 2400, 4800, or 9600 bits-per-second. Includes serial-to-parallel and parallel-to-serial conversion. Complete with cables for data set or terminal.

DIGITAL INPUT INTERFACE

24 line digital input interface available with TTL or optically isolated inputs. PIO controller. Complete with cable with spade lugs on one end.

DIGITAL OUTPUT INTERFACE

24-line digital output interface available with TTL (open collector) or contact closure output. PIO controller. Complete with cable with spade lugs on one end.

ANALOG-TO-DIGITAL CONVERTER

12-bit analog-to-digital converter with 50 us conversion time and 8 differential inputs. PIO controller.

DIGITAL-TO-ANALOG CONVERTER

Four 12-bit digital-to-analog converters with 5 us conversion time and voltage outputs. PIO controller.

MAGNETIC TAPE DRIVE INTERFACE

DMA controller for Industry Standard 800 BPI 9-track tape drive with standard error checking. Comes complete with cables for Pertec Model 5840.

MAGNETIC TAPE CASSETTE INTERFACE

PIO controller for dual cassette tape system. Complete with cables for Remex Model RCS1300.

SOFTWARE

Standard software for the FABRI-TEK MP12 includes an assembler, loader, debugging utility, source edit utility, and diagnostic programs. The assembler translates symbolic assembly language programs into executable machine programs. The loader loads object tapes produced by the assembler or debugging utility. The debugging utility aids program checkout and features multiple breakpoints, instruction trace, and several other standard functions. The source edit utility is used to generate assembly language source tapes or modify existing source tapes. The diagnostics are used to verify MP12 processor operation. The cross-assembler enables programs written for the MP12 to be assembled on IBM 360/370 series computers.

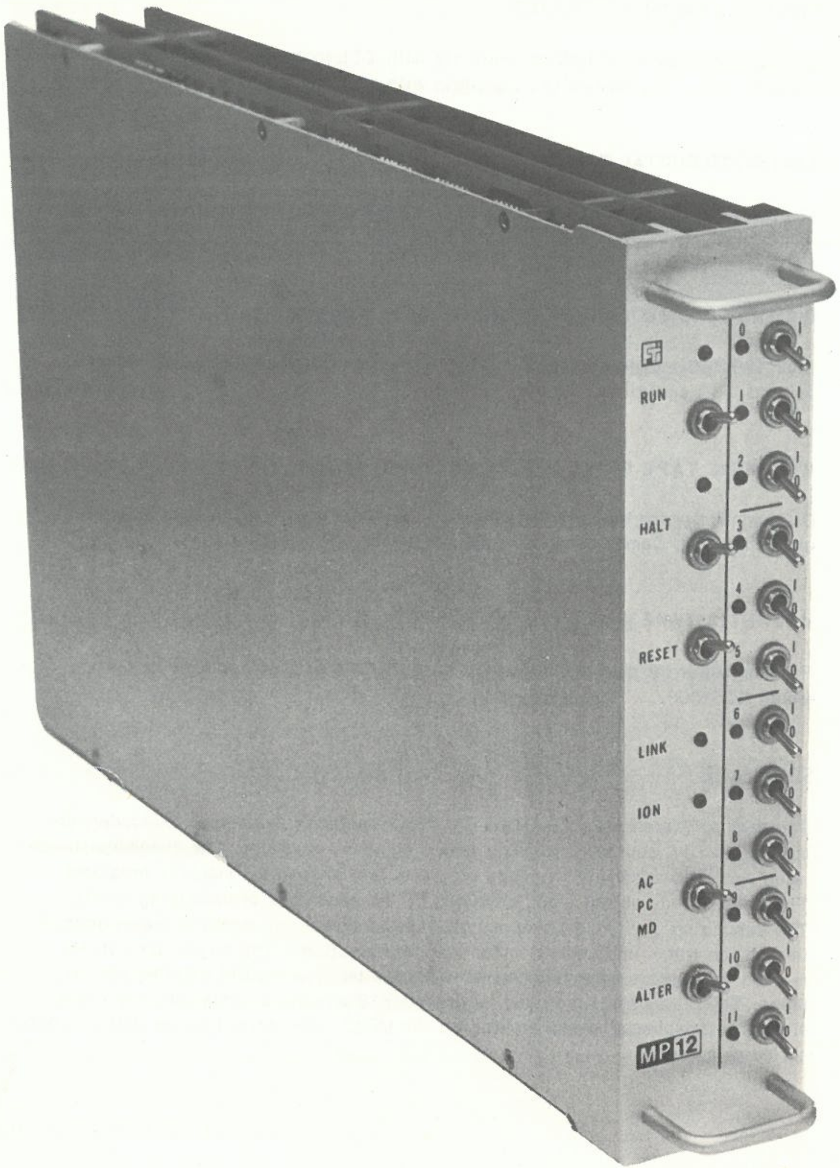


Figure 1-1
MP12 MICROPROCESSOR

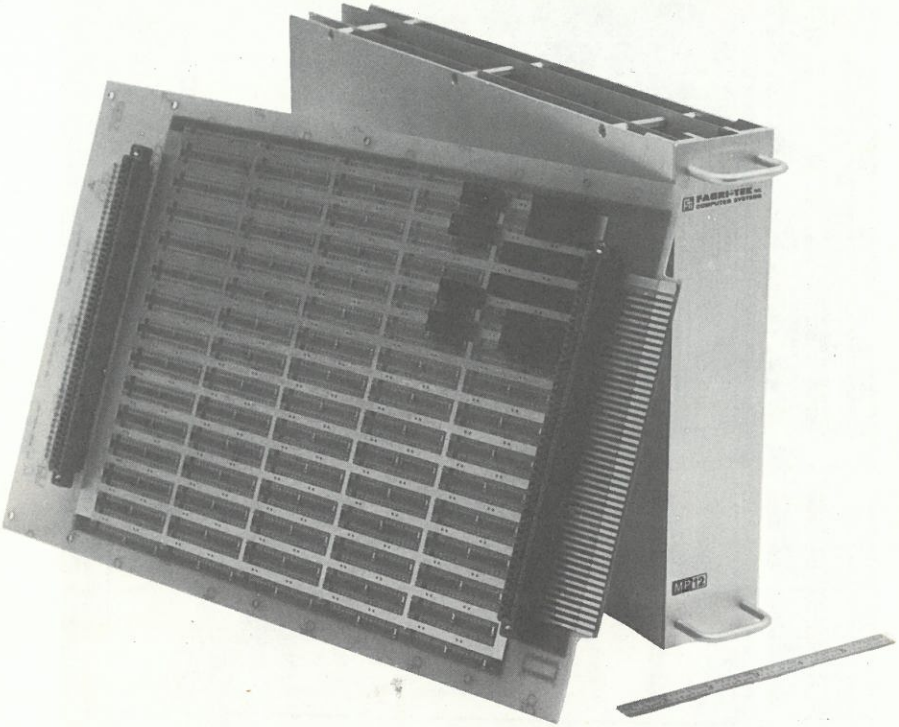


Figure 1-2
MP12 WIRE-WRAP ASSEMBLY

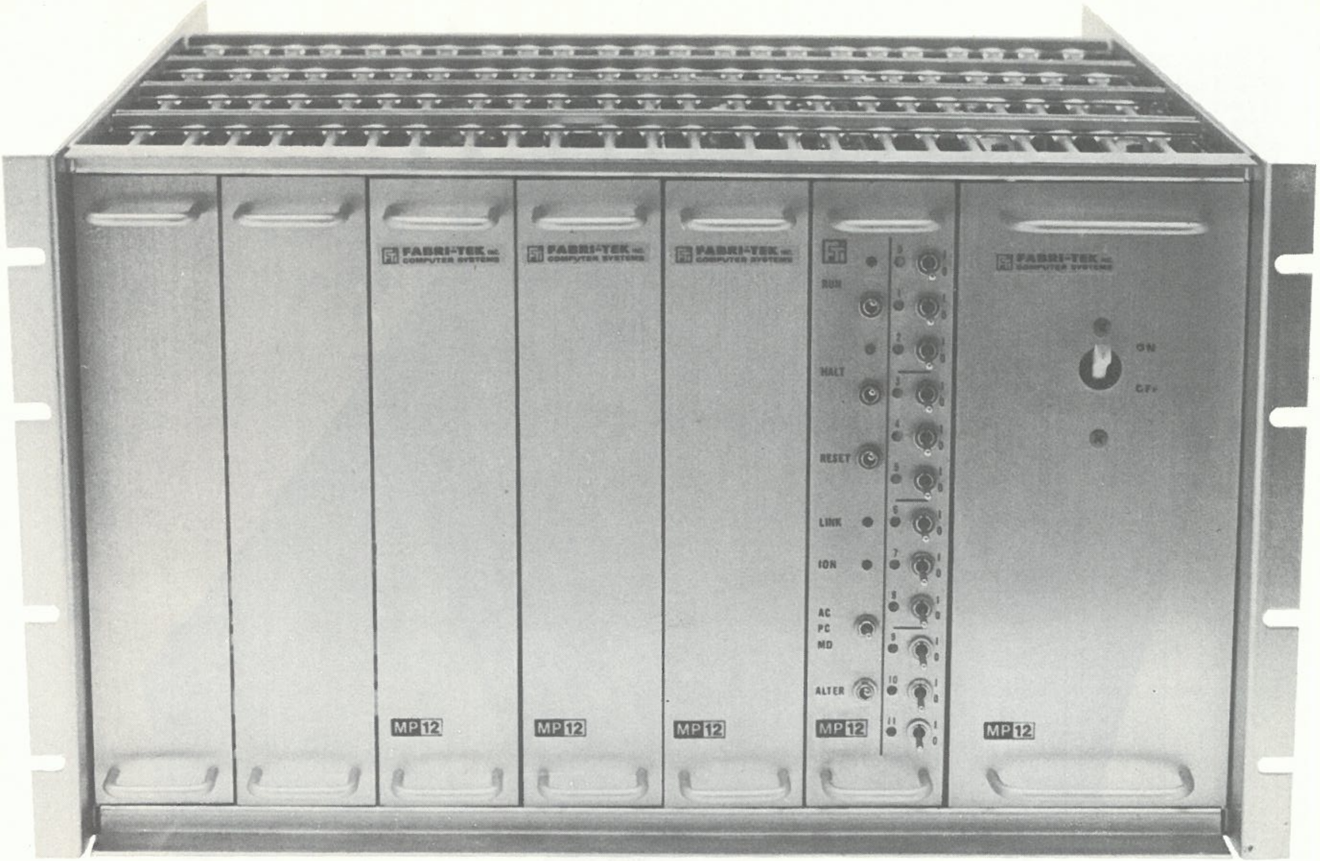


Figure 1-3
MP12 SYSTEM

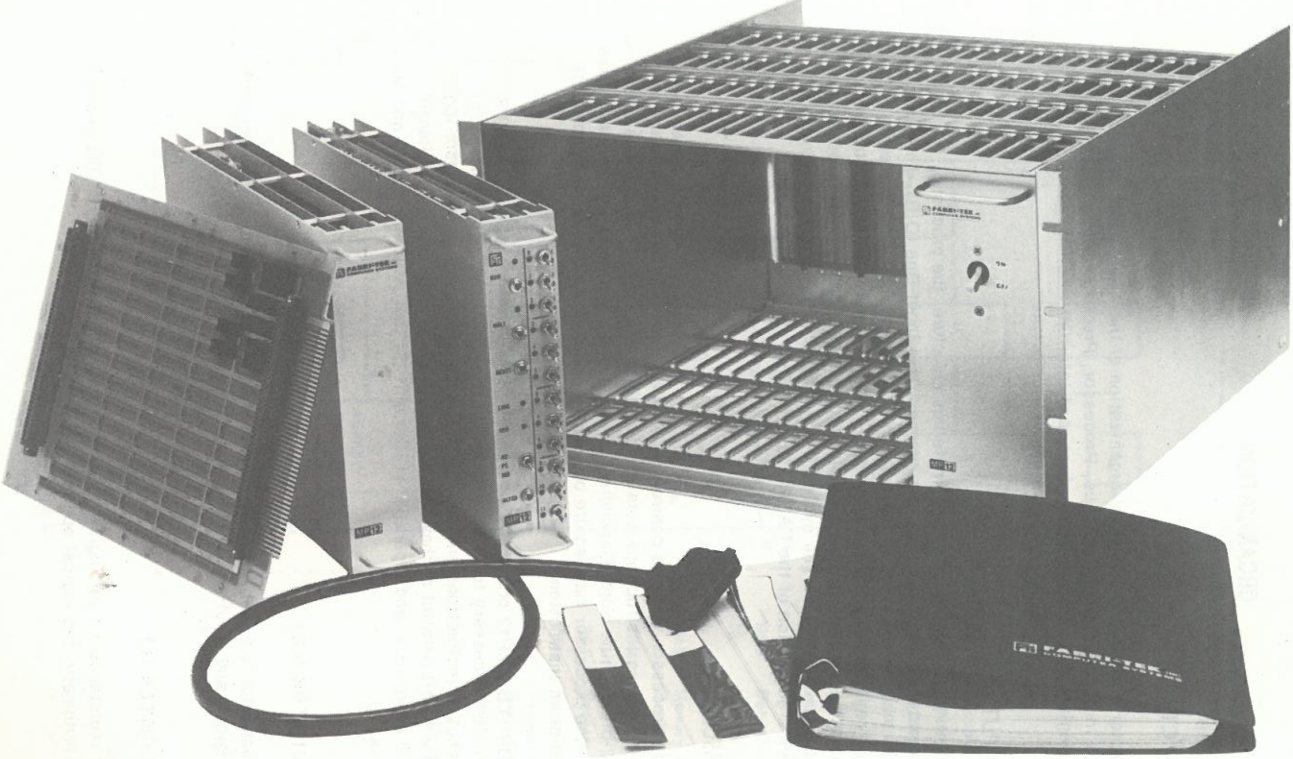


Figure 1-4
MP12 SYSTEM COMPONENTS

ORGANIZATION

INTRODUCTION

This chapter describes the internal organization of the FABRI-TEK MP12. Chapter topics include the Data Input Bus, Processor, Processor Registers, Memory, Memory Registers, Input-Output Interface, and Interrupt Facility.

