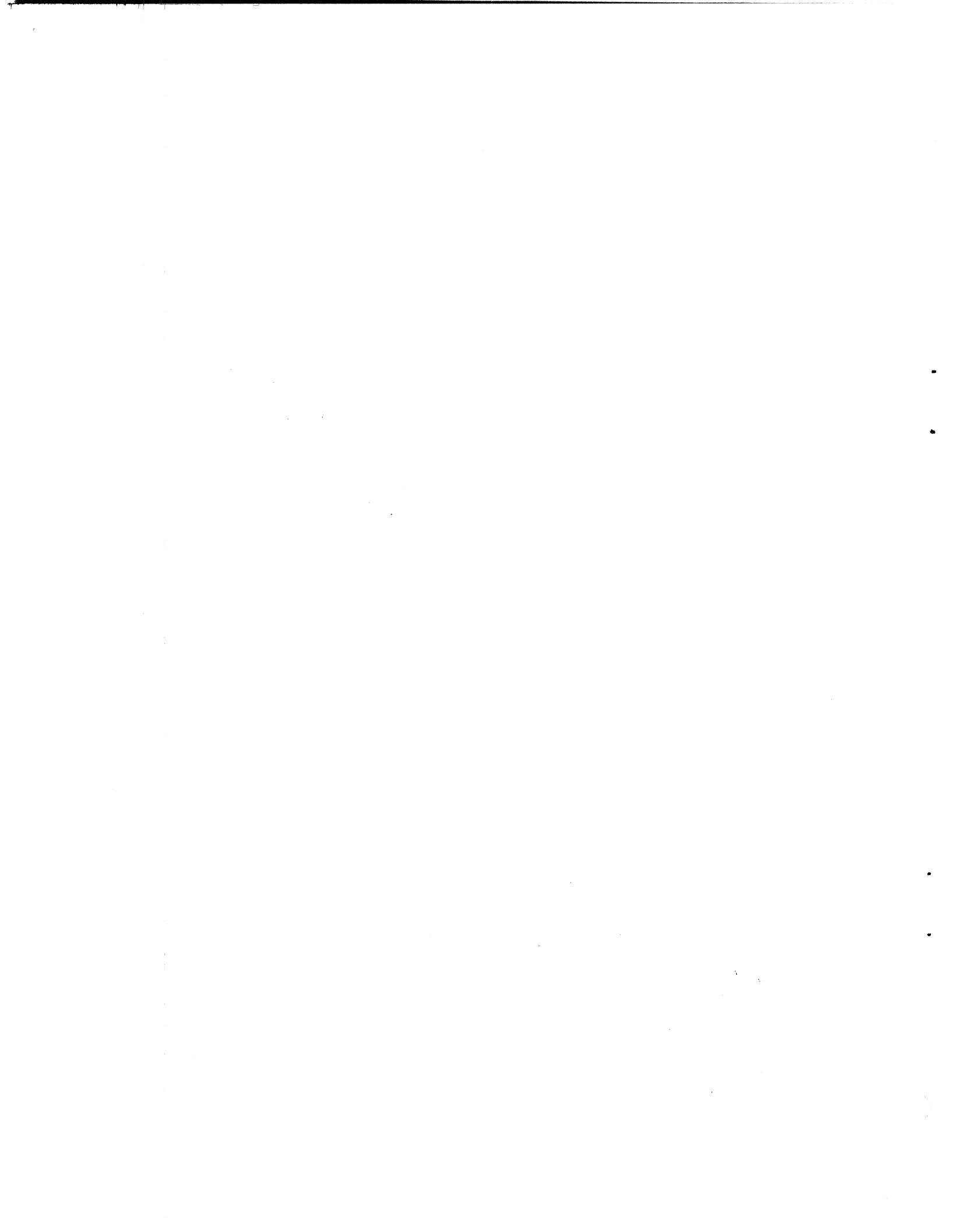


**POINT 4  
MARK III  
COMPUTER  
USER MANUAL**

**POINT 4**  
**DATA CORPORATION**

The logo for Point 4 Data Corporation features the word "POINT" in a large, bold, sans-serif font. To its right is a stylized number "4" composed of four curved segments. Below "POINT" and the "4" is the word "DATA CORPORATION" in a smaller, bold, sans-serif font.



P O I N T 4 D A T A C O R P O R A T I O N  
2569 McCabe Way, Irvine, California 92714

D R A F T

P O I N T 4  
M A R K I I I  
C O M P U T E R  
U S E R M A N U A L

444  
4444 4 44  
44 444 44444  
4 44444 444444  
4444444 444444

4444444444 44444  
4444444444 4444  
4444444 44  
44444

D R A F T V E R S I O N

## NOTICE

Every attempt has been made to make this reference manual complete, accurate and up-to-date. However, all information herein is subject to change due to updates. All inquiries concerning this manual should be directed to POINT 4 Data Corporation.

**D R A F T**

Copyright © 1981 by POINT 4 Data Corporation (formerly Educational Data Systems, Inc). Printed in the United States of America. All rights reserved. No part of this work covered by the copyrights hereon may be reproduced or copied in any form or by any means--graphic, electronic, or mechanical, including photocopying, recording, taping, or information and retrieval systems--without the written permission of:

POINT 4 Data Corporation  
2569 McCabe Way  
Irvine, CA 92714  
(714) 754-4114

# REVISION RECORD

---

**PUBLICATION NUMBER: HM-080-0019**

<u>Revision</u>	<u>Description</u>	<u>Date</u>
01	Draft Version to coincide with delivery of first MARK III	05/15/81

## LIST OF EFFECTIVE PAGES

---

Changes, additions, and deletions to information in this manual are indicated by vertical bars in the margins or by a dot near the page number if the entire page is affected. A vertical bar by the page number indicates pagination rather than content has changed.

<u>Page</u>	<u>Rev</u>	<u>Page</u>	<u>Rev</u>	<u>Page</u>	<u>Rev</u>
Cover	-				
Title	-				
ii thru xiii	01				
1-1 thru 1-12	01				
2-1 thru 2-6	01				
3-1 thru 3-20	01				
4-1 thru 4-5	01				
5-1 thru 5-35	01				
6-1 thru 6-95	01				
Appendix Title	-				
A-1 thru A-n	01				
B-1 thru B-n	01				
C-1 thru C-n	01				
D-1 thru D-n	01				
Comment Sheet	01				
Mailer	-				
Back Cover	-				

**PREFACE**

---

**DRAFT**





## CONTENTS

---

<u>Section</u>	<u>Title</u>	<u>Page</u>
1	INTRODUCTION AND GENERAL DESCRIPTION	1-1
1.1	Scope	1-1
1.2	General Description	1-1
1.2.1	Features	1-3
1.3	Equipment Characteristics	1-4
1.3.1	Performance Characteristics	1-4
1.3.2	Equipment Specifications	1-5
1.4	System Architecture	1-6
1.4.1	System Functional Units	1-8
1.4.1.1	Central Processor and Memory Board	1-9
1.4.1.2	Processor Chassis and Front Panel	1-11
1.4.1.3	Processor Mini-Panel	1-11
1.4.1.4	Power Supply Board	
2	INSTALLATION	2-1
2.1	Environmental Requirements	2-1
2.2	Unpacking Instructions	2-2
2.3	Assembly Procedures	2-4
3	OPERATING PROCEDURES	3-1
3.1	Processor Mini-Panel	3-1
3.1.1	Power Controls	3-3
3.1.2	Processor Operation Monitoring	3-4
3.1.3	Program Execution Control	3-4
3.2	Virtual Control Panel	3-5
3.2.1	Command Descriptions	3-7
3.3	Processor/CTU Interface	3-10
3.3.1	CTU Commands	3-10
3.3.1.1	Command Functions	3-11
3.3.1.2	Command Format	3-12
3.3.1.3	CTU Error Conditions	3-13
3.3.2	CTU Commands in MANIP	3-14
3.3.3	CTU Commands in DEBUG	3-16
3.4	Powering Up the System	3-19
3.5	Diagnostic Checks	3-20
3.5.1	Diagnostic Capabilities	3-20
3.5.2	Self-Test Operating Procedures	3-20
3.5.3	Self-Test Errors	3-20

4	INPUT/OUTPUT INTERFACES	4-1
4.1	Input/Output Bus Interface Signals	4-1
4.1.1	Input/Output Interface Signals	4-1
4.1.2	Backplane Pin Signal Connectors	4-4
5	INSTRUCTION REPERTOIRE	5-1
5.1	Introduction	5-1
5.2	Octal Representation and Two's Complement Notation	5-1
5.3	Instruction Types	5-3
5.4	Memory Reference	5-5
5.4.1	Memory Addressing	5-5
5.4.1.1	Indexing Mode	5-6
5.4.1.2	Indirect Addressing Operations	5-7
5.4.2	Types of Memory Reference Instructions	5-8
5.4.2.1	Move Data Instructions	5-8
5.4.2.2	Jump and Modify Memory Instructions	5-9
5.4.2.3	Assembler Language Conventions and Addressing Examples	5-10
5.5	Arithmetic and Logic Instruction Group	5-12
5.5.1	Arithmetic and Logic Processing	5-12
5.5.1.1	Arithmetic/Logic Operations	5-14
5.5.1.2	Overflow and Carry-Out Operations	5-15
5.5.2	Arithmetic/Logic Functions	5-18
5.5.3	Secondary Functions	5-19
5.5.3.1	Shift Field (SH)	5-19
5.5.3.2	Carry Control Field (CY)	5-20
5.5.3.3	No-Load Field (NL)	5-20
5.5.3.4	Skip Control Field (SK)	5-21
5.5.4	Assembler Language Conventions and Examples	5-22
5.6	Input/Output Instruction Group	5-24
5.6.1	Programmed I/O Instructions	5-24
5.6.1.1	I/O Transfer and Device Control Instructions	5-26
5.6.1.2	Assembler Language Conventions and Examples	5-27
5.6.2	Special Code 77 (CPU) Instructions	5-29
5.6.2.1	Special Mnemonics for CPU Instructions	5-29
5.6.2.2	Control Field Uses	5-31
5.6.2.3	Skip Instructions	5-33
5.6.2.4	Assembler Language Conventions and Examples	5-33
5.7	Instruction Execution Times	5-34