DATA INPUT BUS (DIB)

The FABRI-TEK MP12 features a single bus structure; the processor, memory, and input-output channels all share a common Data Input Bus (DIB). The Data Input Bus is the mechanism whereby address information and data are transferred between the switch register (SR) and the processor, between the processor and the memory, between the memory and the input-output interface, and between the processor and the input-output interface. The system block diagram, Figure 1-5, illustrates the single bus organization of the FABRI-TEK MP12.

PROCESSOR

The MP12 processor performs control, input-output, arithmetic, and logical operations by executing instructions obtained from the memory. All instructions use a single 12-bit machine word. Depending on the number of separate memory accesses required, the processor may require one, two, or three memory cycles to complete execution of an instruction.

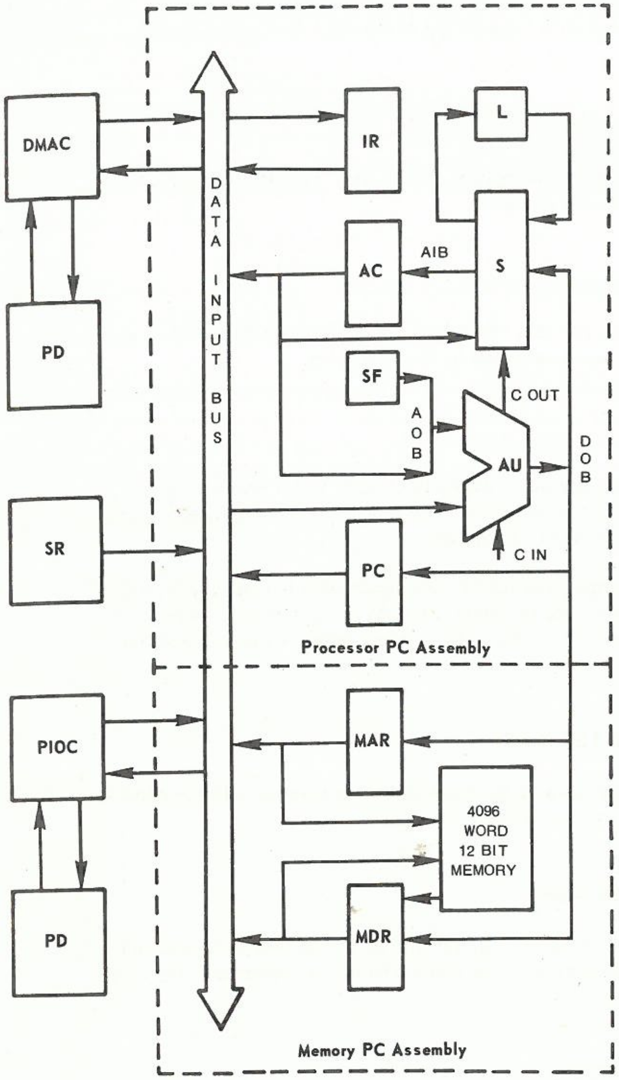
The FABRI-TEK MP12 processor incorporates four hardware registers and the logic circuitry necessary to perform control, arithmetic, and logical operations with respect to instructions and data stored in the memory. The processor logic includes a 12-bit parallel arithmetic unit (AU) that performs two's complement arithmetic operations, and a parallel shifter unit (S) that performs logical and shift operations.

ACCUMULATOR (AC)

The accumulator is a 12-bit register that functions as a holding register for arithmetic, logical, and input-output operations.

LINK REGISTER (L)

The link register is a 1-bit register that functions as an extension of the accumulator for arithmetic and logical operations.



AC	Accumulator
AIB	Accumulator In Bus
AOB	Accumulator Out Bus
AU	Arithmetic Unit
C IN	Carry In
C OUT	Carry Out
DMAC	Direct Memory Access Controller
DOB	Data Out Bus
IR	Instruction Register
L	Link
MAR	Memory Address Register
MDR	Memory Data Register
PC	Program Counter
PD	Peripheral Device
PIOC	Processor Input/Output Controller
S	Shifter
SF	Skip Flag
SR	Switch Register

Figure 1-5
MP12 BLOCK DIAGRAM

PROGRAM COUNTER (PC)

The program counter is a 12-bit register that holds the memory address of the next instruction to be processed. The execution of each instruction causes the program counter to be loaded with the address of the next instruction to be executed.

INSTRUCTION REGISTER (IR)

The instruction register is a 12-bit register that is used to hold the instruction currently being executed by the processor.

SKIP FLAG (SF)

The skip flag is a 1-bit register that represents a true/false skip condition with respect to the instruction being executed by the processor.

MEMORY

The FABRI-TEK MP12 memory has a capacity of 4096 12-bit words with a read/write cycle time of 1.5 microseconds. The memory is non-volatile; if power is removed, data stored in memory is not lost.

The processor and input-output interface communicate with the memory by way of the data input bus. Two hardware registers are used to hold memory address information and data received via the bus: the memory address register and the memory data register.

MEMORY ADDRESS REGISTER (MAR)

The memory address register is a 12-bit register that is used to hold the address of a data word to be read from, or written into, memory.

MEMORY DATA REGISTER (MDR)

The memory data register is a 12-bit register that holds the last data word read from, or written into, the memory location addressed by the contents of the memory address register.

INPUT-OUTPUT INTERFACE

The FABRI-TEK MP12 input-output interface incorporates two input-output channels, a processor input-output channel, and a direct memory access input-output channel.

PROCESSOR INPUT-OUTPUT (PIO) CHANNEL

The processor input-output channel enables data transfer between the accumulator and a selected input-output controller and device, as directed by the execution of a series of input-output transfer (IOT) instructions.

DIRECT MEMORY ACCESS (DMA) CHANNEL

The direct memory access channel functions as an independent data path to the memory. For a DMA transfer, control and address information are transmitted to a selected DMA controller via the processor input-output channel. The DMA controller then initiates and controls the transfer of data between the memory and a specified input-output device.

INTERRUPT FACILITY

The FABRI-TEK MP12 interrupt facility provides a processor interrupt when an input-output device is ready to receive or send data, or when a primary power failure is detected. The interrupt facility may be enabled or disabled using the interrupt on (ION) and interrupt off (IOF) instructions. If the interrupt facility is enabled when an interrupt occurs, the processor disables the interrupt facility, stores the contents of the program counter in memory location 0, and executes the instruction at memory location 1. If the interrupt system is disabled when an interrupt occurs, the interrupt is remembered by the processor and will remain active until cleared. When the ION instruction is executed, the interrupt facility is enabled after the instruction that follows the ION instruction is executed.

FUNCTIONAL DESCRIPTION

INTRODUCTION

This chapter describes the functional characteristics of the FABRI-TEK MP12. It discusses addressing techniques, including effective address generation and auto-index addressing; describes the formats used to represent various types of data internally; and concludes with a detailed description of the FABRI-TEK MP12 instruction set.

ADDRESS STRUCTURE

The FABRI-TEK MP12 possesses a contiguous memory address space of 4096 12-bit words. Any location within memory is accessable by way of a 12-bit address. This 12-bit address may be interpreted as: (1) a 5-bit page address field which specifies one of 32 pages of 128 words each, and (2) a 7-bit displacement address field which specifies one of 128 locations within the specified page.

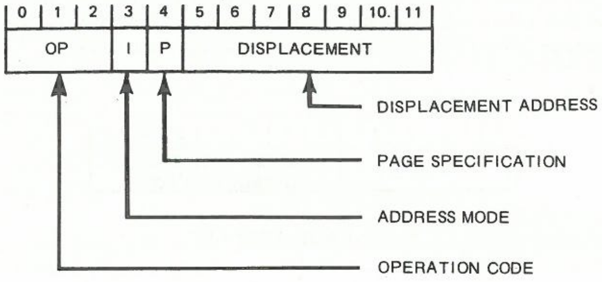
0	1	2	3	4	5	6	7	8	9	10	11
PAGE					DISPLACEMENT						

12-BIT ADDRESS FORMAT

Pages are assigned consecutive addresses in the range 0 to 31 (0_8-37_8) with page 0, the first 128 locations of memory, referred to as the base page. Displacement addresses range from 0 to 127 (0_8-177_8).

ADDRESSING TECHNIQUES

Each FABRI-TEK MP12 memory reference instruction dedicates three bits for operation code, one bit for address mode, one bit for page specification, and seven bits for displacement address.



MEMORY REFERENCE INSTRUCTION FORMAT

The effective address of a memory reference instruction operand is computed using the displacement address and the mode and page specification bits. The effective address is then used to access the required location in memory.

EFFECTIVE ADDRESS GENERATION

The effective address of a memory reference instruction operand is generated in the following manner:

1. A 12-bit primary address is generated from bit 4, the page specification bit, and the displacement address, bits 5 through 11.

2. A 12-bit effective address is generated from bit 3, the address mode bit, and the 12-bit primary address.

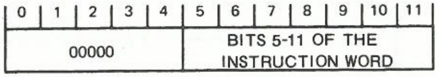
Primary Address

Bit 4, the page specification bit, controls the selection of the page address. When set to one, this bit indicates that the specified memory location lies within the current page; the page containing the instruction itself. In this case, a 12-bit primary address is obtained by combining bits 0 through 4 of the program counter with the displacement address of the instruction word as illustrated below.

0	1	2	3	4	5	6	7	8	9	10	11
BITS 0-4 OF THE PC					BITS 5-11 OF THE INSTRUCTION WORD						

CURRENT PAGE ADDRESS

A zero in the page specification bit position indicates that the specified memory location lies within the base page. In this case, as illustrated below, a 12-bit primary address is obtained directly from the displacement address.



BASE PAGE ADDRESS

Effective Address

Bit 3, the address mode bit, controls effective address generation. When zero, this bit indicates the direct address mode. When the direct address mode is specified, the primary address is interpreted as the effective operand address.

When set to one, the address mode bit indicates the indirect address mode. In this case, the contents of the primary address are interpreted as the effective operand address.

AUTO-INDEX ADDRESS

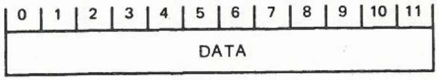
Base page locations 10_8 through 17_8 are referred to as auto-index locations. When any of these locations are indirectly addressed, the contents are read, incremented by one, rewritten back in the same location, and then used as an effective address.

DATA FORMATS

Data is logically represented in three internal binary formats: single word, double word, and floating point.

SINGLE WORD DATA

Single word data uses a single machine word to represent a 12-bit binary number in the following manner:

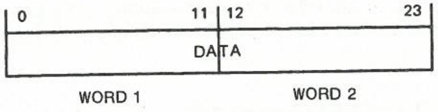


SINGLE WORD DATA

The data is right justified, in bit positions 0 through 11. Negative data is represented in two's complement form. The numerical range of signed data, which may be represented in this format, is $-2048_{10} \leq i \leq 2047_{10}$ ($4000_8 \leq i \leq 3777_8$). The numerical range of absolute (unsigned) data is $0_{10} \leq i \leq 4095_{10}$ ($0_8 \leq i \leq 7777_8$).

DOUBLE WORD DATA

Double word data, illustrated below, uses two consecutive machine words to represent a 24-bit signed binary number.

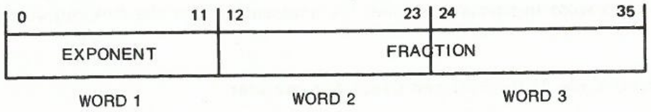


DOUBLE WORD DATA

The data is right justified, in bit positions 0 through 23. Negative data is represented in two's complement form. The numerical range of signed data, which may be represented in this format, is $-8388608_{10} \leq i \leq 8388607_{10}$ ($40000000_8 \leq i \leq 37777777_8$).

FLOATING POINT DATA

Floating point data is represented in three consecutive machine words as depicted below:



FLOATING POINT DATA

The 12-bit two's complement exponent occupies bits 0 through 11 of word one. The 24-bit two's complement fraction occupies bit positions 12 through 35 of words two and three. The radix point of the number is located immediately to the right of the high order fraction bit. Six-plus significant digits are representable in this format within an absolute numerical range of approximately $10^{\pm 616}$.

INSTRUCTION SET

The MP12 instruction set is organized into three instruction classes: Memory Reference, Operate, and Input-Output. The following sections describe each class by format and instruction functions.

MEMORY REFERENCE INSTRUCTIONS

The MP12 instruction set includes six basic memory reference instructions. Each instruction occupies a single 12-bit machine word and consists of a 3-bit operation code, a 2-bit address modification field, and a 7-bit displacement address as illustrated in Figure 1-6.

AND (Octal Code 0) Logical AND

The AND instruction results in a bit-by-bit Boolean AND operation between the contents of the accumulator and the memory data word addressed by the instruction. The result of the operation is retained in the accumulator. The contents of the addressed memory word are not altered and the link register is not affected.

TAD (Octal Code 1) Two's Complement Add

The TAD instruction performs a binary addition between the addressed data word and the contents of the accumulator. The result of the addition is retained in the accumulator. If a carry from the most significant bit of the accumulator occurs, the link register is complemented. The contents of the addressed memory word are not altered.

ISZ (Octal Code 2) Increment, Skip on Zero

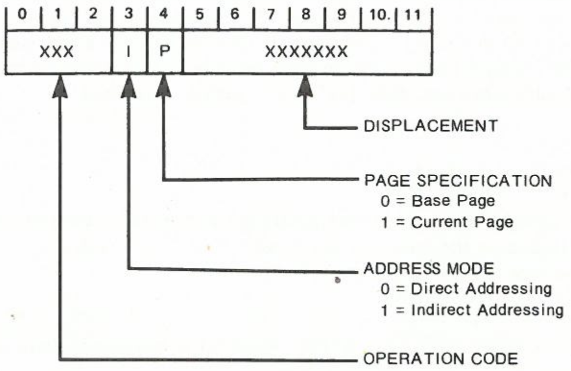
The ISZ instruction increments the contents of the addressed memory location and examines the result. If the result is zero, the next instruction in sequence is skipped and the following instruction is executed. If the result is non-zero, the next instruction in sequence is executed. In either case, the incremented result replaces the original data word in memory. Neither the accumulator nor the link register is affected.

DCA (Octal Code 3) Deposit and Clear Accumulator

The DCA instruction stores the contents of the accumulator in the addressed memory location, replacing the original contents of that location. The accumulator is then set to zero. The link register is not affected.

JMS (Octal Code 4) Jump to Subroutine

The JMS instruction causes the contents of the PC, the address of the JMS instruction plus one, to be stored in the addressed memory location, replacing the original contents. The PC is then set to the address of this location plus one; thus, the next instruction executed is the one following the location at which the PC was stored by the JMS instruction. Neither the accumulator nor the link register is affected.



Mnemonic	Op Code	Instruction
AND	000	Logical AND
TAD	001	Two's Complement Add
ISZ	010	Increment, skip on zero.
DCA	011	Deposit and clear accumulator
JMS	100	Jump to subroutine
JMP	101	Jump

Figure 1-6
MEMORY REFERENCE INSTRUCTION FORMAT

JMP (Octal Code 5) Jump

The JMP instruction causes the PC to be loaded with the address specified by the instruction word, resulting in a transfer of control to this location. The contents of the accumulator and link register are not affected.

OPERATE INSTRUCTIONS

The MP12 operate instructions enable the manipulation and testing of data located in the accumulator and link register. The operate instructions are separated into two functional groupings, referred to as Groups I and II. Each operate instruction occupies a single 12-bit machine word and consists of a 3-bit operation code, a group specification bit, and eight instruction specification bits.

Group I Operate Instructions

The Group I operate instructions manipulate the accumulator and link register. Figure 1-7 illustrates the format of the Group I operate instructions. The operation of each individual instruction is described below.

CLA CLEAR ACCUMULATOR. Each bit in the accumulator is set to zero.

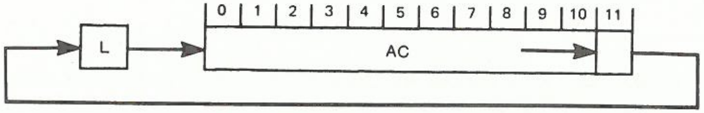
CLL CLEAR LINK REGISTER. The link register is set to zero.

CMA COMPLEMENT ACCUMULATOR. Each bit in the accumulator is complemented.

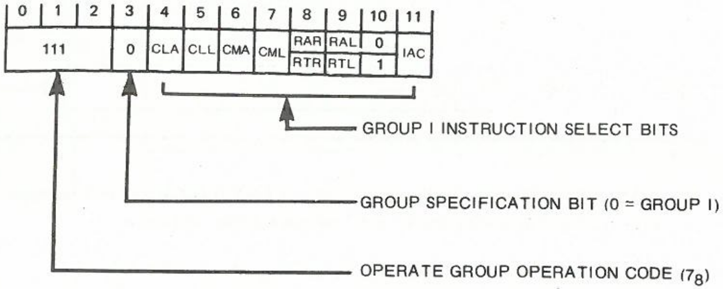
CML COMPLEMENT LINK. The link register is complemented.

IAC INCREMENT ACCUMULATOR. The accumulator is incremented by one. The link register is complemented in case a carry occurs from the most significant bit of the accumulator.

RAR ROTATE ACCUMULATOR AND LINK RIGHT. The link register and accumulator are rotated one position to the right as illustrated below:



RTR ROTATE ACCUMULATOR AND LINK RIGHT TWICE. The link register and accumulator are rotated two positions to the right.


0
1
2
3
4
5
6
7
8
9
10
11
111
0
CLA
CLL
CMA
CML
RAR
RTR
RAL
RTL
0
1
IAC
GROUP I INSTRUCTION SELECT BITS
GROUP SPECIFICATION BIT (0 = GROUP I)
OPERATE GROUP OPERATION CODE (7_8)

Mnemonic	Octal Code	Instruction
CLA	7200	Clear accumulator
CLL	7100	Clear link register
CMA	7040	Complement accumulator
CML	7020	Complement link
IAC	7001	Increment accumulator
RAR	7010	Rotate accumulator and link right
RTR	7012	Rotate accumulator and link right twice
RAL	7004	Rotate accumulator left
RTL	7006	Rotate accumulator and link left twice
NOP	7000	No operation

COMBINING GROUP I INSTRUCTIONS

1. Only one of the shift instructions RAR, RTR, RAL, RTL may appear in a combined instruction.

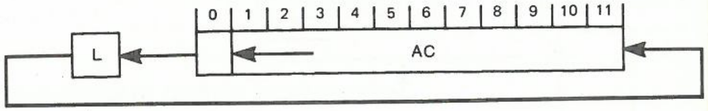
2. In a combined instruction, CLA and CLL, if specified, are executed first; CMA and CML, if specified, are executed next; IAC, if specified, is executed next; one of RAR, RTR, RAL, RTL, if specified, is executed last.

COMBINED INSTRUCTION EXAMPLES:

CMA IAC	Negate accumulator
CLA CMA	Load accumulator with -1
CLA CLL CMA RAL	Load accumulator with -2
CLA CML	Clear accumulator and complement link
CLA CLL CML IAC RTR	Load accumulator with -1024
CLA CLL CML RTL	Load accumulator with +2
NOP	No operation

Figure 1-7
GROUP I OPERATE INSTRUCTION FORMAT

RAL ROTATE ACCUMULATOR AND LINK LEFT. The link register and accumulator are rotated one position to the left as illustrated below:



RTL ROTATE ACCUMULATOR AND LINK LEFT TWICE. The link register and accumulator are rotated two positions to the left.

NOP NO OPERATION. No operation is performed.

Combining Group I Instructions

Group I operate instructions may be combined into a composite instruction subject to the following rules:

1. NOP is not combinable.

2. Only one of the shift instructions RAR, RTR, RAL, RTL may appear in a combined instruction.

3. The execution sequence for a composite Group I instruction is defined as follows:

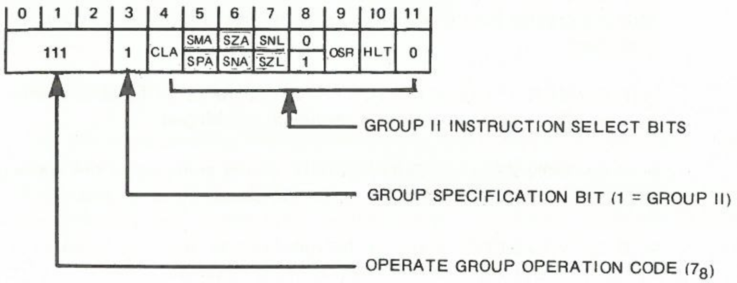
   a. CLA and CLL, if specified, are executed first.
   b. CMA and CML, if specified, are executed next.
   c. IAC, if specified, is executed next.
   d. One of RAR, RTR, RAL, or RTL, if specified, is executed last.

Group II Operate Instructions

Group II operate instructions provide test and skip capability based on the contents of the accumulator and link register. The format of the Group II operate instructions is illustrated in Figure 1-8. The following information describes the operation of each individual instruction.

CLA CLEAR ACCUMULATOR. Each bit in the accumulator is set to zero.

SMA SKIP ON MINUS ACCUMULATOR. If the contents of the accumulator are less than zero, the next instruction in sequence is skipped.


0
1
2
3
4
5
6
7
8
9
10
11
111
1
CLA
SMA
SZA
SNL
SPA
SNA
SZL
0
1
OSR
HLT
0
GROUP II INSTRUCTION SELECT BITS
GROUP SPECIFICATION BIT (1 = GROUP II)
OPERATE GROUP OPERATION CODE (7_8)

Mnemonic	Octal Code	Instruction
CLA	7600	Clear accumulator
SMA	7500	Skip on minus accumulator
SZA	7440	Skip on zero accumulator
SNL	7420	Skip on non-zero link
SPA	7510	Skip on positive accumulator
SNA	7450	Skip on non-zero accumulator
SZL	7430	Skip on zero link
OSR	7404	Inclusive "OR" switch register with accumulator
HLT	7402	Halt
SKP	7410	Unconditional skip

COMBINING GROUP II INSTRUCTIONS

1. For the skip group SMA, SZA, and SNL, a combination of these instructions will result in a skip only when at least one of the specified skip conditions is true.

2. For the skip group SPA, SNA, and SZL, a combination of these instructions will result in a skip only when all specified skip conditions are true.

3. Only members of one skip group may appear in a combined instruction.

4. SKP is combinable only with CLA, OSR, and HLT.

5. In a combined instruction, skip instructions are executed first; CLA, if specified, is executed next; OSR, if specified, is executed next; and HLT, if specified, is executed last.

COMBINED INSTRUCTION EXAMPLES:

SMA SZA – Skip if accumulator is negative or zero
CLA OSR – Transfer switch register into accumulator
SZA CLA – Skip if accumulator is zero and clear accumulator
CLA HLT – Clear accumulator and halt

Figure 1-8
GROUP II OPERATE INSTRUCTION FORMAT

SPA SKIP ON POSITIVE ACCUMULATOR. If the contents of the accumulator are greater than or equal to zero, the next instruction in sequence is skipped.