6	PERIPHERAL DEVICE HANDLING	6-1
6.1	Peripheral Interface Board	6-1
6.1.1	System Description	6-1
6.1.2	Equipment Characteristics	6-3
6.1.2.1	Performance Characteristics	6-3
6.1.2.2	Equipment Specifications	6-4
6.1.3	PIB Internal Architecture	6-4
6.2	Input/Output Interface Handling	6-6
6.2.1	Program Interrupt and Priority Scheme	6-6
6.2.1.1	Interrupt Sequence	6-7
6.2.1.2	Programming Polling and Interrupts	6-8
6.2.2	Programmed Transfers	6-14
6.2.2.1	Master Terminal Interface	6-14
6.2.3	Direct Memory Access Transfers	6-16
6.2.3.1	Enabling/Disabling DMA Transfers	6-17
6.2.3.2	Initiating DMA Transfer	6-18
6.2.3.3	Turning Off Automatic Block Transfer Mode for Disc or Tape	6-19
6.2.3.4	Turning Off Automatic Block Transfer Mode for Multiplexer	6-19
6.3	MUX General Description	6-20
6.3.1	MUX Hardware Configuration	6-21
6.3.1.1	Baud-Rate Selection	6-21
6.3.1.2	MARK III Cabling Connections	6-22
6.3.2	Communications Byte Mode	6-24
6.3.2.1	Communications Controller IC 6850	6-25
6.3.2.2	Command Register	6-25
6.3.2.3	Standard Mode of Operation	6-28
6.3.2.4	Operational Mode Modifications	6-28
6.3.2.5	Status Register	6-29
6.3.2.6	Software Polling	6-33
6.3.3	Communications Auto Mode	6-35
6.3.3.1	MUX I/O Control Block	6-35
6.3.3.2	Control Word Definitions	6-35
6.3.3.3	Input Operations	6-37
6.3.3.3.1	Input Control Word	6-37
6.3.3.3.2	Input Termination Status	6-42
6.3.3.3.3	Input Byte Pointer	6-43
6.3.3.3.4	Last Input Byte (LIB)	6-43
6.3.3.4	Output Operations	6-44
6.3.3.4.1	Output Control Word	6-44
6.3.3.4.2	Output Termination Status	6-49
6.3.3.4.3	Output Byte Pointer	6-50
6.3.3.4.4	Last Output Byte	6-50
6.3.3.5	Initialization Procedures	6-51
6.3.3.5.1	Setting The Pointer	6-51
6.3.3.5.2	Initialization Guidelines	6-52
6.3.3.5.3	Enabling/Disabling Ports and Port Parameters	6-53
6.3.3.5.4	Enabling/Disabling Transmitters and Receivers	6-54
6.3.3.5.5	Polling MUX Interrupt	6-56
6.3.3.6	Deactivating MUX	6-56

6.4	SMD/CMD Disc Interface	6-57
6.4.1	Performance Characteristics	6-58
6.4.2	Drive Requirements	6-59
6.4.3	Multi-Drive Connection	6-59
6.4.4	Operation	6-61
6.4.4.1	Sector Format	6-62
6.4.4.2	Sector Verification	6-63
6.4.4.3	Data Transfer	6-63
6.4.4.4	File Linking	6-63
6.4.4.5	Data Verification	6-63
6.4.4.6	Error Checking and Status Reporting	6-64
6.4.4.7	Interrupt Operation	6-64
6.4.4.8	Initiating an Operation	6-64
6.4.4.9	Deactivating The Controller	6-65
6.4.4.10	Seek Control	6-65
6.4.4.11	Seek Error Recovery	6-65
6.4.5	Input/Output Control Block	6-66
6.4.5.1	Opcode (Word 0)	6-68
6.4.5.2	Unit Select (Word 1)	6-69
6.4.5.3	Cylinder Select (Word 2)	6-70
6.4.5.4	Head Select (Word 3)	6-71
6.4.5.5	Sector Select (Word 4)	6-72
6.4.5.6	Sector Count (Word 5)	6-73
6.4.5.7	Memory Address (Word 6)	6-74
6.4.5.8	Termination Status (Word 7)	6-75
6.4.6	Input/Output Instructions	6-78
6.4.6.1	Programmed I/O Input	6-79
6.4.6.2	Programmed I/O Output	6-81
6.4.6.3	Controller Command Input	6-85
6.4.6.4	Controller Command Output	6-86
6.4.7	Write Data Operations	6-87
6.4.8	Read Data Operation	6-87
6.4.9	Read Verify Operation	6-88
6.4.10	Read Regardless Operation	6-88
6.4.11	Formatting Operation	6-89
6.4.11.1	Memory Sector Block Description	6-90
6.4.11.2	Formatting Procedure	6-93
6.4.11.3	Format Considerations	6-95
6.4.12	Programming Considerations	6-95

## FIGURES

<u>Number</u>	<u>Title</u>	<u>Page</u>
1-1	The POINT 4 MARK III Computer	1-2
1-2	Typical Configuration	1-7
1-3	POINT 4 MARK III Computer System Block Diagram	1-10
2-1	POINT 4 MARK III Processor Chassis Packaging	2-3
2-2	Power Supply Chassis and Processor Chassis Mounting Slots	2-5
2-3	Typical POINT 4 MARK III Board Configuration	2-6
3-1	POINT 4 Processor Mini-Panel	3-2
4-1	Input/Output Signals	4-2
4-2	Backplane I/O Signals	4-5
5-1	POINT 4 Instruction Format Summary	5-4
5-2	Arithmetic/Logic Operations	5-13
5-3	Overflow and Carry Operations Analysis for Signed Integers	5-17
6-1	Basic System Block Diagram	6-2
6-2	PIB Internal Architecture	6-5
6-3	Software Polling/Interrupt Handling Flowchart	6-10
6-4	I/O Service Routine Multiplexer Handling Procedures	6-12
6-5	I/O Service Routine Disc Tape Handling Procedures	6-13
6-6	MARK III Asynchronous MUX/CRT Cable Connector Wiring	6-22
6-7	MARK III Asynchronous MUX/Line Printer Cable Connector Wiring	6-23
6-8	Command Register on Port 0	6-26
6-9	Status Register on Port 0	6-30
6-10	Software Polling Flowchart For Output Operations	6-34
6-11	MUX IOCB	6-36
6-12	Input Control Word - Word 0	6-38
6-13	Output Control Word - Word 1	6-45
6-14	Transmitter/Receiver Example	6-54
6-15	Daisy Chain Drive Connection	6-60
6-16	Disc IOCB Format	6-67
6-17	Sector Table in Memory	6-89
6-18	Content of Each Block in The Sector Block	6-90
6-19	Sector Format	6-92
6-20	Format IOCB Example	6-93

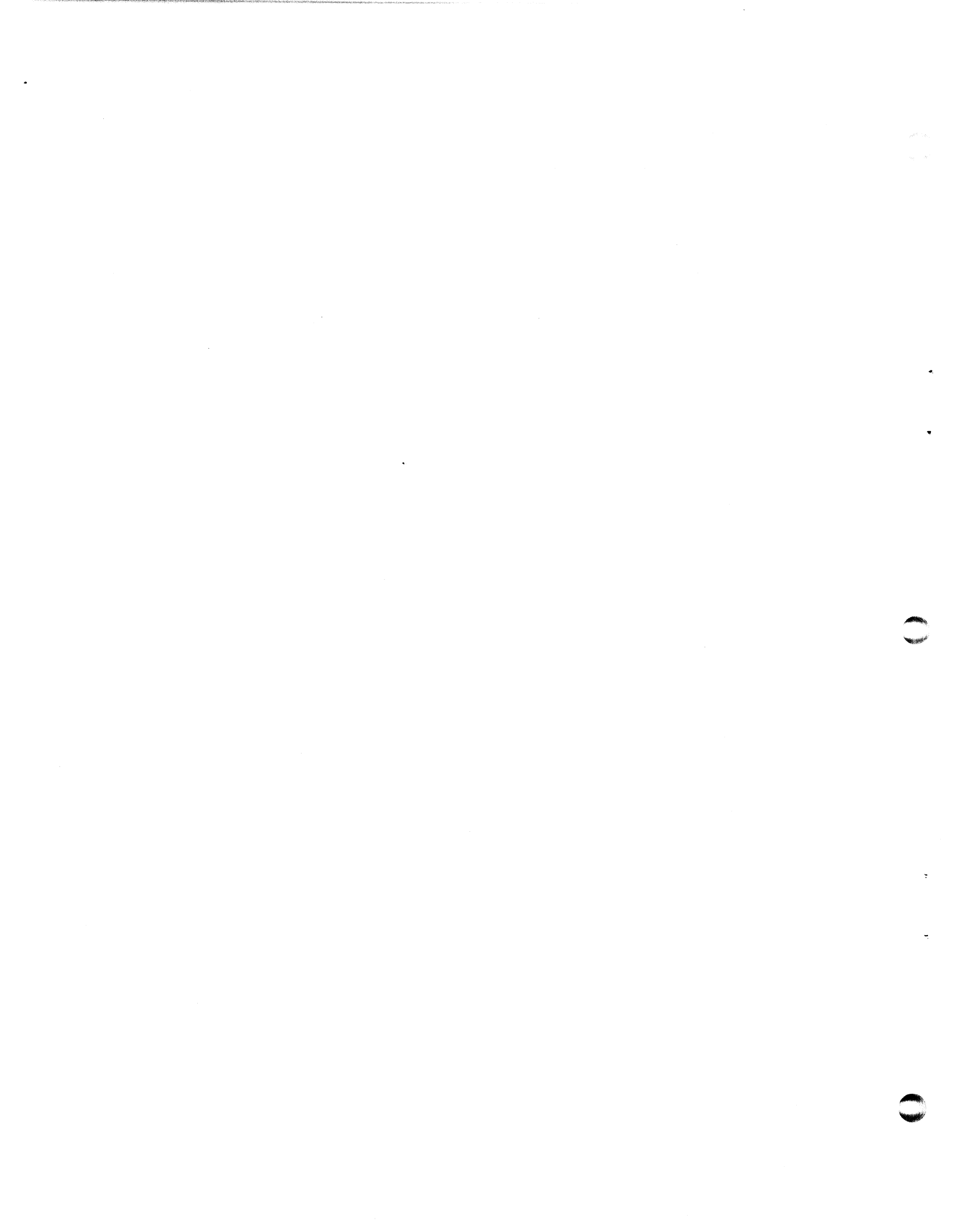
## TABLES

<u>Number</u>	<u>Title</u>	<u>Page</u>
3-1	Power Control Switch Functions	3-3
3-2	Power OK LED Interpretations	
3-3	Summary of MANIP Command Functions	3-5
3-4	MANIP Commands	3-8
3-5	CTU Command Functions	3-11
3-6	CTU Error Codes	3-13
3-7	CTU Commands in MANIP	3-14
3-8	CTU Commands in DEBUG	3-16
3-9	Summary and Overview of Data Transfer Commands	3-18
4-1	Input/Output Signals by Classification	4-3
5-1	Indexing Modes	5-6
5-2	Move Data Instructions	5-8
5-3	Jump and Modify Memory Instructions	5-9
5-4	Assembler Language Conventions for Memory Reference Instructions	5-11
5-5	Arithmetic Logic Instructions	5-18
5-6	Shift Field Functions	5-19
5-7	Carry Control Field Functions	5-20
5-8	Skip Control Field Functions	5-21
5-9	Assembler Language Conventions for Arithmetic and Logic Instructions	5-23
5-10	I/O Transfer Instructions	5-26
5-11	Assembly Language Conventions for Input/Output Instructions	5-28
5-12	Special CPU I/O Instructions	5-30
5-13	Control Field Definitions for I/O Instructions with Device Code 77	5-32
5-14	Skip Instructions	5-33
5-15	Instruction Execution Times	5-34
6-1	Baud-Rate Configuration	6-21
6-2	Port Assignments for Baud Rate Headers	6-21
6-3	DOA, DIA Device Codes	6-24
6-4	Command Register Bit Functions	6-27
6-5	Status Register Bit Operations	6-32
6-6	Input Control Word Definit	6-39
6-7	Output Control Word Definitions	6-46
6-8	Enabling/Disabling Operation	6-55
6-9	Disc IOCB Word 4 Definition	6-68
6-10	Disc IOCB Word 1 Definition	6-69
6-11	Disc IOCB Word 2 Definition	6-70
6-12	Disc IOCB Word 3 Definition	6-71
6-13	Disc IOCB Word 4 Definition	6-72
6-14	Disc IOCB Word 5 Definition	6-73
6-15	Disc IOCB Word 6 Definition	6-74
6-16	Disc Termination Status Definition	6-76

6-17 DIA Instruction Accumulator Bits  
6-18 DOA Instruction Accumulator Bits  
6-19 TAG 3 Bit Functions  
6-20 Memory Sector Block Fields