SZA SKIP ON ZERO ACCUMULATOR. If the contents of the accumulator are zero, the next instruction in sequence is skipped.

SNA SKIP ON NON-ZERO ACCUMULATOR. If the contents of the accumulator are non-zero, the next instruction in sequence is skipped.

SNL SKIP ON NON-ZERO LINK. If the link register does not contain a zero, the next instruction in sequence is skipped.

SZL SKIP ON ZERO LINK. If the link register contains a zero, the next instruction in sequence is skipped.

SKP UNCONDITIONAL SKIP. The next instruction in sequence is skipped.

OSR INCLUSIVE "OR" SWITCH REGISTER WITH ACCUMULATOR. The console switch register is inclusive OR'ed with the contents of the accumulator and the result is retained in the accumulator.

HLT HALT. Causes the computer to halt at the conclusion of the current machine cycle.

Combining Group II Instructions

Group II operate instructions, subject to the following rules, may be combined into a composite instruction.

1. Skip group instructions SPA, SNA, and SZL. A combination of these instructions will result in a skip only when all specified skip conditions are true.
2. Skip group instructions SMA, SZA, and SNL. A combination of these instructions will result in a skip only when at least one of the specified skip conditions is true.
3. Only members of one skip group may appear in a combined instruction.
4. SKP is combinable only with CLA OSR, and HLT.
5. The execution sequence for a composite Group II operate instruction is defined as follows:
   a. Skip instructions are executed first.
   b. CLA, if specified, is executed next.

   c. OSR, if specified, is executed next.
   d. HLT, if specified, is executed last.

INPUT-OUTPUT INSTRUCTIONS

The MP12 input-output instructions enable data transfer between the computer and peripheral units by way of the accumulator. Each input-output instruction consists of a 3-bit operation code, a 6-bit device address, and a 3-bit function code. Input-output instructions are described in the Input-Output Interface information.

OPERATION

INTRODUCTION

This chapter describes the operation of the FABRI-TEK MP12. It discusses the layout of the operating console, console controls and indicators and the functions which may be performed at the operating console.

OPERATING CONSOLE

The operating console, mounted on the front of the processor enclosure, contains all controls and indicators necessary for the operation of the processor. The console frame measures 9.5 in. by 2.125 in. Figure 1-9 depicts the layout of the FABRI-TEK MP12 operating console.

CONTROLS AND INDICATORS

The following information describes the controls and indicators on the FABRI-TEK MP12 operating console.

SWITCH REGISTER — The Switch Register consists of twelve data entry switches that are used to manually alter the contents of the accumulator, program counter, or memory data register. The switch register can also be read under program control.

DISPLAY REGISTER — The Display Register consists of twelve indicators that display the contents of the register selected by the Display switch.

RUN SWITCH — The Run switch, when toggled, causes the processor to commence instruction execution beginning at the address contained in the program counter.

HALT SWITCH — The Halt switch, when toggled, causes the processor to stop instruction execution; the program counter contains the address of the next instruction to be executed. When the processor is halted, toggling the Halt switch causes the instruction located at the address contained in the program counter to be executed.

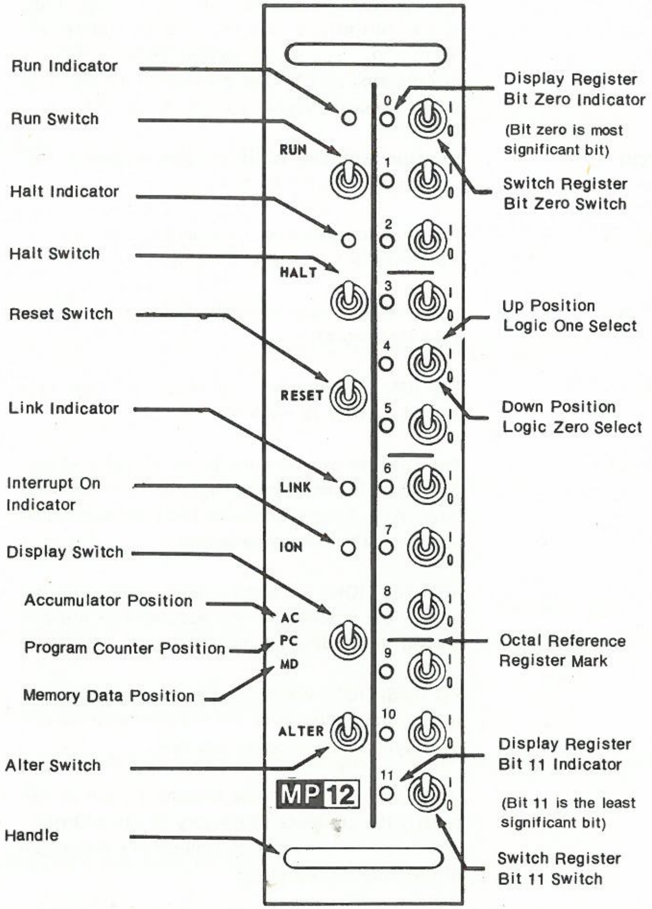

Run Indicator
Run Switch
Halt Indicator
Halt Switch
Reset Switch
Link Indicator
Interrupt On Indicator
Display Switch
Accumulator Position
Program Counter Position
Memory Data Position
Alter Switch
Handle
RUN
HALT
RESET
LINK
ION
AC
PC
MD
ALTER
MP12
0
1
2
3
4
5
6
7
8
9
10
11
1
0
Display Register Bit Zero Indicator
(Bit zero is most significant bit)
Switch Register Bit Zero Switch
Up Position Logic One Select
Down Position Logic Zero Select
Octal Reference Register Mark
Display Register Bit 11 Indicator
(Bit 11 is the least significant bit)
Switch Register Bit 11 Switch

Figure 1-9
MP12 OPERATING CONSOLE

RESET SWITCH

The Reset switch, when toggled, generates a master reset condition. The processor is halted, all internal registers are set to zero, the interrupt facility is disabled, the input-output interface is initialized, and the program counter is set to 200_8. The Reset switch also functions as an indicator test in that all indicators are illuminated when the Reset switch is toggled.

RUN INDICATOR

The Run indicator is illuminated whenever the processor is in the RUN mode.

HALT INDICATOR

The Halt indicator is illuminated whenever the processor is in the HALT mode.

LINK INDICATOR

The Link indicator displays the content of the 1-bit link register.

ION INDICATOR

The ION indicator is illuminated when the interrupt facility is enabled.

DISPLAY SWITCH

The Display switch selects the register to be displayed in the display register. The accumulator (AC), program counter (PC) or memory data register (MD) can be selected.

AC POSITION. When the display switch is set to AC, the contents of the accumulator are displayed in the display register.

PC POSITION. When the display switch is set to PC, the contents of the program counter are displayed in the display register.

MD POSITION. When the display switch is set to MD, the contents of memory at the address contained in the program counter are displayed in the display register.

ALTER SWITCH

The Alter switch, when toggled, causes the contents of the switch register to be copied into the register selected by the display switch, or the memory location contained in the program counter if the display switch is set to MD.

REMOTE CONTROL SIGNALS

Signal lines for the RUN, HALT, and RESET functions as well as LOAD and POWER-ON are present on the MP12 CPU connector. These signals permit the processor to be operated from a remote control panel located up to 50 feet from the processor. The function of each remote control signal is indicated below.

REMOTE-RESET (Dual-Rail)

Asserting the REMOTE-RESET signal initializes the processor and all input-output controllers.

REMOTE-HALT (Dual-Rail)

Asserting the REMOTE-HALT signal halts the processor.

REMOTE-RUN (Dual-Rail)

Asserting the REMOTE-RUN signal starts the processor.

REMOTE-LOAD (Dual-Rail)

Asserting the REMOTE-LOAD signal initializes the processor and all input-output controllers, sets the program counter to 7777_8, and starts the processor.

REMOTE-HALT-INDICATOR

The REMOTE-HALT-INDICATOR signal is asserted when the processor is halted.

REMOTE-RUN-INDICATOR

The REMOTE-RUN-INDICATOR signal is asserted when the processor is running.

REMOTE-POWER-ON

Grounding the REMOTE-POWER-ON signal turns on the +5VDC power supply.

INPUT-OUTPUT INTERFACE

INTRODUCTION

This chapter describes the FABRI-TEK MP12 input-output interface. It discusses the input-output bus, input-output controllers, input-output instructions, and the MP12 interrupt facility.

INPUT-OUTPUT BUS

The MP12 input-output bus (IOB) consists of the twelve bi-directional data lines comprising the DIB together with twelve additional control lines. Seven of the twelve control lines together with the twelve data lines constitute the processor input-output (PIO) channel. The remaining five control lines together with the common data lines comprise the direct memory access (DMA) channel. Figure 1-10 illustrates the arrangement of the IOB with respect to the MP12 processor, input-output device controllers, and extension memory modules. All input-output bus signal lines are terminated inside the processor with a 220 ohm resistor to +5 volts DC and a 330 ohm resistor to ground. If the bus is extended outside the standard chassis a second terminator of equal value should be used on the far end of the bus. Recommended bus drivers and receivers are SN74H38 and SN7404 respectively.

PROCESSOR INPUT-OUTPUT (PIO) CHANNEL

The processor input-output channel is used to communicate with low-speed, character-oriented devices which are asynchronous in nature. Each item of data is transferred to or from an addressed device, via the accumulator, by executing a separate IOT instruction for each transfer. IOT instructions, in addition to transferring data, are also used to test the status of a device and to initiate input or output operations. The PIO channel is capable of transfer rates of up to 66,000 words-per-second.

DIRECT MEMORY ACCESS (DMA) CHANNEL

The direct memory access input-output channel is used to communicate with high-speed, record-oriented devices such as disk units and magnetic tape equipment. DMA input-output requests require control and address information to be transmitted to a selected DMA controller via the PIO channel; a series of IOT commands are executed to consumate this transfer of information. Once started, a DMA input-output operation proceeds to completion independently of the processor. The DMA channel is capable of sustaining burst transfer rates of up to 666,666 words-per-second.

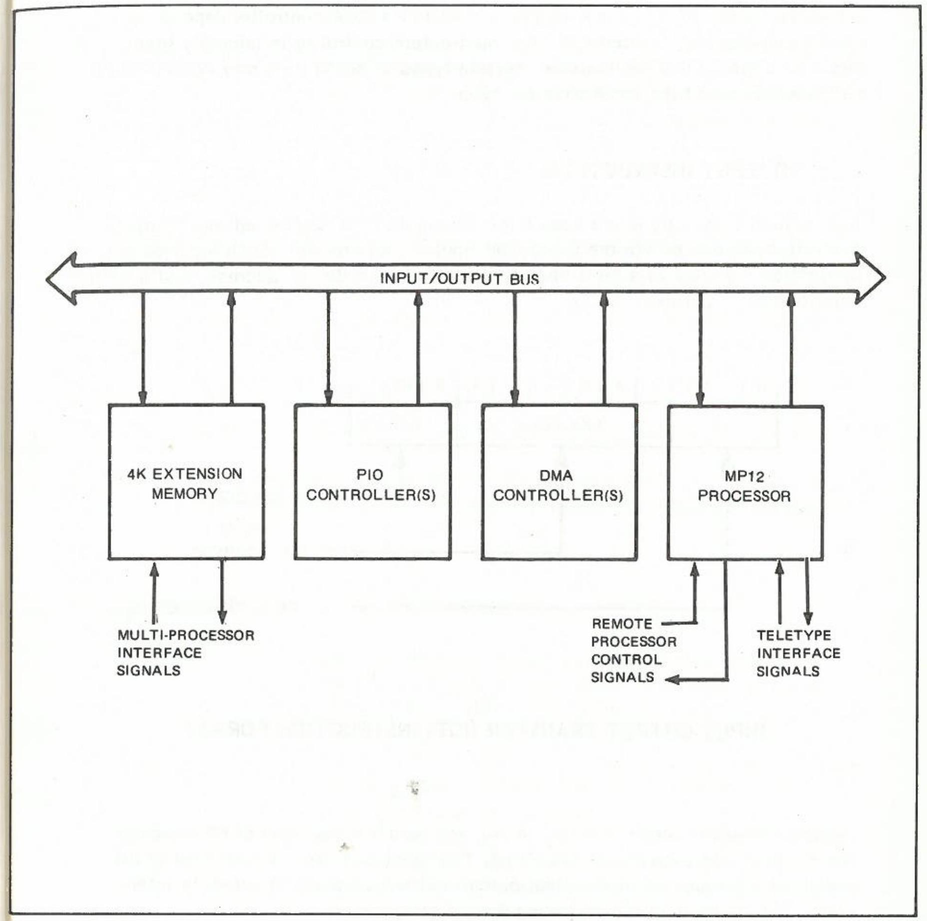

INPUT/OUTPUT BUS
4K EXTENSION MEMORY
PIO CONTROLLER(S)
DMA CONTROLLER(S)
MP12 PROCESSOR
MULTI-PROCESSOR INTERFACE SIGNALS
REMOTE PROCESSOR CONTROL SIGNALS
TELETYPE INTERFACE SIGNALS

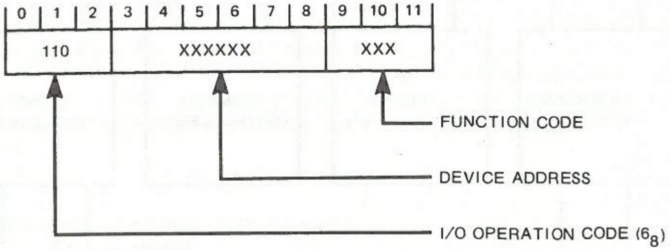
Figure 1-10
INPUT/OUTPUT BUS

INPUT-OUTPUT DEVICE CONTROLLERS

Input-output controllers consist of the necessary logic circuitry required to interconnect one or more peripheral devices with the input-output interface. Each input-output controller functions as either a PIO or a DMA controller depending upon which channel is interfaced. An input-output controller is normally identified with a single device; however, certain types of controllers may accommodate multiple devices of the same physical type.