6-80  
6-82  
6-84  
6-91

DRAFT





## SECTION 1

### INTRODUCTION AND GENERAL DESCRIPTION

---

#### 1.1 SCOPE

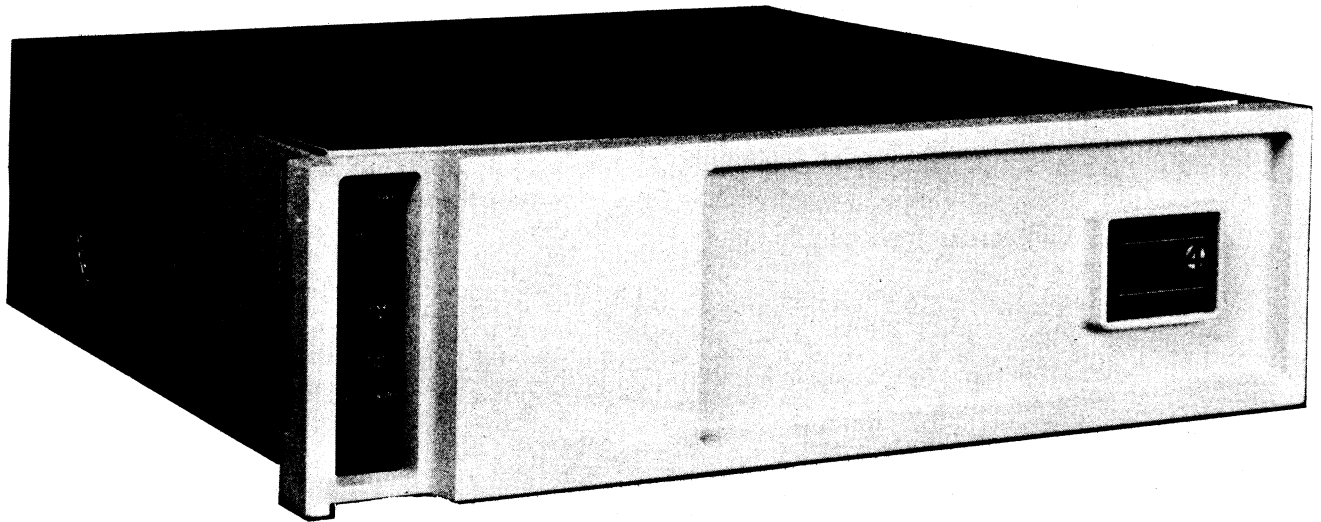
This manual provides general reference information, including system description, equipment specifications, description of the instruction set, interface data, and installation and operation information for the POINT 4 MARK III Computer. This manual is designed to be a general reference guide for both the programmer and the operator, and as such contains both detailed information to familiarize the user with the system, and charts for use as quick reference material.

The manual is organized into six sections, as summarized below:

Section I	Introduction and system architecture description
Section II	Installation methods
Section III	Operating procedures and diagnostic guidelines
Section IV	System interface signals
Section V	Instruction set
Section VI	Peripheral interface guidelines

#### 1.2 GENERAL DESCRIPTION

The POINT 4 MARK III Computer is a 16-bit, high-speed, general purpose minicomputer with a versatile instruction set. The POINT 4 MARK III employs a novel design architecture to achieve the simplicity and flexibility of a microprogrammed design and the speed of a hard-wired logic design. In addition the design allows direct addressing of up to 32K words of MOS random access memory. These features make the POINT 4 MARK III Computer well suited to OEM applications in business data systems, and control systems. See Figure 1-1 for a photograph of the POINT 4 MARK III Computer.



**Figure 1-1. The POINT 4 MARK III Computer**

### 1.2.1 FEATURES

The POINT 4 MARK III Computer includes the following features:

- o CPU and 64 bytes of RAM on the same board
- o Peripheral Interface Board (PIB) with built-in Multiplexer, printer, disc, and tape interfaces
- o Up to four asynchronous ports, baud-rate selectable to 9600 baud
- o Up to two sector-mark signal type SMD/CMD drives
- o Up to four streamer tape drives
- o Disc transfer rates to 1.25 megabytes per second
- o Virtual control panel
- o Internal power supply for CPU and PIB boards

## 1.3 EQUIPMENT CHARACTERISTICS

### 1.3.1 PERFORMANCE CHARACTERISTICS

Word Length:	16-bits
General Purpose Accumulators:	4
Memory Cycle Time:	600 nanoseconds <i>ALU</i>
RAM Access Time:	200 nanoseconds <i>Read &amp; write</i>
Microprogram Cycle Time:	200 nanoseconds
Memory:	64K bytes
DMA for Disc Controller	
Input -	1200 nanoseconds per word
Output -	1200 nanoseconds per word
DMA for Multiplexer	
Automatically vectored byte access -	
10 microseconds per byte	
Interrupt Response:	2000 nanoseconds
Backplane:	100-pin edge connector

## 1.3.2 EQUIPMENT SPECIFICATIONS

### POWER SUPPLY

#### AC Voltages:

105 to 125 VAC, 5 amps max. 47 to 63 Hz  
210 to 250 VAC, 2.5 amps max, 47 to 63 Hz

#### Voltages Provided:

+5V @ 20A  
-5V @ .5A  
+12V @ 1.5A  
-12V @ .1A

### OPERATING ENVIRONMENT

Ambient Temperature Range: 0-122F (0-50C)  
Relative Humidity: 10 to 90% noncondensing

### MECHANICAL

#### Processor Chassis

Dimensions - 5.25 inches high  
19.0 inches wide  
17.5 inches deep  
(14 cm x 48.3 cm x 44.5 cm)  
Weight: 30 pounds (13.5 kilograms)

#### CPU/Memory PCB

Dimensions: 14.5" x 12"  
(36.8 cm x 30.5 cm)

#### Peripheral Interface Board

Dimensions: 14.5" x 12"  
(36.8 cm x 30.5 cm)

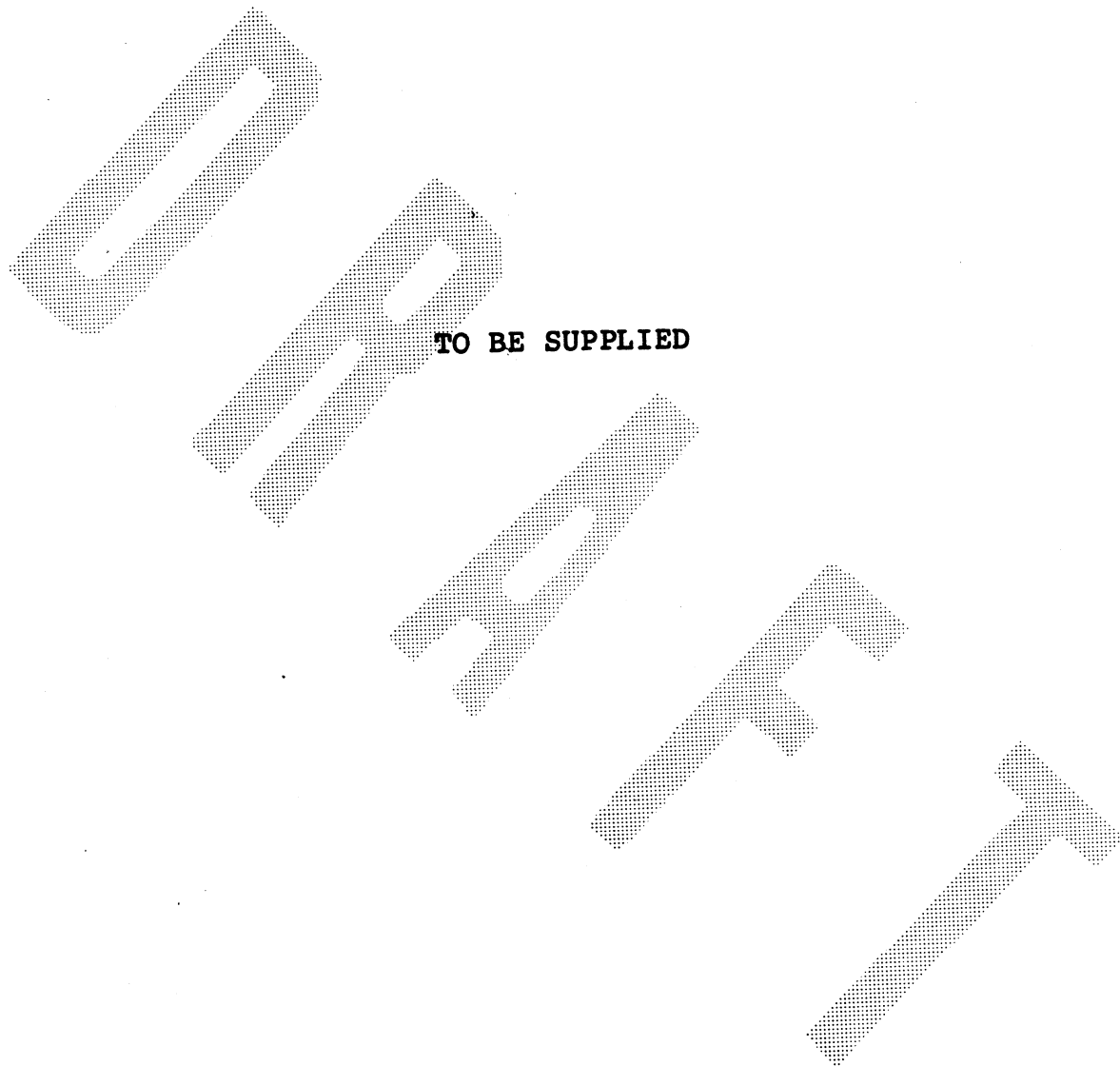
#### Power Supply Board

Dimensions: 14.5" x 12"  
(36.8 cm x 30.5 cm)

**1.4 SYSTEM ARCHITECTURE**

**TO BE SUPPLIED**

**DRAFT**



**Figure 1-2. Typical Configuration**

**1.4.1 SYSTEM FUNCTIONAL UNITS**

**TO BE SUPPLIED**

**SECRET**



#### 1.4.1.1 Central Processor and Memory Board

The Central Processor board contains all basic elements of the CPU:

- o 2903 Bit Slice Microprocessor (4 bits wide) containing four general-purpose accumulators, plus 12 special-purpose registers and arithmetic logic functions
- o Microprogrammed Control Store
- o Instruction Register
- o Main Data Bus
- o Program Counter
- o Effective Address Register
- o Timing Control
- o Input/Output Control
- o 32K Words of Random Access Memory (RAM)
- o CRC Error Detection Logic for Disc Drive Interface
- o APL PROM containing a program to implement Virtual Control Panel Features, and the Self Test

Figure 1-3 is a block diagram of the POINT 4 MARK III system, showing logic to handle each of the above functions.



**Figure 1-3. POINT 4 MARK III Computer System Block Diagram**

### **1.4.1.2 Processor Chassis and Front Panel**

The POINT 4 MARK III processor chassis is designed to be mounted in a standard 19-inch equipment rack. The 3-slot processor chassis is 5.25 inches high, 19 inches wide and 17.5 inches deep.

Cooling is provided by two fans mounted on the left side of the chassis behind the Mini-panel.

The front panel snaps on and off. There are no screws or hinges holding it in place. No cabling exists between the front panel and the chassis since the Mini-panel is mounted directly onto the chassis and its controls are accessible through a slot on the left side of the front panel.

### **1.4.1.3 Processor Mini-Panel**

The Mini-Panel on the POINT 4 MARK III chassis houses 2 sets of controls and indicators for basic processor operation. Controls and indicators are located on the left side of the chassis, as indicated in Figure 1-5. The Mini-Panel is capable of three operating functions: processor monitoring indicators, program execution controls, and power controls and indicators. For further detail, see Section 3.1 on Mini-Panel operations.

The Virtual Control Panel allows monitoring and control of the processor from a master terminal, using the manipulator program MANIP. For further details see Section 3.2 on Virtual Control Panel operations.

#### 1.4.1.4 Power Supply Board

The power supply is packaged on a single board (14.5" wide, 12" deep, and 2" high) with protective cover. It plugs directly into the MARK III backplane. Input voltage requirements and power supply output voltages are:

##### 1. AC INPUT

1.1 Voltage: 117 VAC + 10% - 15%  
234 VAC + 10% - 15% (jumper programmed)

1.2 Frequency: 47 - 63 HZ

1.3 Power: 180 VA maximum

1.4 Inrush Current: 12 amps peak

##### 2. DC OUTPUTS

###### 2.1 Voltages:

+ 5 VDC @ 20 amps  
- 5 VDC @ 0.5 amps  
+ 12 VDC @ 1.5 amps  
- 12 VDC @ 0.1 amps

## SECTION 2

### INSTALLATION

---

#### 2.1 ENVIRONMENTAL REQUIREMENTS

The location in which the POINT 4 MARK III Computer will be used must be evaluated prior to installation, to ensure that all power and cooling requirements are met. The following pre-installation considerations are necessary:

**Power Requirements:** The POINT 4 MARK III requires a power source of 105 to 125 VAC, 47 to 63 Hz with 5 amperes maximum current draw; or a 210 to 250 VAC, 47 to 63 Hz power source with 2.5 amperes current draw. In addition to power requirements for the POINT 4 MARK III, consideration must be made of power resources and electrical outlets to handle all peripheral devices to be used with the processor.

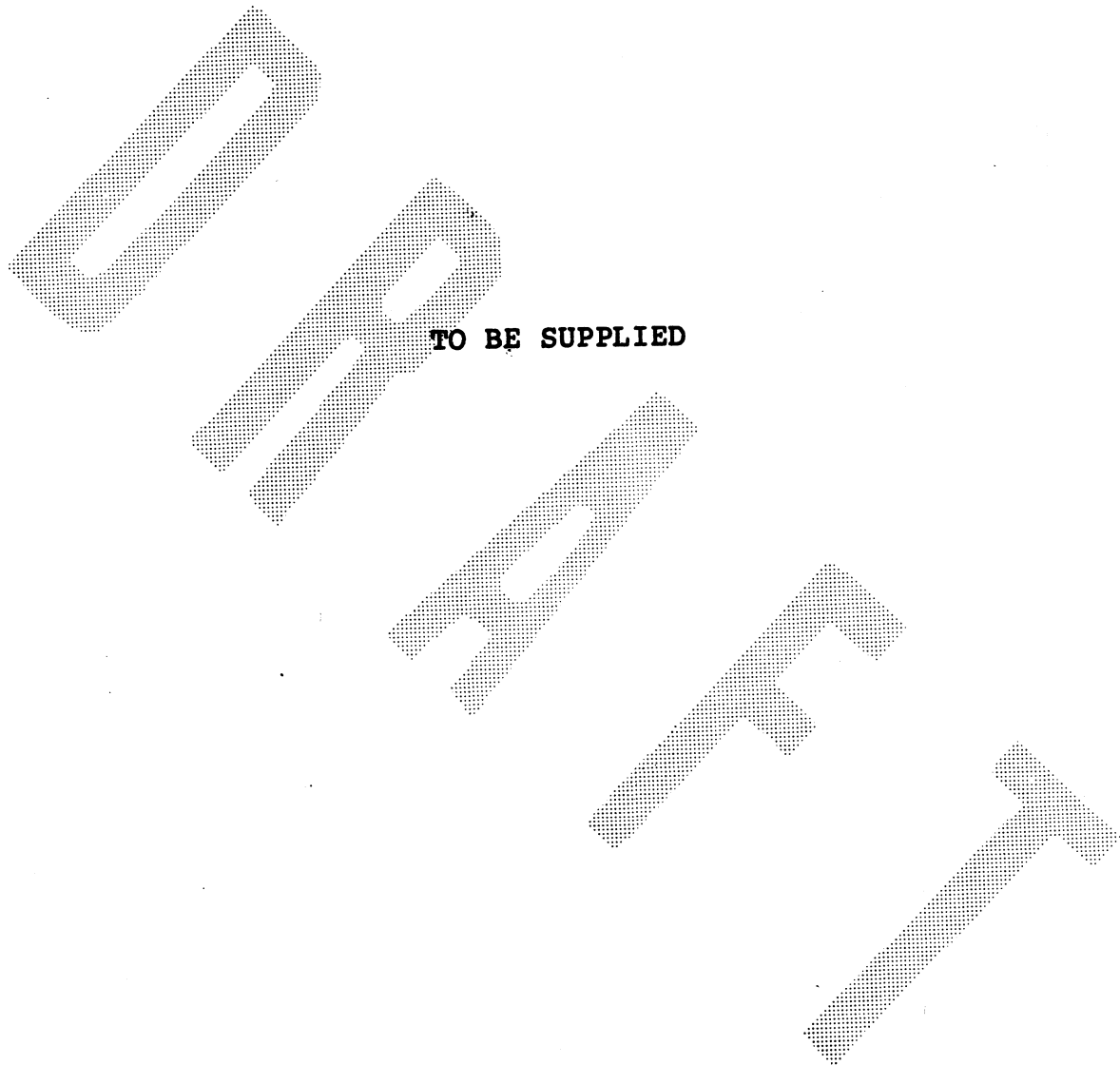
**Temperature Requirements:** The POINT 4 MARK III requires an environment with an adequate temperature control system to maintain a recommended 20 to 30 degrees Celsius. Maximum operating range is between 0 and 50 degrees Celsius. Relative humidity is 10 to 90 percent, noncondensing.

**Enclosure Requirements:** The POINT 4 MARK III is packaged in a 3-slot chassis, measuring 5.25 inches high, 19 inches wide, and 17.5 inches deep. The CPU/Memory PCB, Peripheral Interface Board, and Power Supply Board have the same dimensions: 14.5" x 12".

**2.2 UNPACKING INSTRUCTIONS**

**TO BE SUPPLIED**





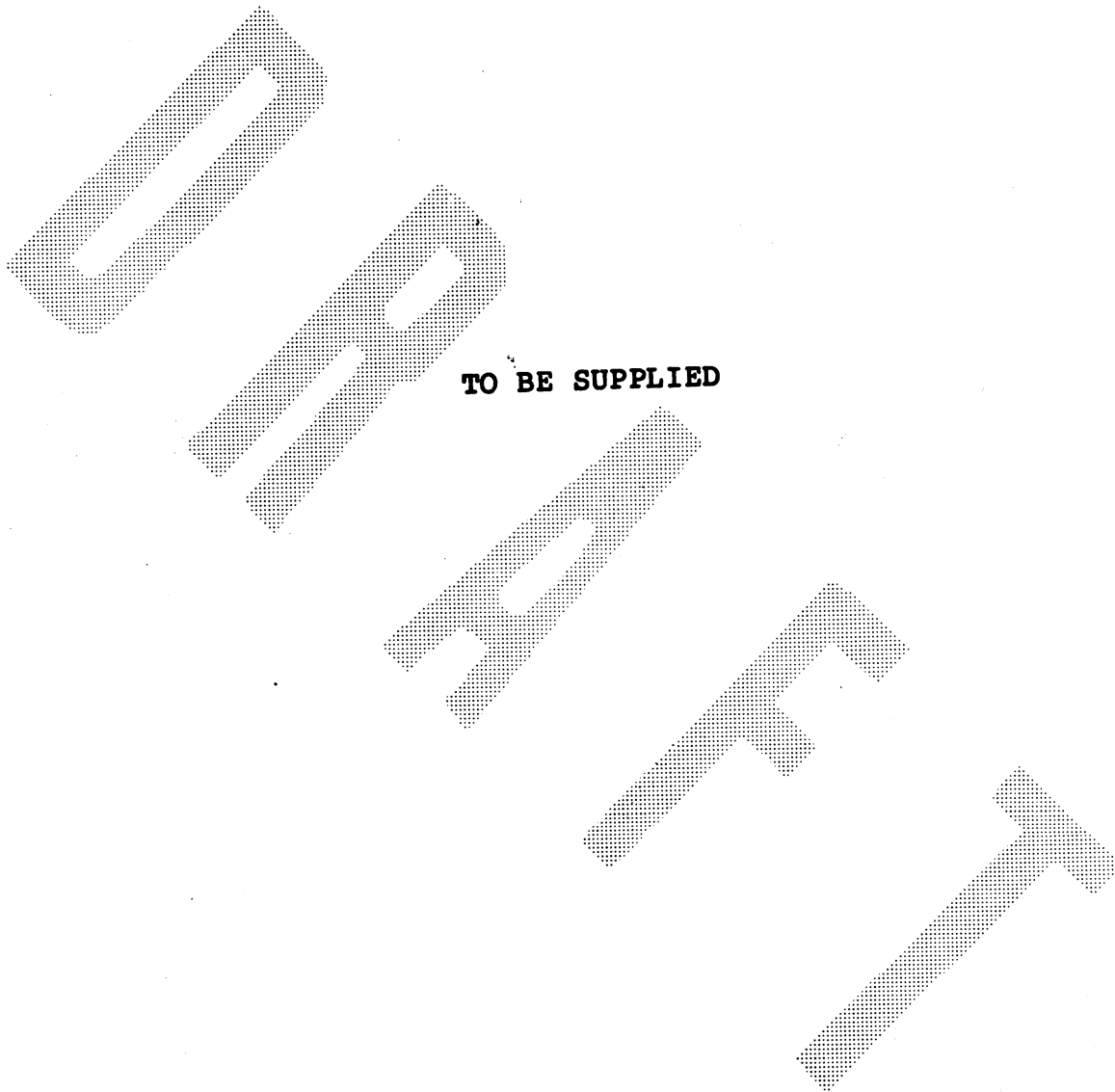
**Figure 2-1. POINT 4 MARK III Processor Chassis Packaging**