INPUT-OUTPUT INSTRUCTIONS

Input-output instructions are used to communicate with a selected input-output controller and device via the processor input-output channel. Each input-output instruction consists of a 3-bit operation code, a 6-bit device address, and a 3-bit function code as illustrated.



INPUT-OUTPUT TRANSFER (IOT) INSTRUCTION FORMAT

Device addresses range from 01_8 to 76_8 and permit a maximum of 62 separate input-output addresses to be specified. The function code, as specified in bit positions 9 through 11 of the input-output instruction word, is normally interpreted with respect to the following device states:

STATE **INTERPRETATION**

AVAILABLE A device is AVAILABLE provided that it is powered, on-line, properly enabled, and otherwise capable of operation.

READY A device is READY provided that it is available and not in the process of performing a previously ordered input or ouput operation.

DONE A device is DONE if it has generated an interrupt request to the processor, indicating that a previously ordered input or output operation has been completed. For input devices, the state DONE implies that data is present in the device data buffer. A DONE device is always READY, but not conversely.

For the standard MP12 teletype and high speed paper tape reader/punch interfaces, the operations performed by each input-output function are described below. The function code is decoded in the manner shown in Figure 1-11. Note that bits 9 and 10 of the function code result in the clearing of an interrupt request as follows:

Bit 9: When an input device is addressed and bit 9 of the function code is set to one, the interrupt request is cleared if the device is DONE.

Bit 10: When an input or output device is addressed and bit 10 of the function code is set to one, the interrupt request is cleared if the device is DONE.

FUNCTION CODE **INTERPRETATION**

1_8 SKIP IF READY. If the addressed device is READY, the next instruction in sequence is skipped. If the device is not READY, no skip occurs.

2_8 START OPERATION. If the addressed device is READY, the next input or output operation is started. If the device is not READY, no operation is performed.

3_8 SKIP IF READY AND START OPERATION. If the addressed device is READY, the next instruction in sequence is skipped and the next input or output operation is started. If the device is not READY, no skip occurs and no operation is performed.

4_8 TRANSFER DATA. For input devices, if the addressed device is DONE, the contents of the device data buffer are inclusive OR'ed with the accumulator and the result retained in the accumulator. If the device is not DONE, no operation is performed.

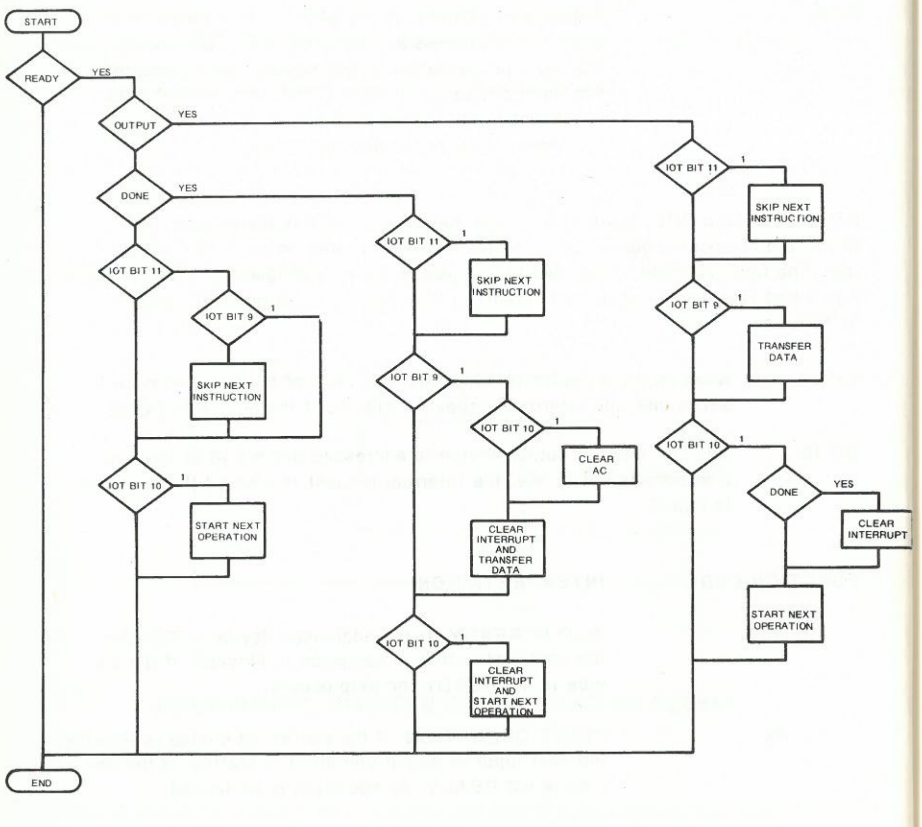


Figure 1-11
IOT FUNCTION DECODE

FUNCTION CODE **INTERPRETATION**

For output devices, if the device is READY, the contents of the accumulator are transferred to the device data buffer. If the device is not READY, no operation is performed.

5_8 SKIP IF DEVICE READY AND TRANSFER DATA. For input devices, if the addressed device is DONE, the next instruction in sequence is skipped, the contents of the device buffer are inclusive OR'ed with the accumulator, and the result is retained in the accumulator. If the device is not DONE, no skip occurs and no operation is performed.

For output devices, if the addressed device is READY, the next instruction in sequence is skipped and the contents of the accumulator are transferred to the device data buffer. If the device is not READY, no skip occurs and no operation is performed.

6_8 TRANSFER DATA AND START NEXT OPERATION. For input devices, if the addressed device is DONE, the accumulator is cleared, the contents of the device data buffer are inclusive OR'ed with the accumulator, and the result is retained in the accumulator. If the device is either READY or DONE, the next input operation is started. If the device is not READY, no operation is performed.

For output devices, if the addressed device is READY, the contents of the accumulator are transferred to the device data buffer and the next output operation is started. If the device is not READY, no operation is performed.

7_8 SKIP IF DEVICE READY, TRANSFER DATA, AND START NEXT OPERATION. For input devices, if the addressed device is DONE, the next instruction in sequence is skipped, the accumulator is cleared, the contents of the device data buffer are inclusive OR'ed with the accumulator, and the result is retained in the accumulator. If the device is READY and not DONE, no skip occurs but the next input operation is started. If the device is not READY, no skip occurs and no operation is performed.

For output devices, if the addressed device is READY, the next instruction in sequence is skipped, the contents of the accumulator are transferred to the device data buffer, and the next output operation is started. If the device is not READY, no skip occurs and no operation is performed.

The following examples illustrate the use of the FABRI-TEK MP12 input-output (IOT) instructions, and do not utilize the interrupt facility. The examples are formulated in symbolic assembly language notation as described in the Assembly Language information.

Example 1: Print the letter "A" on the teletype (Teletype printer/punch address = 4).

```
-------------------------------------------------
.  IØT   041  /SKIP IF PRINTER READY            .
.  JMP   .-1  /NØT READY, TRY AGAIN             .
.  TAD   LET  /READY, FETCH THE LETTER "A"      .
.  IØT   046  /TRANSFER TØ PRINTER AND          .
.             /START PRINT ØPERATIØN            .
.  ---                                          .
.  LET,0301   /ASCII "A"                        .
-------------------------------------------------
```

Example 2: The same operation performed in Example 1 may be performed with a single input-output instruction using the function code 7 as follows:

```
-------------------------------------------------
.  TAD   LET  /FETCH THE LETTER "A"             .
.  IØT   047  /SKIP IF READY AND TRANSFER       .
.  JMP   .-1  /NØT READY, TRY AGAIN             .
.  ---        /PRINT ØPERATIØN STARTED          .
.  ---                                          .
.  LET,0301   /ASCII "A"                        .
-------------------------------------------------
```

Example 3: Read a character from the teletype keyboard (Keyboard/reader address = 3).

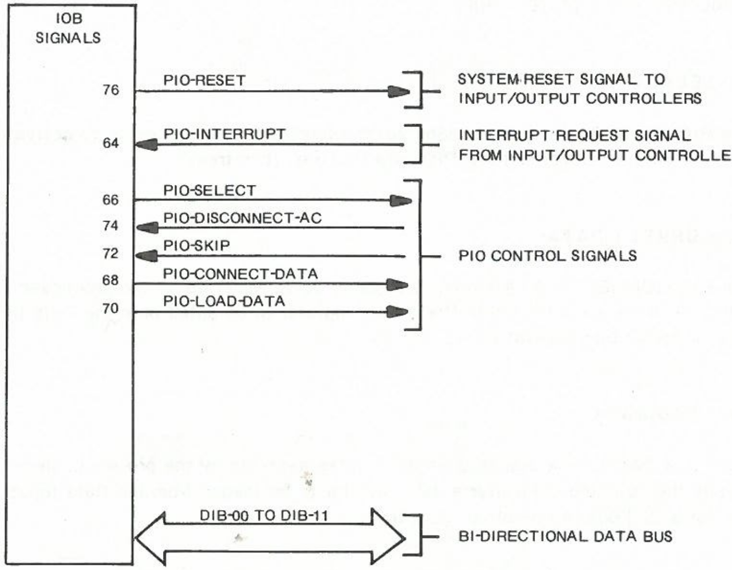
```
-------------------------------------------------
.  IØT   032  /START KEYBØARD                   .
.                                               .
.                                               .
.  IØT   031  /SKIP IF KEYBØARD READY           .
.  JMP   .-1  /NØT READY, TRY AGAIN             .
.  IØT   036  /READY, READ CHARACTER AND        .
.             /START NEXT READ ØPERATIØN        .
-------------------------------------------------
```

Example 4: The same operation performed in Example 3 may be performed with a single input-output instruction using the function code 7 as follows:

```
-------------------------------------------------
.  IØT   037  /READ WHEN READY AND SKIP         .
.  JMP   .-1  /NØT READY, TRY AGAIN             .
.  ---        /CHARACTER READ AND THE NEXT      .
.             /INPUT ØPERATIØN STARTED          .
-------------------------------------------------
```

PROCESSOR INPUT-OUTPUT OPERATION

Processor input-output operations are supported by seven control signal lines and the twelve DIB signal lines. These nineteen signal lines consitute the MP12 PIO channel as depicted below.



PIO SIGNALS

PROCESSOR INPUT-OUTPUT SIGNALS

The following information describes the function of each PIO signal.

DIB-00 to DIB-11, DATA INPUT BUS

Twelve-line bidirectional bus that connects the processor and input-output controllers. All address, data, and control words exchanged between the processor and input-output controllers are transmitted by way of the DIB. (L = true)

PIO-RESET

The PIO-RESET signal is asserted by the processor during power-up/down, or when the console reset switch is operated, and serves as a master reset signal to all peripheral controllers. (L = true)

PIO-INTERRUPT

The PIO-INTERRUPT signal is asserted by an input-output controller to request a processor interrupt. (L = true)

PIO-SELECT

The PIO-SELECT signal is an 83ns pulse asserted by the processor to activate an input-output controller for a PIO data transfer. (L = true)

PIO-CONNECT-DATA

The PIO-CONNECT-DATA signal is a 500ns pulse asserted by the processer that causes the selected controller's data register to be gated onto the Data Input Bus for a PIO input operation. (L = true)

PIO-LOAD-DATA

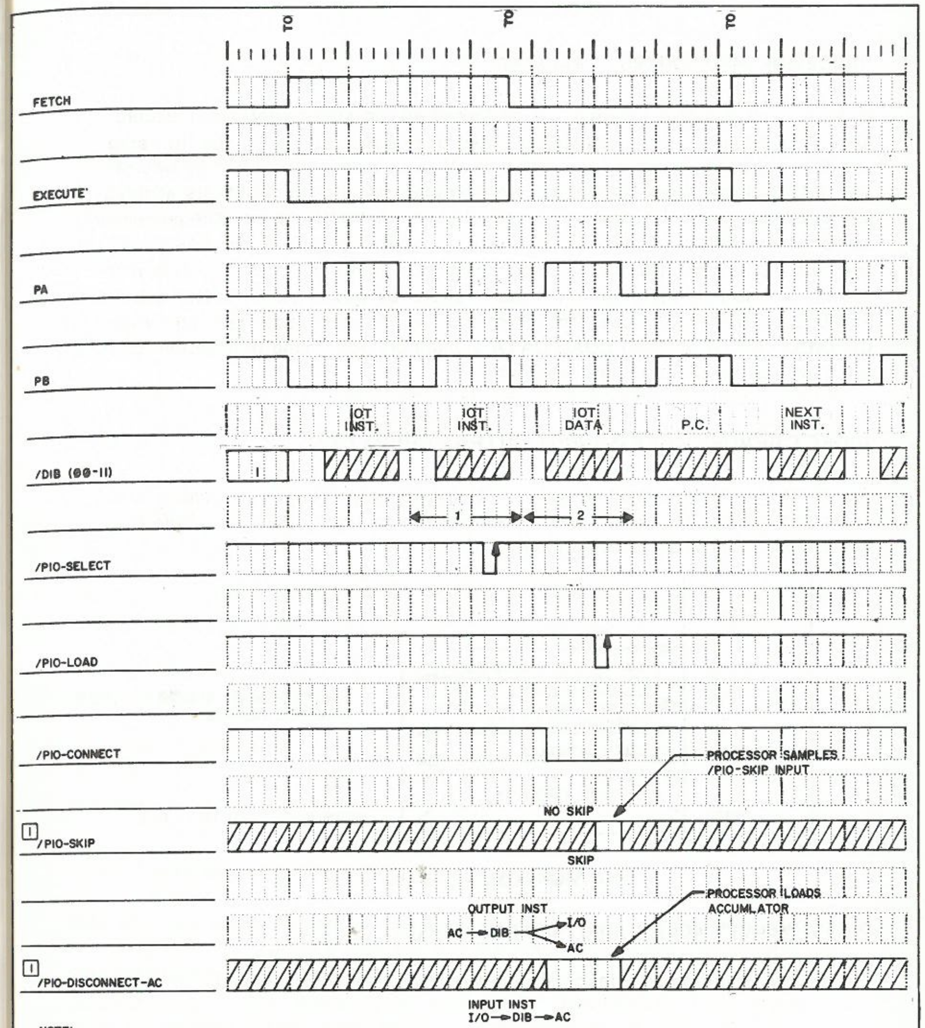
The PIO-LOAD-DATA signal is an 83ns pulse asserted by the processor that causes the selected controller's data register to be loaded from the Data Input Bus for a PIO output operation. (L = true)

PIO-SKIP

The PIO-SKIP signal is asserted by a selected input-output controller to cause the instruction following an IOT instruction to be skipped. (L = true)

PIO-DISCONNECT-AC

The PIO-DISCONNECT-AC signal is asserted by a selected input-output controller to cause the accumulator not to be gated onto the Data Input Bus during a PIO



NOTE:
1 SIGNALS GENERATED BY PIO DEVICE CONTROLLER.
2. EACH DIVISION REPRESENTS 83 NS.

Figure 1-12
PIO TIMING DIAGRAM

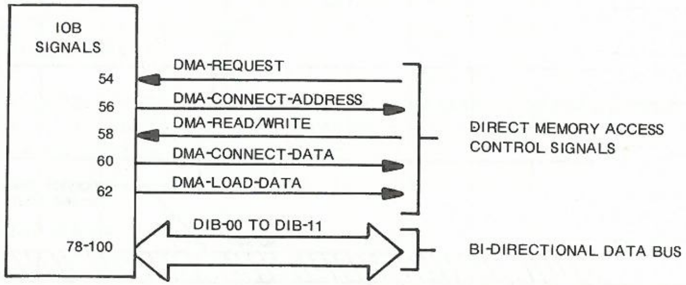
data transfer operation. This signal allows a controller to control the direction of a PIO data transfer. (L = true)

PROCESSOR INPUT-OUTPUT TRANSFERS

Processor input-output transfers are performed as two-step operations in accordance with the timing requirements indicated in Figure 1-12. During the first step the IOT instruction word is transmitted to all input-output controllers by way of the 12 DIB signal lines. Each controller contains circuitry to decode its address and store the control information contained in the instruction word. The processor selects the addressed controller and loads its control register by asserting the PIO-SELECT signal. During the second step the controller transmits status indications and selects the direction of data transfer by way of the PIO-SKIP and PIO-DISCONNECT-AC signal lines, respectively. The processor generated PIO-CONNECT-DATA and PIO-LOAD-DATA signals provide timing information for input and output data transfers, respectively.

DIRECT MEMORY ACCESS INPUT-OUTPUT OPERATION

Direct memory access input-output operations are supported by five control signal lines and twelve DIB signal lines. These seventeen signal lines constitute the MP12 DMA channel as depicted below.



DMA SIGNALS

DIRECT MEMORY ACCESS INPUT-OUTPUT SIGNALS

The following information describes the function of each DMA signal.

DIB-00 to DIB-11, DATA INPUT BUS

Twelve-line bidirectional bus that connects the processor and input-output controllers. All address, data, and control words exchanged between the processor and input-output controllers are transmitted by way of the Data Input Bus. (L = true)

DMA-REQUEST

The DMA-REQUEST signal is asserted by an input-output controller to request a DMA data transfer. (L = true)

DMA-CONNECT-ADDRESS

The DMA-CONNECT-ADDRESS signal is a 500ns pulse asserted by the processor in response to a DMA request to gate the requesting controller's address register onto the Data Input Bus. (L = true)

DMA-READ/WRITE

The DMA-READ/WRITE signal is asserted by the requesting input-output controller in response to the DMA-connect address signal to control the direction of the DMA data transfer. (L = true)

DMA-CONNECT-DATA

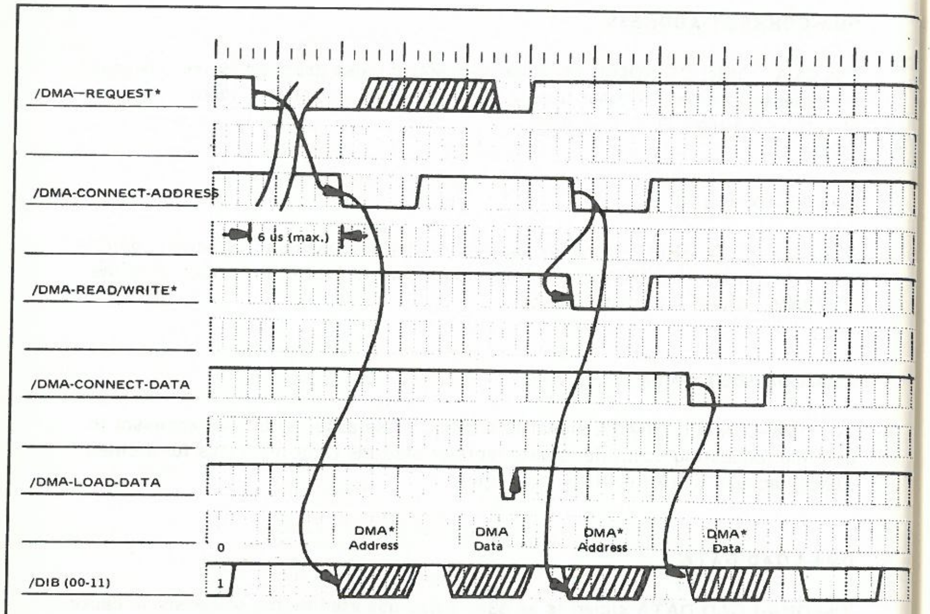
The DMA-CONNECT-DATA signal is a 500ns pulse asserted by the processor to gate the requesting controller's data register onto the Data Input Bus for a direct memory access write operation. (L = true)

DMA-LOAD-DATA

The DMA-LOAD-DATA signal is an 83ns pulse asserted by the processor to cause the requesting controller's data register to be loaded from the Data Input Bus for a direct memory access read operation. (L = true)

DIRECT MEMORY ACCESS INPUT-OUTPUT TRANSFERS

DMA transfers are performed in accordance with the timing requirements indicated in Figure 1-13. To initiate a DMA transfer, an input-output controller asserts the DMA-REQUEST signal. The channel responds by asserting the DMA-CONNECT-ADDRESS signal. The controller then removes the DMA-REQUEST signal and gates the address of the memory location to be accessed onto the DIB. If the operation to be performed is a write operation, the controller also asserts the DMA-READ/WRITE signal in response to the DMA-CONNECT-ADDRESS signal. If the operation is a read, the addressed data word is read from memory and gated onto the DIB. The channel then asserts the DMA-LOAD-DATA signal to cause the data on the DIB to be loaded into the controller's data register. If the operation is a write, the channel asserts the DMA-CONNECT-DATA signal causing the controller's data register to be gated onto the DIB and written into the specified memory location.



NOTE: Each division represents 83 ns.

*Device controller is signal source.

Figure 1-13
DMA TIMING DIAGRAM

INTERRUPT FACILITY

The FABRI-TEK MP12 interrupt facility provides a processor interrupt when an input-output device is ready to send or receive data, or a power failure is detected. If the interrupt facility is enabled when an interrupt occurs, the processor disables the interrupt facility, stores the contents of the program counter in location 0, and executes location 1. The following instructions are used to control the FABRI-TEK MP12 interrupt facility.

MNEMONIC	INSTRUCTION
ION (Octal Code 6001)	TURN INTERRUPT SYSTEM ON. Enables the interrupt system after a one instruction delay.
IOF (Octal Code 6002)	TURN INTERRUPT SYSTEM OFF. Disables the interrupt system. No interrupts can occur until the interrupt system is enabled.
SPL (Octal Code 6004)	SKIP ON POWER LOW. The next instruction in sequence is skipped if power is low. This instruction is used to identify interrupts originated by the automatic power fail detection circuitry in the optional FABRI-TEK MP12 power supply module.
CON (Octal Code 6774)	TURN LINE FREQUENCY CLOCK ON. Enables the line frequency clock in the optional power supply module. When enabled, the clock will generate a processor interrupt each 16 2/3 milliseconds until disabled.
COF (Octal Code 6772)	TURN LINE FREQUENCY CLOCK OFF. Disables the line frequency clock.
SCD (Octal Code 6771)	SKIP ON CLOCK DONE AND CLEAR INTERRUPT. The next instruction in sequence is skipped if the line frequency clock has generated an interrupt request.

ASSEMBLY LANGUAGE

INTRODUCTION

This chapter describes the FABRI-TEK MP12 assembly language. It discusses the characteristics of symbolic assembly language programs and describes the mechanism by which such programs are translated into executable machine programs. Chapter topics include the Assembler, the Assembly Language Character Set, Statements, Expressions, Machine Instructions, Error Processing, and the Assembly Listing.

ASSEMBLER

Programs written in FABRI-TEK MP12 assembly language are translated by an assembler program into executable machine programs. The assembly process is basically one of converting symbolic instructions into binary machine instructions, generating data, assigning storage locations for machine instructions and data, and performing auxiliary functions necessary to produce an executable machine program.