### 2.3 ASSEMBLY PROCEDURES

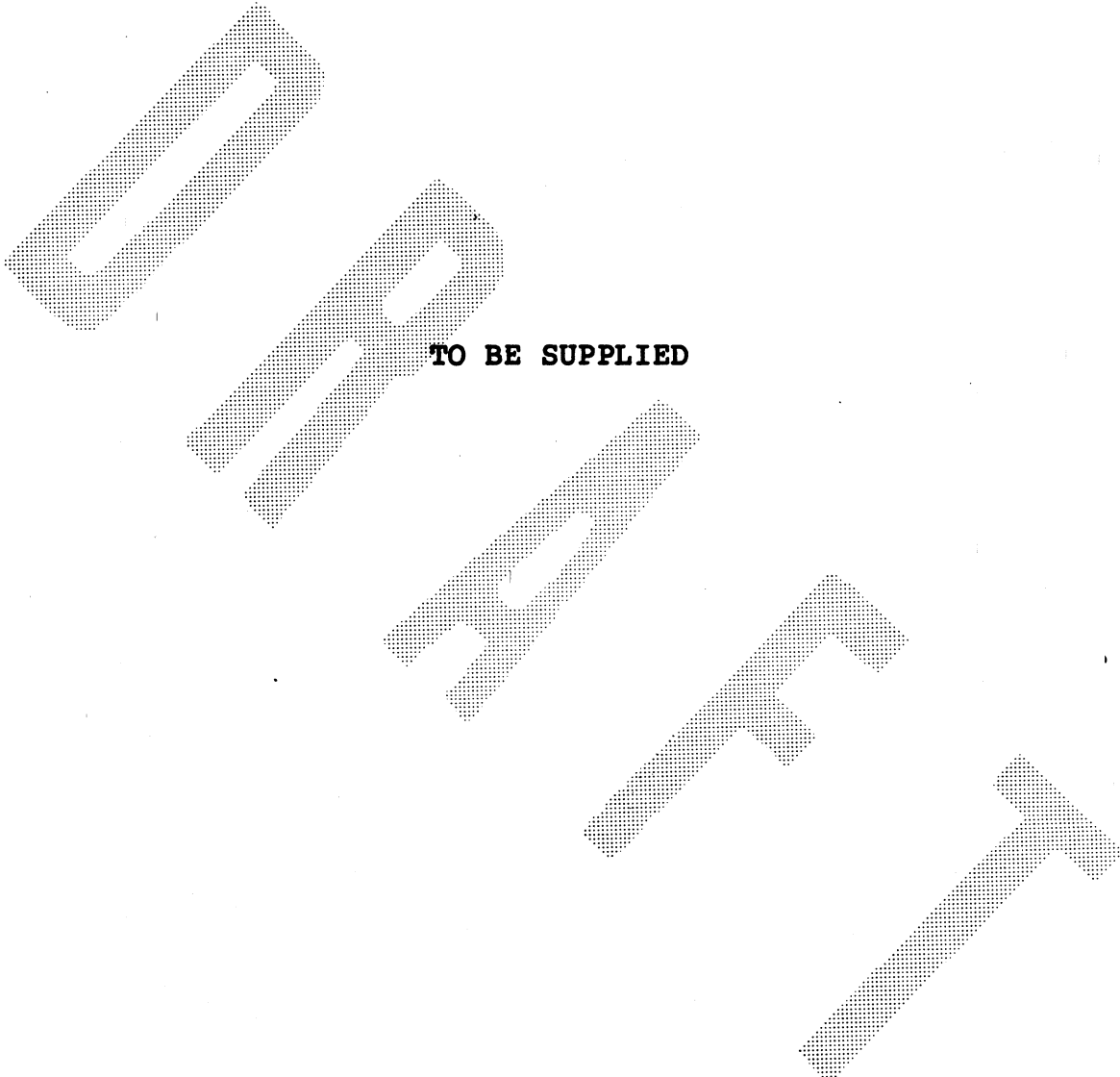
The following assembly procedures are required for MARK III installation.

1. Plug AC cord into back of chassis.
  2. Unscrew the four screws holding the top panel on the back of the chassis. This allows access to the Peripheral Interface Board, second board from the top).
  3. On the PIB, plug CRT cable into connector J5. This is the first white Molex connector to the left of the four connectors.
  4. Pull the white tabs at both corners of the board. Pull out board approximately 2 or 3 inches.
  5. To connect the disc drive:
    - a. Plug into connector J6 (60 pin, left side of PIB), Cable "A", with pin 1 of cable to the right edge.
    - b. Plug into connector J8 (26 pin, behind J6), Cable "B", with pin 1 of cable to the right edge.
    - c. If a second drive is to be connected, the second "B" cable will plug into J7 (to the right of J8).
  6. To connect the streamer tape drive, plug the 50-pin cable into J1 (right side of PIB) with pin 1 to the right edge.
- CAUTION**
- J1 has no cable guides. Make sure cable is not shifted to right or left.
7. To connect a printer, plug its cable into either J3 or J2 (white, 6-pin Molex).
  8. The remaining port may be used for another CRT terminal.
  9. All the MUX ports are strapped for 9600 baud rate. They may be restrapped if another rate is desired.
  10. Push the PIB back into the chassis.
  11. Power is turned on via an ON/OFF switch in the back of the chassis. With power ON, the red light below the switch is illuminated.
  12. Proceed to Powering Up the System, Section 3.4.





**Figure 2-2. Power Supply Chassis and Processor Chassis Mounting Slots**



**Figure 2-3. Typical POINT 4 MARK III Board Configuration**

## Section 3

### OPERATING PROCEDURES

---

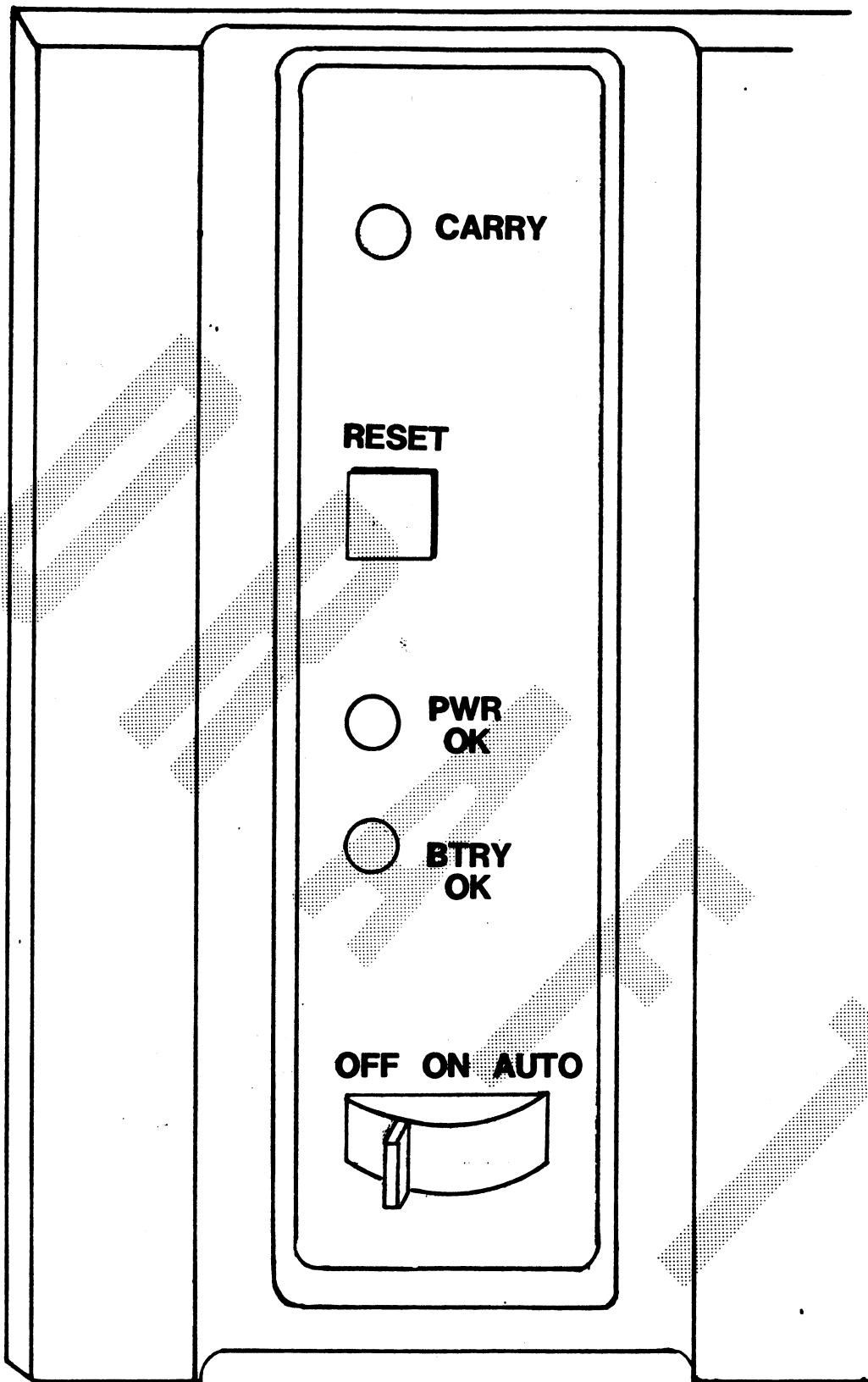
#### 3.1 PROCESSOR MINI-PANEL

There are two types of control units available on the POINT 4 MARK III Computer. These control units are:

- o Processor Mini-Panel
- o Virtual Control Panel

Controls and indicators are outlined and specific procedures for performing common types of operations are given in the following subsections.

The POINT 4 MARK III processor chassis houses essential controls and indicators for basic processor control functions. The controls and indicators for this processor Mini-panel are located on the left-hand side of the chassis (see Figure 1-1). There are three types of operating functions on the Mini-panel: processor monitoring indicators, program execution controls and power controls and indicators. Figure 3-1 is an illustration of the processor Mini-panel controls and indicators.



**Figure 3-1. POINT 4 Processor Mini-Panel**

### 3.1.1 POWER CONTROLS

The Mini-panel contains power controls and indicators. POWER ON is controlled by a three-position switch. Table 3-1 lists the three functions of the power switch.

Also provided is a Light Emitting Diode (LED) indicator, which illuminates to indicate an active state for AC power.

Table 3-2 shows the interpretations of the POWER OK LED.

See Figure 3-1 for positions of the three-position switch and the POWER ON indicator.

**TABLE 3-1. POWER CONTROL SWITCH FUNCTIONS**

Switch Setting	Function
ON	Turns on power to the processor and places the Mini-panel in the Panel-On Mode. In this mode all controls and indicators on the Mini-panel are enabled. One pass of Self-test is executed, the carry light illuminates once and goes off, and the message OK is displayed on the port 0 terminal.
AUTO	Functions the same as ON for POINT 4 MARK III.
OFF	Turns off the power supply and thus all processor functions.

**TABLE 3-2. POWER OK LED INTERPRETATIONS**

POWER OK	Interpretation
OFF	Power supply not connected to AC.
OFF	This condition (if keyswitch is in ON or AUTO) indicates that one of the power supply voltages is out of tolerance.
ON	All power supply voltages are in tolerance and available to the processor chassis.

### 3.1.2 PROCESSOR OPERATION MONITORING

In addition to the power monitoring indicators discussed above, the Mini-panel has an LED indicator for monitoring the carry state of processor operation. This LED functions as follows:

Indicates the current state of the processor carry flag. The LED illuminates when the carry flag is set to a 1.

See Figure 3-1 for location of the carry indicators.

### 3.1.3 PROGRAM EXECUTION CONTROL

A pushbutton switch is available to reset program execution in the processor. This switch is enabled in the Panel-On Mode (power switch set to ON or AUTO position) and disabled in the Panel-Off Mode (power switch set to OFF position). RESET functions as follows:

The RESET switch loads contents of an octal debugger/manipulator/Self-test PROM into top 1000 (octal) words of memory. The debugger/manipulator is used for access to accumulators and memory. Allows examination and deposit of data for operation monitoring and control. It optionally allows loading of system software from disc. See Section 2.5 for debugger/manipulator program commands.

See Figure 3-1 for location of the RESET switch.

### 3.2 VIRTUAL CONTROL PANEL

The POINT 4 MARK III has the ability to do front panel operations plus some system monitoring functions from a master terminal. This feature is designed for use by programmers to debug system problems and to manipulate the contents of registers and memory. The feature is implemented in a system program called MANIP which is loaded into RAM from a PROM when the RESET switch is pressed.