An assembly language program consists of a series of symbolic statements which are normally written on coding forms and later transcribed to paper tape for input to the assembler. The assembler reads the source tape containing the symbolic program and produces a printed listing which contains the machine code resulting from each statement, and a punched paper tape, or object tape, containing the machine program. The object tape may then be loaded into the computer and the program executed. Three separate passes or readings of the source program are required to complete the assembly process. The function of each assembly pass is described below:

Pass 1 – The assembler reads the source tape and constructs an internal symbol table which records the value of each symbol in the program.

Pass 2 – The assembler punches an object tape containing the assembled machine program.

Pass 3 – The assembler prints a listing of the assembled machine program.

ASSEMBLY LANGUAGE CHARACTER SET

Program statements are constructed with characters taken from the following character set:

Letters: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Digits: 0 1 2 3 4 5 6 7 8 9

Special Characters: space ! '' # % & ' ( ) + , - : < > ? @ [/] ↑ ←

All other characters, except for the following, are ignored.

Slash (/): Indicates the start of a comment string.

Carriage-Return: Indicates the end of a symbolic statement.

Semicolon (;): Same as carriage-return unless appearing within a comment string. Allows multiple statements to be coded on the same physical source line.

Equal sign (=): Used to define equality symbols.

Asterisk (*): Used to specify location counter value.

Rubout: Ignored. May be used to overpunch preparation errors.

Dollar-sign ($): The dollar sign ($) is used to indicate the last physical statement of the program. It must appear as the first non-blank character of the last statement.

Apostrophe ('): Indicates a character string.

STATEMENTS

The statement is the basic unit used to construct assembly language programs. Each statement begins in character position one of a source line and is terminated by a carriage-return, or semi-colon (;). Use of the semicolon enables multiple statements to be coded on the same physical source line. If a statement extends past character position 72, the assembler ignores all succeeding characters until a carriage-return is encountered.

The text of a statement may be preceeded by one or more blank positions. The first non-blank position may then contain any one of the following characters:

Slash (/)
Apostrophe (')
Letter (A-Z)
Digit (0-9), plus (+), or minus (-)
Asterisk (*)

The treatment of each of these characters is described below.

SLASH

The appearance of a slash in the first non-blank statement position signifies a comment string. No action is taken by the assembler except to reproduce the statement on the program listing.

```
---------------------------------------------
.  /THIS IS A CØMMENT STATEMENT             .
.     /  THE ASSEMBLER IGNØRES              .
. /       CØMMENT STATEMENTS                .
.   / CØMMENTS ARE PRECEDED BY A SLASH (/)  .
---------------------------------------------
```

APOSTROPHE

The appearance of an apostrophe (') in the first non-blank statement position indicates the start of a character string. The ASCII value of each successive character following the initial apostrophe is output as a data word until a closing apostrophe is encountered. All characters following the closing apostrophe are ignored until a carriage-return or semicolon is encountered.

```
---------------------------------------------
.  'A'                                      .
.  'XYZ 123'                                .
.  'CHARACTER STRING'                       .
---------------------------------------------
```

LETTER

The appearance of a letter in the first non-blank statement position signifies the presence of a label, an equality symbol, an assembler mnemonic, or an arithmetic expression.

LABEL. A label consists of at most eight letters and digits beginning with a letter and followed by a comma (,). Each label is assigned a value, during assembly pass 1, equal to the value of the program location counter at the time it is encountered. Refer to the discussion of the location counter in the "Asterisk" statement text. The first non-blank character following the comma may be a semicolon (;), carriage-return, or one of the characters listed above. If one of these characters is present, it is processed exactly as though it was the first non-blank statement character.

```
---------------------------------------------
.    TEMP,                                  .
. LABEL151,                                 .
.    A12345,                                .
.       X15B24,                             .
---------------------------------------------
```

EQUALITY SYMBOL. An equality symbol consists of at most eight letters and digits beginning with a letter and followed by an equal sign (=). An expression must appear to the right of the equal sign. The assembler evaluates the expression and assigns the value to the symbol on the left of the equal sign. All expression terms must be previously defined, and the expression must be terminated by a slash (/), semicolon (;), or carriage-return. Refer to the "Expressions" text for a discussion of expression formation and evaluation.

```
---------------------------------------------
.  TEN = 10                                 .
.  TWELVE = TEN + 2                         .
.     TWENTY = TEN + TEN                    .
.  NEG = 07041    /NEGATE ØPERATØR          .
---------------------------------------------
```

ASSEMBLER MNEMONIC. An assembler mnemonic consists of at most eight letters followed by a blank, slash (/), carriage-return, or semi-colon (;). If more than four letters are specified, only the first four are used in processing the mnemonic. Mnemonics are assigned for each FABRI-TEK MP12 instruction and are described in the "Machine Instructions" text. The remaining assembler mnemonics are described below.

MNEMONIC	MEANING
DECIMAL	Set decimal conversion mode. Normally, all numeric program data is treated in the following manner: Numbers preceded by a zero (0) are treated as octal while those not preceeded by a zero are treated as decimal. The DECIMAL mnemonic directs the assembler to regard all subsequent numeric data as decimal data.
OCTAL	Set octal conversion mode. The OCTAL mnemonic directs the assembler to regard all subsequent numeric data as octal data.

All characters following a DECIMAL or OCTAL mnemonic are ignored until a carriage-return or semicolon (;) is encountered.

```
-----------------------------------------------------
.  ØCTAL   /DECLARE ØCTAL CØNVERSIØN                .
.    DECIMAL;128;-512;+1024 /DECIMAL DATA           .
. ØCTAL;77;777;DECIMAL;99;999                       .
.                                                   .
-----------------------------------------------------
```

ARITHMETIC EXPRESSION. If a label, equality symbol, or assembler mnemonic is not present, the assembler assumes an expression is specified and attempts to evaluate it. If the evaluation is successful, the value of the expression is output as a data word. The expression must be terminated by a slash (/), semicolon (;), or carriage-return.

```
-----------------------------------------------------
.  LØØP + 6                                         .
.      START+0200                                   .
.    BUFF2 - BUFF1 + 1                              .
.  DATA3, DATA + 2/ LABELED EXPRESSIØN              .
-----------------------------------------------------
```

DIGIT, PLUS, OR MINUS

The appearance of a digit (0-9), a plus sign (+), or a minus sign (-) signifies the presence of an expression which is evaluated and the value output as a data word. The expression must be terminated by a slash (/), semicolon (;) or carriage-return.

```
-----------------------------------------------------
.  -2047                                            .
.       +999                                        .
.    -SWITCH+3                                      .
. 1 - TAG                                           .
-----------------------------------------------------
```

ASTERISK

The asterisk character controls the setting of the program location counter. The location counter is used by the assembler to assign machine instructions and data into consecutive memory addresses. The value of the location counter represents the physical memory address into which any data generated by the current statement is to reside when the machine program is loaded. The assembler increments the location counter by one for each instruction or data item assembled. Statement labels are assigned the value of the location counter, during pass 1 of the assembly, at the time they are encountered. The assembler initially sets the location counter to octal 200.

The asterisk must be followed by an expression. The expression is evaluated and the value assigned to the program location counter. Each term of the expression must be previously defined, and the expression must be terminated by a slash (/), semicolon (;), or carriage-return. Refer to the ''Expressions'' text for a discussion of expression formation and evaluation.

```
---------------------------------------------
.  *0400   /SET LØCATIØN                     .
.     START, *0200                           .
.  *START+128                                .
.  BUFF,*.+72  /RESERVE 72 LØCATIØNS FØR     .
.              /BUFFER AREA                  .
---------------------------------------------
```

Note that if a label appears in conjunction with an asterisk, it is assigned the value of the location counter prior to establishing the new value of the location counter. For example, if the value of the location counter was 0763 prior to the statement BUFF, *01000 then the value assigned to the label ''BUFF'' would be 0763.

EXPRESSIONS

Expressions are formed by combining ''terms'' from left to right using plus (+) and (-) signs. Blanks may appear before, between, and after terms; however, terms may not contain imbedded blanks. An expression must be terminated by a slash (/), semicolon (;) or carriage-return.

TERMS

Terms are the basic units used in constructing expressions. The following types of terms are defined:

PERIOD (.). The period is a term which, in statement context, represents the current value of the program location counter.

NUMERIC CONSTANT. A numeric constant is a self-defining term which is treated as an octal or decimal number depending upon the conversion mode in affect at the time the term is encountered. Initially, numeric terms beginning with a zero are treated as octal numbers. Those not beginning with a zero are treated as decimal numbers. The DECIMAL and OCTAL directives may, subsequently, be used to declare strict decimal or octal conversions. All numeric terms are converted modulo 4096. Octal numbers may not contain the digits 8 or 9.

```
---------------------------------------------
.  0100      /ØCTAL 100                      .
.  2769      /DECIMAL 2769                   .
. 07776      /ØCTAL 7776                     .
.  4099      /3 [4099 MØD(4096) = 3]         .
.  8192      /0 [8192 MØD(4096) = 0]         .
---------------------------------------------
```

SYMBOL. A symbol consists of up to eight letters and digits beginning with a letter. Symbols are defined by their appearance as statement labels or equality symbols. The value of a symbol, defined as a label, is the value of the location counter at the time the label was encountered. The value of a symbol, defined by equality, is the value of the expression appearing on the right of the equal sign.

```
---------------------------------------------
.  SYMB     /FØUR LETTER SYMBØL              .
.  P1234    /ØNE LETTER, FØUR DIGITS         .
.  P1QR23   /MIXED LETTERS AND DIGITS        .
.  2SYM     /ILLEGAL, FIRST CHARACTER IS     .
.           /NØT A LETTER                    .
.  X.15     /ILLEGAL, PERIØD IS NØT A        .
.           /LETTER ØR DIGIT                 .
.  Y12 Z    /ILLEGAL, IMBEDDED BLANK         .
---------------------------------------------
```

EXPRESSION EVALUATION

Expressions are evaluated from left to right by combining the terms as indicated.

```
---------------------------------------------
.  .+6        /VALUE ØF LØCATIØN CØUNTER     .
.             /PLUS SIX                      .
.  LØØP - 3   /VALUE ØF SYMBØL "LØØP"        .
.             /MINUS THREE                   .
.   -128      /TWØ'S CØMPLEMENT ØF           .
.             /DECIMAL 128                   .
.  A-B+C      /VALUE ØF SYMBØL "A" MINUS     .
.             /VALUE ØF SYMBØL "B" PLUS      .
.             /VALUE ØF SYMBØL "C"           .
.  LAST - .   /VALUE ØF SYMBØL "LAST"        .
.             /MINUS CURRENT VALUE ØF        .
.             /LØCATIØN CØUNTER              .
---------------------------------------------
```

MACHINE INSTRUCTIONS

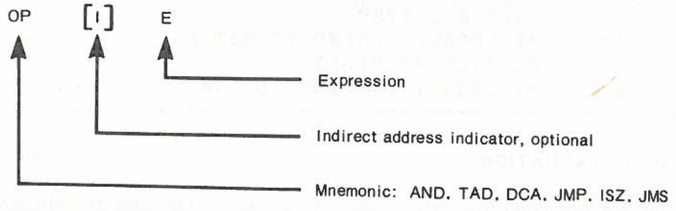
Each FABRI-TEK MP12 machine instruction is identified by a symbolic instruction mnemonic. The assembler recognizes each mnemonic and generates a binary machine instruction which corresponds to the symbolic instruction. In the following instruction descriptions, optional statement parameters are enclosed in square brackets ([ ]). All instruction statements may contain a label preceeding the instruction mnemonic, and a comment string preceeded by a slash (/). Labels and comment strings are not depicted in the following general instruction format descriptions.