MANIP is a position-independent memory manipulator and debug package. MANIP occupies only 1000 (octal) words of memory.\* All operations are executed by typing one letter followed by octal parameters as required (except colon (:)) which is also preceded by an octal parameter) and ending with a RETURN.

Table 3-3 lists the functions provided by MANIP (the number in parentheses indicates the number of parameters required for that particular function).

**TABLE 3-3. SUMMARY OF MANIP COMMAND FUNCTIONS**

Code	Function	Parameters Required
A	Type initial PC, accumulators and carry flip-flop	(0)
C	Change accumulator or carry flip-flop	(2)
D	Dump (octal, word or byte)	(1 or 2)
J	Jump with accumulators and carry restored	(1 or 2)
K	Store a constant in a block of memory	(3)
M	Move a block in memory	(3)
P	Program load from disc	(0 or 1)
R	Read 3 blocks from CTU	
T	Run Self-test program	(0)
:	Examine or deposit into a specified location	(2)

\*For those who are familiar with POINT 4's IRIS Operating System, MANIP is comparable to DEBUG. The main differences are that MANIP does not have (1) symbolic capability, (2) breakpoints or trace, (3) disc read or write, and (4) Ctrl H/Ctrl A (backspace) capability. MANIP occupies only 1000 (octal) words of memory, while DEBUG occupies 3000 (octal) words of memory.

These functions are described in detail in the following subsections.

MANIP normally occupies the memory locations 77000 through 77777. Location 77000 is reserved for saving the initial value of the program counter (PC), that is, the value of PC where the CPU had halted before MANIP was started. MANIP may be moved at any time by use of its MOVE (M) instruction. The carry light flashes while MANIP is waiting for an input character to be entered. This is a signal that MANIP is active and will respond to input.

If an error is made while entering control information, two choices are available for correcting it.

1. Press ESC (or any other control character except RETURN) to delete the type-in and enable a new type-in.
2. If the error was in entering an octal value, type a few zeros followed by the correct octal number, as MANIP only uses the last six octal digits typed in for the octal word.



### 3.2.1 COMMAND DESCRIPTIONS

A MANIP command consists of a single letter which is the command identifier and parameters which specify addressing modes, memory addresses and data input. All parameters must be entered in octal form. The letters x, y, z, a, m, and n are used on the following pages to represent octal parameters. Press the RETURN key after entering any command. Table 3-4 lists all MANIP commands and their functions.

See Appendix D for a summary chart of MANIP commands.



**TABLE 3-4. MANIP COMMANDS**

Command & Parameters	Definition
A	Causes initial value of PC (program counter) saved in first location of MANIP, contents of accumulators A0, A1, A2, A3, and carry flip-flop as they were at the time MANIP was entered to be typed on the master terminal screen.
Cx,y	<p>Change accumulator or carry flip-flop.</p> <ul style="list-style-type: none"> <li>o If x is 0, 1, 2, or 3, n y is stored as saved value for accumulator x (A0, A1, A2, A3, respectively).</li> <li>o If x is 4, then saved value of the carry flip-flop is set to 0 or 1 according as y is 0 or not.</li> <li>o If x is greater than 4 and an address offset has been established (see F command), x is interpreted as a real address using the offset previously established, and typed out on the master terminal. The y parameter is not used in this case.</li> </ul> <p>o Parameter Description</p> <ul style="list-style-type: none"> <li>x - 1 octal digit 0-7</li> <li>y - 1 octal word</li> </ul>
Dx	<p>Dump memory in octal, beginning at location x, using addressing mode a. Eight words (or bytes if a byte address mode is used) are typed per line, with the address of the first word (byte) at the beginning of each line.</p> <p>o Parameter Description</p> <ul style="list-style-type: none"> <li>x - an octal number representing a 16-bit memory address</li> </ul>
Jx	<p>Jump to location x (using addressing mode a) with accumulators and carry stored.</p> <p>o Parameter Description</p> <ul style="list-style-type: none"> <li>x - an octal number representing 16-bit memory address</li> </ul>
Kx,y,z	<p>Store the octal constant z in locations x through y, inclusive.</p> <p>o Parameter Description</p> <ul style="list-style-type: none"> <li>x - octal number representing 16-bit beginning memory address</li> <li>y - octal number representing 16-bit ending memory address</li> <li>z - octal number representing constant</li> </ul>

TABLE 3-4. MANIP COMMANDS (Cont)

Command & Parameters	Definition
Mx,y,z	<p>Move block in memory. Locations x through y, inclusive, are moved to area starting at location z.</p> <ul style="list-style-type: none"><li>o Source and destination areas may overlap in either direction without bad effects.</li><li>o May be used to move MANIP itself as long as destination area does not overlap source area.</li><li>o Parameter Description<ul style="list-style-type: none"><li>x - octal number representing 16-bit beginning memory address</li><li>y - octal number representing 16-bit ending memory address</li><li>z - octal number representing 16-bit beginning memory address of new location</li></ul></li></ul>
P	<p>Initial Program Load from disc (Sector 0, Surface 0, Cylinder 0). Performs standard bootstrap APL function (i.e., gives an NIOS instruction and then idles at location 377 waiting for the disc to overwrite that location).</p>
R	<p>Read blocks 2, 3 and 4 from the Cassette Tape Unit (CTU) tape. Bootstrap will run automatically. After the R Command has been given, the only valid functions are the Jump (J) Command and/or any CTU command. See Subsection 3.3.2 for description of CTU commands in MANIP for use in CTU mode.</p>
T	<p>Run the Self-test program. Successful completion results in OK being displayed on port 0 terminal. Self-test then moves itself randomly in memory and repeats the above. Main memory will be destroyed. See the POINT 4 MARK III Diagnostics Manual for further details.</p>
x:y	<p>Octal value y is stored at location x, and next cell is opened.</p> <ul style="list-style-type: none"><li>o Parameter Description<ul style="list-style-type: none"><li>x - octal number representing 16-bit memory address</li><li>y - 1 to 6 digits representing an octal value</li></ul></li></ul>

### 3.3 PROCESSOR/CTU INTERFACE

This section describes commands used to transfer stand-alone programs such as diagnostics between the CTU and the POINT 4 MARK III. There are sixteen basic CTU commands which can be enabled from the master terminal. DEBUG (POINT 4's stand-alone debug program) can perform all sixteen commands; MANIP (the Virtual Control Panel program built into POINT 4 MARK III) can perform only a subset of the CTU commands. Section 3.3.2 describes CTU commands enabled in MANIP; Section 3.3.3 describes those enabled in DEBUG.

#### 3.3.1 CTU COMMANDS

A CTU command consists of a single character control code (CTRL and an ASCII character), a block number for the starting block, an additional block count and a RETURN. There are sixteen functions which can be specified by control codes from an ASCII keyboard.

### 3.3.1.1 Command Functions

The control code of a CTU command is a single nonprinting character entered while holding down the CTRL key on the keyboard. The CTU will echo two printable characters, a caret for the control key and the ASCII letter representing the command for ease or command verification. The CTU command functions are listed in Table 3-5.

**TABLE 3-5. CTU COMMAND FUNCTIONS**

Function	Control Code	CTU ASCII Echo
Read Blocks	CTRL R	^R
Write Blocks	CTRL W	^W
Seek Block	CTRL S	^S
Enquiry	CTRL E	^E
Verify Block	CTRL V	^V
Write Buffer	CTRL B	^B
Access Buffer	CTRL A	^A
Fill Buffer	CTRL F	^F
Put in Buffer	CTRL P	^P
List Directory	CTRL D	^D
Open File	CTRL O	^O
Kill File	CTRL K	^K
Rewind Drive	CTRL Z	^Z
Select Track	CTRL T	^T
Initialize Track	CTRL I	^I
Cancel Command	CTRL X	^X

### 3.3.1.2 Command Format

All CTU commands are structured as follows:

COMMAND [BLOCK NO.], [ADD'L BLOCK COUNT] RETURN

#### NOTE

All fields enclosed by brackets are optional fields.

Field functions can be described as follows:

#### COMMAND

This is a one-character control code specifying the function to be performed. See Subsection 3.3.1.1 for a complete listing of control codes and their functions.

#### [BLOCK NO.]

This is an optional DECIMAL block address specification. Zero (0) is the first block address. The maximum block address depends upon tape length (typically 999).

#### NOTE

CTU blocks contain only 128 words; IRIS blocks contain 256 words.

#### [ADDITIONAL BLOCK COUNT]

This is an optional DECIMAL field which specifies the number of blocks to be operated upon in addition to the block specified in the [BLOCK NO.] field. A specification of zero (0) for this field instructs the CTU to operate only on the block specified in the [BLOCK NO.] field. The maximum count is 255.

#### RETURN

An ASCII carriage return character is the execute instruction for the CTU. If the CTU receives a CTRL X before a RETURN, the CTU cancels (does not execute) the preceding command string specified. A new command can follow the RETURN.

### 3.3.1.3 CTU Error Conditions

The CTU will report error conditions by presentation of an error code. An error condition is given in the following format:

BELL, Error Code, BELL, RETURN, Line Feed

The error codes and the errors they represent are shown in Table 3-6.

TABLE 3-6. CTU ERROR CODES

Error	Description
P	A write operation was attempted on a write protected tape.
M	Tape motion failure. This error occurs either as a result of a jam or mechanical malfunction, or as a result of incorrect tape positioning due to operator handling. In case of incorrect tape positioning, the CTU does not know the tape location and thus runs into the stops.
R	Read error. The operator should retry the command. An excessive number of read errors usually indicates noise interference, faulty system ground, a defective tape or a CTU hardware malfunction.
U	Unknown name. An attempt was made to delete a name not found in the directory.
?	Syntax error in command string.
F	Track Directory is full (126) names. An old name must be killed before a new name may be entered.

### 3.3.2 CTU COMMANDS IN MANIP

To enter CTU mode from MANIP use MANIP command R to read the first three blocks from the CTU tape.

MANIP allows reading from CTU into RAM (CPU main memory), but does not allow writing onto tape. To write to the CTU, get a cassette which contains DEBUG, read it into memory by means of MANIP, then Jump into DEBUG and use its CTU write capabilities (see Section 3.3.3).

All MANIP commands consist of a control character (CTRL and an ASCII character), followed optionally by one or more parameters, and terminated by a RETURN. The only exception is CTRL X which cancels any partially entered command immediately. Data is stored on tape in blocks of 256 bytes (128 words) each. Table 3-7 lists the CTU commands used in MANIP. All numeric parameters (x, y below) are in DECIMAL, origin 0.

**TABLE 3-7. CTU COMMANDS IN MANIP**

Control Character	Description
CTRL D	List directory (index) from tape, if tape is so formatted.
CTRL E	Enquire (error status).
CTRL Ofile	Open the named file, if it is in the directory.
CTRL R	Read the open file from tape into memory.
CTRL Rx,y	Read from tape into memory; read y+1 blocks starting at block x.
CTRL Sx	Seek to block x on tape. CTRL S999 will wind the tape all the way forward.
CTRL Tn	Select track n (0 to 1).
CTRL X	Cancel partially entered command.
CTRL Z	Rewind tape to starting position.



## NOTE

ESC exits CTU mode and reverts to normal MANIP commands, but does not cancel any partial command that may already have been transmitted to the CTU. Use CTRL X to cancel a partial command.

The Read command transfers data into memory starting at address 0. To start the transfer at some other address, precede the CTU command with:

Memory address (octal), colon, RETURN

MANIP will then display the content of the chosen location, followed by a colon. This allows examination of the word before starting the tape transfer. Then type the CTRL R, followed by its parameters (if any) and a RETURN.