MEMORY REFERENCE INSTRUCTIONS

Memory reference instructions consist of the following six instructions:

MNEMONIC	INSTRUCTION
AND	Logical "AND"
TAD	Two's complement Add
DCA	Deposit and clear accumulator
JMP	Jump
ISZ	Increment and skip if zero
JMS	Jump to subroutine

Each memory reference instruction is coded in the format:


OP
[I]
E
Expression
Indirect address indicator, optional
Mnemonic: AND, TAD, DCA, JMP, ISZ, JMS

At least one blank position must separate each of the above fields. The value of the expression E represents the primary address of the instruction operand. If the I indicator is present, the assembler sets the address mode bit in the instruction to 1 to specify indirect addressing. If the I parameter is not present, the address mode bit is set to zero to specify direct addressing. The expression E is evaluated and if the value lies within the same page as the location counter, the page specification bit in the instruction is set to one; specifying current page addressing. If the value lies within the first 128 memory locations, the page specification bit is set to zero to specify base page addressing. In either the base or current page specification mode, the least significant seven bits of the value of the expression are inserted into bit positions 5 through 11 of the instruction word.

```
-----------------------------------------------
.         AND MASK    /MASK ØFF CERTAIN BITS .
.   FETCH, TAD LØC          /FETCH DATA      .
.     DCA  SAVE           /SAVE DATA         .
.    ISZ I 010    /INCREMENT          PØINTER .
.      JMS  SUB      /JUMP TØ SUBRØUTINE     .
.    JMP I SUB  /RETURN FRØM SUBRØUTINE      .
-----------------------------------------------
```

OPERATE INSTRUCTIONS, GROUP I

The Group I operate instructions consist of the following:

MNEMONIC	INSTRUCTION
CLA	Clear accumulator
CLL	Clear link
CMA	Complement accumulator
CML	Complement link
IAC	Increment accumulator
RAR	Rotate accumulator and link right
RAL	Rotate accumulator and link left
RTR	Rotate accumulator and link right twice
RTL	Rotate accumulator and link left twice
NOP	No operation

The Group I operate instructions are coded in the format:

$$OP_1\ [OP_2]\ldots[OP_K]$$

OP_1 through OP_K represent Group I mnemonics which are combinable to form a composite Group I instruction. Each of the mnemonics must be separated by at least one blank position. The rules of combination for Group I instructions are summarized below.

1. NOP is not combinable.

2. Only one of the shift instructions RAR, RTR, RAL, RTL may appear in a combined instruction.

3. The execution sequence for a composite Group I instruction is defined as follows:

   a. CLA and CLL, if specified, are executed first.
   b. CMA and CML, if specified, are executed next.
   c. IAC, if specified, is executed next.
   d. One of RAR, RTR, RAL, RTL, if specified, is executed last.

```
---------------------------------------------
.  CLA      /CLEAR ACCUMULATØR              .
.   CLA CLL CML /CLEAR AC AND SET LINK      .
. CMA IAC   /NEGATE ACCUMULATØR             .
.  CLA CLL CML RTL /LØAD AC WITH +2         .
.    NØP    /NØ ØPERATIØN                   .
.  INIT,CLA CMA /LØAD AC WITH 7777          .
---------------------------------------------
```

OPERATE INSTRUCTIONS, GROUP II

The Group II operate instructions are listed below.

MNEMONIC	INSTRUCTION
CLA	Clear accumulator
SMA	Skip on minus accumulator
SPA	Skip on positive accumulator
SZA	Skip on zero accumulator
SNA	Skip on non-zero accumulator
SNL	Skip on non-zero link
SZL	Skip on zero link
SKP	Skip
OSR	Inclusive "OR" switch register with accumulator
HLT	Halt

The Group II operate instructions are coded in the format:

$$OP_1 \; [OP_2] \ldots [OP_K]$$

OP_1 through OP_K represent Group II mnemonics which are combinable to form a composite Group II instruction. Each of the mnemonics must be separated by at least one blank position. The rules of combination for Group II instructions are summarized below.

1. For the skip group SPA, SNA, and SZL, a combination of these instructions will result in a skip only when all specified skip conditions are true.

2. For the skip group SMA, SZA, and SNL, a combination of these instructions will result in a skip only when at least one of the specified skip conditions is true.

3. Only members of one skip group may appear in a combined instruction.

4. SKP is combinable only with CLA, OSR, and HLT.

5. The execution sequence for a combined Group II operate instruction is defined as follows:

   a. Skip instructions are executed first.
   b. CLA, if specified, is executed next.

   c. OSR, if specified, is executed next.
   d. HLT, if specified, is executed last.

```
---------------------------------------------------
.  SMA SZA    /SKIP IF AC NEGATIVE ØR ZERØ        .
.   TEST,SNA SZL  /SKIP IF AC IS NØN-ZERØ         .
.                 /AND LINK IS ZERØ               .
.  CLA  ØSR  /TRANSFER SWITCH REGISTER            .
.            /CØNTENTS TØ ACCUMULATØR             .
.   SNA CLA   /SKIP IF AC IS NØN-ZERØ AND         .
.             /CLEAR AC                           .
.  HALT,CLA HLT  /CLEAR AC AND HALT               .
---------------------------------------------------
```

INPUT/OUTPUT AND INTERRUPT INSTRUCTIONS

The Input/Output and interrupt instructions consist of the following:

MNEMONIC	INSTRUCTION
IOT	I/O Transfer
ION	Enable interrupt system
IOF	Disable interrupt system
SPL	Skip if power is low
CON	Turn clock on
COF	Turn clock off
SCD	Skip if clock done

The IOT instruction is coded in the format:

IOT E

E is an expression that must be separated from the IOT mnemonic by at least one blank position. The expression E is evaluated, and the least significant nine bits of the value inserted into bit positions 3 through 11 of the instruction. Bits 3-8 represent a device address, and bits 9-11 represent one of seven input-output function codes as described in the Input-Output Interface information.

The interrupt instructions ION, IOF, SPL, CON, COF, and SCD are coded in the format:

OP

where OP is one of the above mnemonics. All characters following the mnemonic OP are ignored until a semicolon (;) or carriage-return is encountered.

```
---------------------------------------------------
.  IØT 031   /SKIP IF KEYBØARD/READER READY .
.   XFER, IØT 046  /PRINT CHARACTER         .
.     IØN  /ENABLE INTERRUPT FACILITY       .
.     IØF  /TURN INTERRUPTS ØFF             .
. TEST, SPL /SKIP IF PØWER IS LØW           .
---------------------------------------------------
```

ERROR PROCESSING

Errors detected during the assembly process result in an error flag being printed to the left of the statement which originated the error. The following information describes the error flags.

ERROR FLAG	MEANING
S	STATEMENT ERROR. An illegal or unexpected character was encountered in processing the current statement.
P	PAGE ERROR. A memory reference instruction operand does not lie within the current page or the base page, as required. The instruction cannot be assembled in its present form. A current page address of 0177_8 is assumed.
I	ILLEGAL COMBINATION. An illegal instruction combination has been specified in the current statement.
D	DOUBLY DEFINED SYMBOL. A statement symbol has been previously defined. The value assigned at the first definition is used. This error is only indicated during assembly pass 1.
U	UNDEFINED SYMBOL. A program symbol has not been defined in any statement. The value 0 is assigned.
F	SYMBOL TABLE FULL. No further symbols are stored.

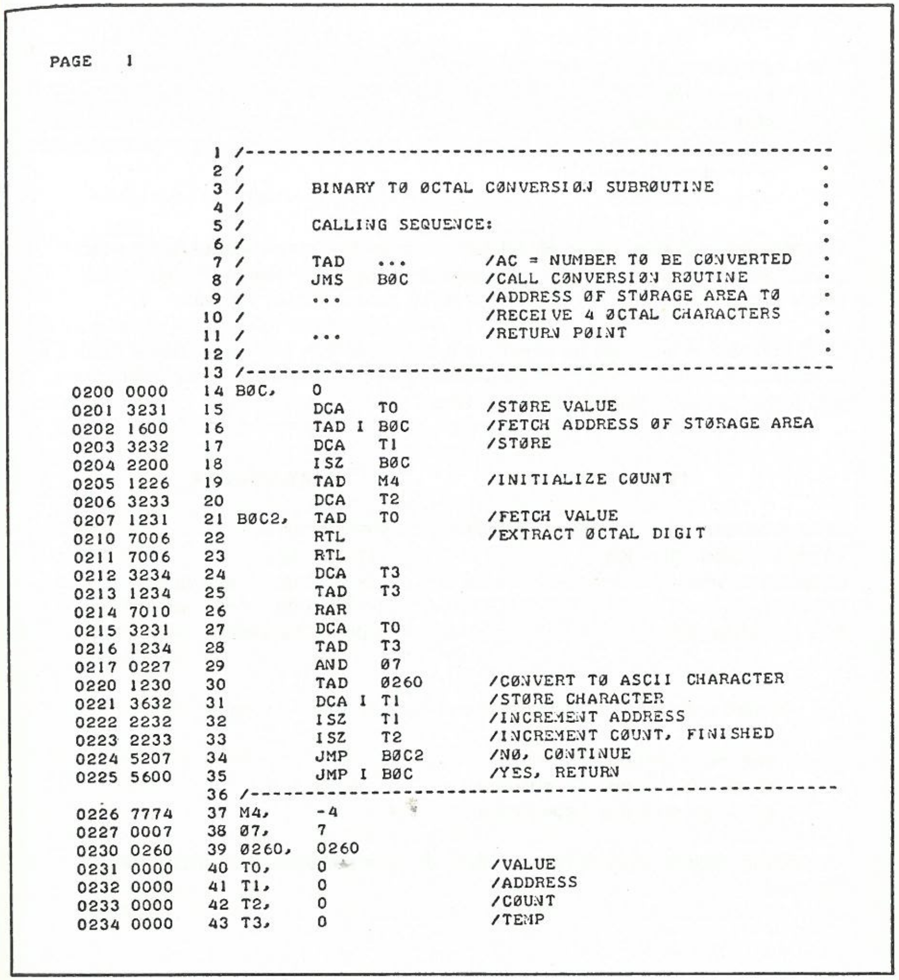
If multiple errors result from the same statement, only the last error detected is indicated.

ASSEMBLY LISTING

The assembly listing is produced during pass 3. Each statement is printed in the following format:

```
-----------------------------------------------
.  PRINT PØSITIØN -     1:  ERRØR FLAG        .
.                       2:  BLANK             .
.                   3 - 6:  LØCATIØN (ØCTAL)  .
.                       7:  BLANK             .
.                   8 -11:  DATA (ØCTAL)      .
.                      12:  BLANK             .
.                   13-16:  STATEMENT NUMBER  .
.                      17:  BLANK             .
.                   18-72:  PRØGRAM STATEMENT .
-----------------------------------------------
```

The listing is printed with 52 lines per page; each page is numbered in decimal. Eleven inch page separation marks, consisting of six dashed lines, are printed to aid manual page separation. A sample listing is provided below.



```
PAGE   1



                 1 /------------------------------------------------------
                 2 /                                                      .
                 3 /     BINARY TØ ØCTAL CØNVERSIØN SUBRØUTINE            .
                 4 /                                                      .
                 5 /     CALLING SEQUENCE:                                .
                 6 /                                                      .
                 7 /     TAD   ...        /AC = NUMBER TØ BE CØNVERTED    .
                 8 /     JMS   BØC        /CALL CØNVERSIØN RØUTINE        .
                 9 /     ...              /ADDRESS ØF STØRAGE AREA TØ     .
                10 /                      /RECEIVE 4 ØCTAL CHARACTERS     .
                11 /     ...              /RETURN PØINT                   .
                12 /                                                      .
                13 /------------------------------------------------------
0200 0000       14 BØC,  0
0201 3231       15       DCA   T0         /STØRE VALUE
0202 1600       16       TAD I BØC        /FETCH ADDRESS ØF STØRAGE AREA
0203 3232       17       DCA   T1         /STØRE
0204 2200       18       ISZ   BØC
0205 1226       19       TAD   M4         /INITIALIZE CØUNT
0206 3233       20       DCA   T2
0207 1231       21 BØC2, TAD   T0         /FETCH VALUE
0210 7006       22       RTL              /EXTRACT ØCTAL DIGIT
0211 7006       23       RTL
0212 3234       24       DCA   T3
0213 1234       25       TAD   T3
0214 7010       26       RAR
0215 3231       27       DCA   T0
0216 1234       28       TAD   T3
0217 0227       29       AND   Ø7
0220 1230       30       TAD   Ø260       /CØNVERT TØ ASCII CHARACTER
0221 3632       31       DCA I T1         /STØRE CHARACTER
0222 2232       32       ISZ   T1         /INCREMENT ADDRESS
0223 2233       33       ISZ   T2         /INCREMENT CØUNT, FINISHED
0224 5207       34       JMP   BØC2       /NØ, CØNTINUE
0225 5600       35       JMP I BØC        /YES, RETURN
                36 /------------------------------------------------------
0226 7774       37 M4,   -4
0227 0007       38 Ø7,   7
0230 0260       39 Ø260, 0260
0231 0000       40 T0,   0                /VALUE
0232 0000       41 T1,   0                /ADDRESS
0233 0000       42 T2,   0                /CØUNT
0234 0000       43 T3,   0                /TEMP
```

Figure 1-14
BINARY TO OCTAL CONVERSION SUBROUTINE