### 3.3.3 CTU COMMANDS IN DEBUG

DEBUG is a position-independent debug package of the IRIS Operating System. When using the Virtual Control Panel on the POINT 4 Computer it is necessary to use DEBUG commands to write to the CTU. The write procedure from MANIP requires a cassette containing DEBUG which must be read into memory using MANIP. A jump into DEBUG allows use of DEBUG CTU commands to write to the CTU. CTU commands in DEBUG may also be used in other CTU transfer procedures.

All CTU commands consist of a control character, followed optionally by one or more parameters, and terminated by a RETURN. The only exception is CTRL X which cancels any partially entered command immediately. Data is stored on tape in blocks of 256 bytes (128 words) each. Table 3-8 lists the CTU commands used in DEBUG. All numeric parameters (x, y below) are in DECIMAL, origin 0.

**TABLE 3-8. CTU COMMANDS IN DEBUG**

Control Character/ Parameters	Description
CTRL Ax,y	Access CTU buffer, i.e., transfer buffer into memory. Transfers y bytes, starting at byte x. Default = 256 bytes starting at byte 0.
CTRL Bx	Write CTU buffer onto tape, at block x.
CTRL D	List directory (index) from tape, if tape is so formatted.
CTRL E	Enquire (error status).
CTRL F	Fill CTU buffer from memory (128 words).
CTRL Ix	Initialize (format) selected track to x+1 blocks of 128 words each. Maximum = 1999 for 1000 blocks.
CTRL Kfile	Kill the named file, i.e., erase its name from the directory.
CTRL Ofile	Open the named file, if it is in the directory.

**TABLE 3-8. CTU COMMANDS IN DEBUG (Cont)**

Control Character/ Parameters	Description
CTRL Ofile,x,y	Create a directory entry for the named file (max. 5 char.), starting at block x and containing y+1 blocks of 128 words each.
CTRL Px,y	Put into CTU buffer from memory, transferring y bytes beginning at byte x in the buffer. Default = 256 bytes starting at byte 0.
CTRL R	Read the open file from tape into memory.
CTRL Rx,y	Read from tape into memory; read y+1 blocks starting at block x.
CTRL Sx	Seek to block x on tape.
CTRL Tn	Select track n (0 or 1).
CTRL V	Verify; i.e., read from tape into CTU buffer, checking checksum.
CTRL W	Write from memory onto tape into the open file, if any.
CTRL Wx,y	Write from memory onto tape, writing y+1 blocks starting at block x.
CTRL X	Cancel partially entered command.
CTRL Z	Rewind tape to starting position.

**NOTE**

ESC exits CTU mode and reverts to normal DEBUG commands, but does not cancel any partial command that may already have been transmitted to the CTU. Use CTRL X to cancel that.

All commands that transfer data into or out of main memory default to an initial address of 0. To start the transfer at some other address, precede the CTU command with:

Memory address (octal), colon, RETURN

DEBUG will then display the content of the chosen location, followed by a colon. This allows examination of the word before starting the tape transfer. Then type the CTU control character (e.g., CTRL R or CTRL W), followed by its parameters and a RETURN.

Table 3-9 is a quick-reference guide to the commands used for data transfer from a source to a destination.

**TABLE 3-9. SUMMARY AND OVERVIEW OF DATA TRANSFER COMMANDS**

Source	Destination	Command
Tape	Memory	^R
Memory	Tape	^W
Tape	Buffer	^V
Buffer	Tape	^B
Buffer	Memory	^A
Memory	Buffer	^F complete buffer ^P selected byte(s) only

### 3.4 POWERING UP THE SYSTEM

The following steps should be followed when first applying power to the POINT 4 MARK III power supply and processor:

1. Turn the power-control switch on the front Mini-Panel to ON.
2. The carry light comes on for approximately 1.5 seconds, goes off for .5 seconds, comes on again and remains illuminated.
3. The Master CRT displays OK, indicating that the system has run a successful CPU self-test program.
4. If the disc drive is in a Ready state, the CRT displays PRESS RETURN. Upon entry of a RETURN, the system loads the program from the disc drive.
5. If the disc drive is not in a Ready state, press the RESET button on the Mini-Panel, which accesses the MANIP program. The CRT displays the contents of the program counter and accumulators. Press P on the CRT keyboard, followed by a RETURN.
6. The CRT displays PRESS RETURN. Upon entry of a RETURN the system loads the program from the disc drive.

## **3.5 DIAGNOSTIC CHECKS**

### **3.5.1 DIAGNOSTIC CAPABILITIES**

The POINT 4 MARK III has a comprehensive built-in diagnostic program, contained in a PROM (Programmable Read-Only Memory).

The Self-Test diagnostic contains the following tests:

1. Compare Instruction Test
2. ALU and Data Bus Test
3. ALU Source Operand Test
4. Exhaustive ALU Instruction Test
5. Page 0 and Base 3 Addressing Modes Test
6. Relative, Base 2, and Indirect Addressing Modes Test
7. Limited I/O Instruction Test
8. Worst-Case Memory Test of all Memory Locations

### **3.5.2 SELF-TEST OPERATING PROCEDURES**

General procedures for initiating the POINT 4 MARK III Self-Test follow. For details on tests executed, expected results, and error interpretation, see the POINT 4 MARK III Diagnostics Manual.

A T (Test) Command, followed by a RETURN initiates the MARK III Self-Test program. Successful completion results in OK displayed on the Master Terminal. Thereafter, each time the Self-Test repeats, an OK is displayed. The Self-Test program continues until RESET is pressed.

### **3.5.3 SELF-TEST ERRORS**

A Self-Test error is indicated by a continuously flashing carry light. Refer to the POINT 4 MARK III Diagnostic Manual for detailed information on diagnostic programs, program listings, and error interpretation.

## Section 4

### INPUT/OUTPUT INTERFACES

---

#### 4.1 INPUT/OUTPUT BUS INTERFACE SIGNALS

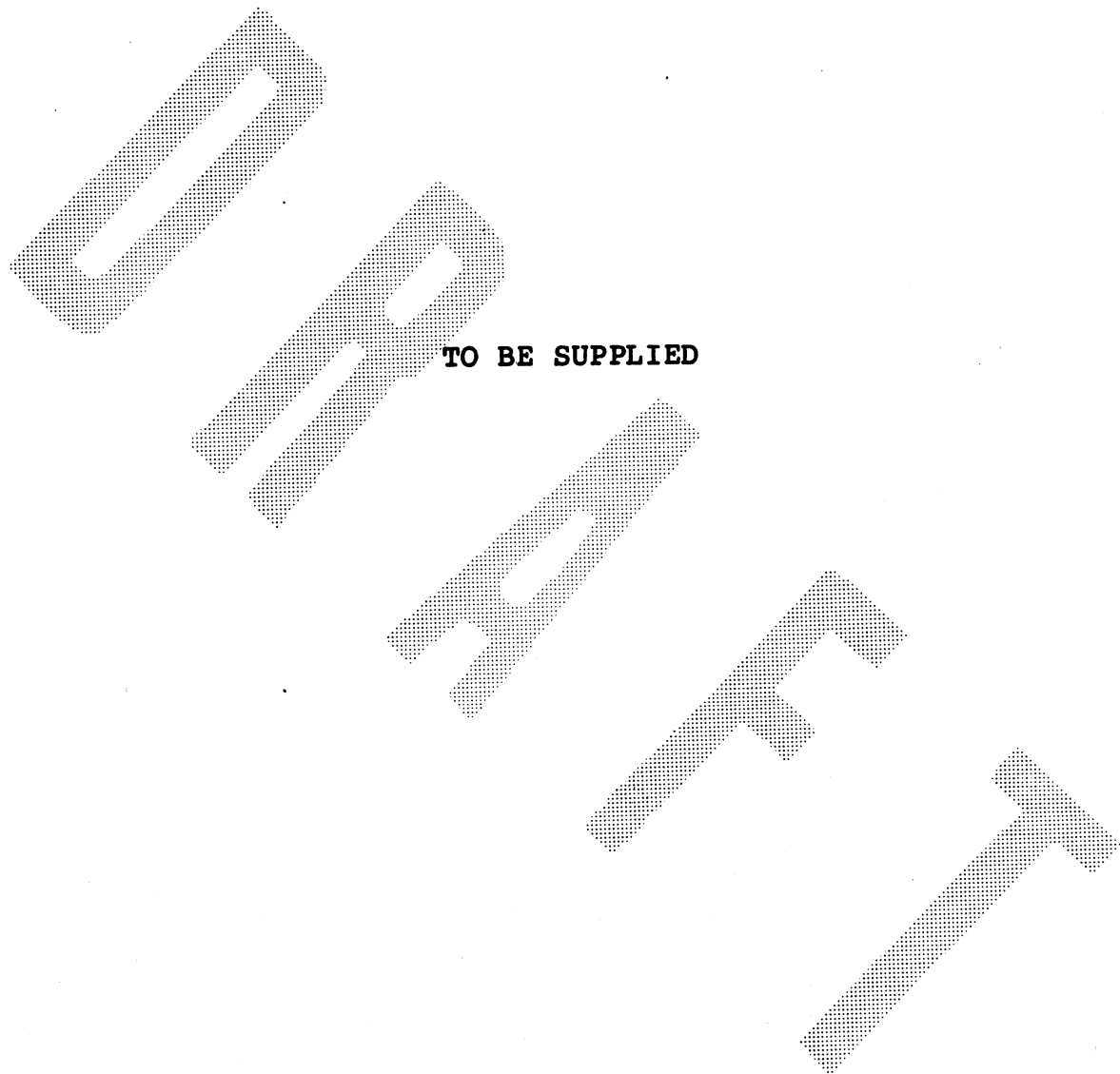
Input/Output Bus signals connect the processor logic to peripheral device logic. The logic for programmed I/O transfers and data channel transfers forms the interface between the processor Main Data Bus and the peripheral device controller logic. Logic to implement I/O transfer instructions is present in all device controllers. Data channel transfer logic is present only in those controllers that control devices using the data channel. Device-end control logic for these functions may vary widely, depending on the requirements of the particular device. This subsection describes the POINT 4 MARK III I/O bus and control signals, including any special signals used for disc or tape controller interface.

##### 4.1.1 INPUT/OUTPUT INTERFACE SIGNALS

Signals on the Input/Output Bus can be grouped into the following signal classifications:

- a. Bidirectional Data Bus (16 lines): Used for transfer of all data and address words between the CPU and a peripheral device, for both programmed I/O and data channel transfers.
- b. Device Codes (6 lines): Codes used to designate the peripheral device involved in an input/output instruction.
- c. Device Control Signals (2 lines): These signals are generated by the CPU in response to input/output instructions, and are used to initialize and control I/O devices.
- d. Interrupt Input Signals (3 lines): Signals used to control which controller (multiplexer disc or tape) will be serviced.
- e. Disc Status Flags (5 lines): Signals used to carry disc status information to the processor.

Figure 4-1 is a diagram of I/O signals across the I/O Bus. Table 4-1 divides these signals by signal classifications, designates the signal name and defines each signal function.



**Figure 4-1. Input/Output Signals**



**TABLE 4-1. INPUT/OUTPUT SIGNALS BY CLASSIFICATION**

Signal Group	Signal*	Direction	Description
Data Bus	DB00 to DB015	Bidirectional	All data and addresses are supplied to and from the device via these lines. DB00+ is the MSB.
Device	DS0- to DS5-	From CPU	The CPU places the device code (bits 10-15 of the instruction word) on these lines during the execution of an input/output instruction. DS0- is the MSB.
Device Control Signals	IOBRST-	From CPU	Input/Output Reset. Generated when APL is pressed on the Mini-panel, when an IOBRST instruction is being executed, and during power turn-on.
	IOBPLS-	From CPU	Generated when CTRL field of an input/output transfer instruction contains code 11. The effect, if any, depends on the device.
	BW-/R	From CPU	Set by firmware to control the direction of data and control transfer based on the instruction issued for that operation.
Interrupt Input	ATNMUX-	From Device	Used by multiplexer to signal its need for interrupt servicing.
	ATNDISC-	From Device	Used by the disc controller to signal its need for interrupt servicing.
	ATNTPE-	From Device	Used by the disc controller to signal its need for interrupt servicing.
Disc Status Flags	FLAG0 to FLAG4	From Device	Used by firmware to transfer status information.

\*All signal names ending with "-" are active low. All others are active high.

#### 4.1.2 BACKPLANE PIN SIGNAL CONNECTORS

All signal connections between the processor and each controller take place via two 100-pin backplane connectors. Figure 4-2 shows the connector pin layout for all I/O signals. The labelled pins refer to the I/O control signals, data transfer signals and the power lines used by peripheral controllers.



Bottom		A	Top	Bottom		B	Top
GND	1	2	GND	GND	1	2	GND
GND	3	4	GND	GND	3	4	GND
+5V	5	6	+5V	+5V	5	6	+5V
+5V	7	8	+5V	+5V	7	8	+5V
DS2-	9	10	DS1-	DS2-	9	10	DS1-
DS0-	11	12	DS5-	DS0-	11	12	DS5-
DSY-	13	14	DS3-	DSY-	13	14	DS3-
BS-/R+	15	16	DB15+	BS-/R+	15	16	DB15+
DB14+	17	18	DB13+	DB14+	17	18	DB13+
DB12+	19	20	DB11+	DB12+	19	20	DB11+
DB10+	21	22	DB09+	DB10+	21	22	DB09+
DB08+	23	24	DB07+	DB08+	23	24	DB07+
DB06+	25	26	DB05+	DB06+	25	26	DB05+
DB04+	27	28	DB03+	DB04+	27	28	DB03+
DB02+	29	30	DB01+	DB02+	29	30	DB01+
DB00+	31	32	GND	DB00+	31	32	GND
GND	33	34	IOBRST-	GND	33	34	IOBRST-
GND	35	36	GND	GND	35	36	GND
IOBPLS-	37	38	GND	IOBPLS-	37	38	GND
GND	39	40	FLAG4-	GND	39	40	FLAG4-
FLAG3-	41	42	FLAG2-	FLAG3-	41	42	FLAG2-
FLAG1-	43	44	FLAG0-	FLAG1-	43	44	FLAG0-
ATNDISC-	45	46	ATNMUX-	ATNDISC-	45	46	ATNMUX-
ATNTPL	47	48	GND	ATNTPL	47	48	GND
GND	49	50		GND	49	50	
	51	52			51	52	
	53	54			53	54	
	55	56	GND		55	56	GND
GND	57	58	AUTO-	GND	57	58	AUTO-
CL-	59	60	GND	CL-	59	60	GND
GND	61	62	PWRGON-	GND	61	62	PWRGON-
PWRF-	63	64	GND	PWRF-	63	64	GND
GND	65	66	XRESET-	GND	65	66	XRESET-
XRESET+	67	68	-5V	XRESET+	67	68	-5V
-5V	69	70	-5V	-5V	69	70	-5V
GND	71	72	GND	GND	71	72	GND
-12V	73	74	-12V	-12V	73	74	-12V
-12V	75	76	GND	-12V	75	76	GND
GND	77	78	+12V	GND	77	78	+12V
+12V	79	80	GND	+12V	79	80	GND
GND	81	82	+12VBU	GND	81	82	+12VBU
+12VBU	83	84	GND	+12VBU	83	84	GND
GND	85	86	-5VBU	GND	85	86	-5VBU
-5VBU	87	88	GND	-5VBU	87	88	GND
GND	89	90	+5VBU	GND	89	90	+5VBU
+5VBU	91	92	+5VBU	+5VBU	91	92	+5VBU
GND	93	94	GND	GND	93	94	GND
+5V	95	96	+5V	+5V	95	96	+5V
+5V	97	98	+5V	+5V	97	98	+5V
FRM GND	99	100	FRM GND	FRM GND	99	100	FRM GND

Figure 4-2. Backplane I/O Signals

0

0

## SECTION 5

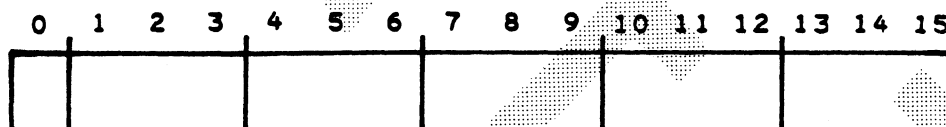
### INSTRUCTION REPERTOIRE

#### 5.1 INTRODUCTION

This section explains the function and use of POINT 4 MARK III instructions. Included is a discussion of two's-complement notation, addressing modes, and the individual instructions in the memory reference, arithmetic/logic, and input/output instruction groups. Input/output instructions and interrupt handling instructions are presented, with details given for special code-77 (CPU) instructions.

#### 5.2 OCTAL REPRESENTATION AND TWO'S COMPLEMENT NOTATION

The computer uses 16-bit binary words for program instructions and data. The bits are numbered 0 through 15 with bit 0 the most significant bit (MSB) and bit 15 the least significant bit (LSB). For convenience, binary words are represented in 6-digit octal form. Each octal digit represents three bits and can have values between 0 and 7, except the most significant digit which represents a single bit and has a maximum value of 1.



POINT 4 16-Bit Binary Word Format

The reader is presumed to be familiar with binary and octal notations. For a simple review, the following example shows the correspondence between decimal, binary and octal representation:

<u>Decimal</u>	<u>Binary</u>	<u>Octal</u>
0	0000000000000000	000000
1	0000000000000001	000001
2	0000000000000010	000002
8	0000000000001000	000010
64	0000000010000000	000100
5407	0001010100011111	012437
32,767	0111111111111111	077777 (15 bit max.)
65,535	1111111111111111	177777 (16 bit max.)

The computer represents negative numbers in two's-complement form. Signed positive and negative numbers are used both as 16-bit operands and as 8-bit address displacements in memory reference instructions. Therefore, a review of two's complement arithmetic is necessary.

In two's-complement arithmetic, positive and negative values are distinguished by a 0 or 1 in the leftmost bit position (sign bit). Positive numbers have a sign bit of 0, with the numerical value expressed in ordinary binary form by the remaining bits. Negative numbers have a sign bit value of 1 and the numerical value expressed in two's-complement form. The two's complement is found by taking the one's complement or logical complement of the number including the sign bit (changing all 0's to 1's and all 1's to 0's) and adding 1.

The number zero is represented by 0's in all bit positions. There is only one representation for zero, since the two's complement of zero is also zero. Zero is a nonnegative value. For this reason also, there is one more negative number than there are nonzero positive numbers.

The range of signed, 8-bit fields is as follows:

	<u>Binary Representation</u>			<u>Octal Value</u>
largest positive	01	111	111	+177
	01	111	110	+176
		...		...
	00	000	001	+1
	00	000	000	0
	11	111	111	-1
	11	111	110	-2
		...		...
most negative	10	000	001	-177
	10	000	000	-200

### 5.3 INSTRUCTION TYPES

From the programmer's point of view, the POINT 4 Computer is comprised of four accumulators, 64K words of memory and an input/output bus. The instructions control and manipulate the data flowing between these elements.

All instruction words can be classified into one of the following three categories:

1. Memory Reference Instructions (instructions that reference a memory location). These include:

- LDA - Load an accumulator from memory
- STA - Store an accumulator into memory
- JMP - Jump to another location in memory
- JSR - Jump to a subroutine in memory
- ISZ - Increment memory and skip if zero
- DSZ - Decrement memory and skip if zero

2. Arithmetic/Logic Instructions (instructions that specify a particular arithmetic or logic operation to be performed on one or two operands stored in the accumulators, and allow for testing the result for skip conditions).

3. Input/Output Instructions (instructions for input/output operations with a specific peripheral device).

Figure 5-1 is an overview of the formats for each type of instruction. Each of these three classes is discussed in detail in the succeeding subsections.

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
MEM. REF.	JMP	0	0	0	0	0	INDIRECT	INDEX	DISPLACEMENT								
	JSR	0	0	0	0	1											
	ISZ	0	0	0	1	0											
	DSZ	0	0	0	1	1											
	LDA	0	0	1	AC												
	STA	0	1	0	AC												
I/O		0	1	1	AC	OPCODE		CTRL	DEVICE CODE								
A/L		1	ACS		AC	OPCODE		SH	CY	NL	SK						

AC = Accumulator  
 CTRL = Control pulse  
 ACS = Source accumulator  
 ACD = Destination accumulator  
 SH = Shift control  
 CY = Carry preselection  
 NL = No-load  
 SK = Skip condition

Figure 5-1. POINT 4 Instruction Format Summary



## 5.4 MEMORY REFERENCE INSTRUCTIONS

Six memory reference instructions are used to move data between memory locations and accumulators, to transfer program control to a new location, and to modify and test memory words. The memory reference instructions fall into three general categories, as follows:

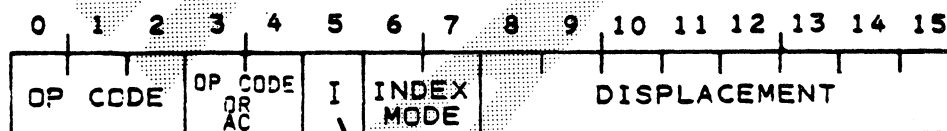
1. Move Data Instructions: LDA, STA
2. Jump Instructions: JMP, JSR
3. Modify Memory Instructions: ISZ, DSZ

Before describing the function of each instruction in this group it is necessary to describe the way in which they address memory.

### 5.4.1 MEMORY ADDRESSING

Each memory reference instruction uses one of several addressing modes to determine an effective memory address, E. The processor accesses the location specified by the effective memory address and uses the contents as the operand of the instruction.

All memory reference instructions use the same binary format:



I - INDIRECT

#### Memory Reference Instruction Format

Bit 5 of the instruction word is the indirect, or I field; bits 6 and 7 are the index, or X field, and bits 8-15 are the displacement, or D field.

- o Since the MARK III does not support indirect addressing, the I field is ignored and the X and D fields specify the effective address (E).
- o The X field defines one of four addressing modes. Each addressing mode may be thought of as a "page" of 256 words which the instruction can address directly.
- o The D field specifies the specific word addressed on the selected page.

All addresses are entered into the Effective Address Register. When this register contains the effective address, E, the instruction specified in bits 0 through 4 of the command is executed.

### 5.4.1.1 Indexing Mode

The X field selects one of the indexing modes listed in Table 5-1.

TABLE 5-1. INDEXING MODES

Bits 6 & 7	Definition
00	<b>Page Zero:</b> Page zero is defined as the first 256 memory locations (addresses in the range from 000000 to 000377 octal). The effective memory address in page zero addressing is equal to the value of the D field, which is an unsigned binary integer that can have values from 000 octal to 377 octal.
01	<b>Relative Addressing:</b> In the relative addressing mode, the address placed in the Effective Address Register is equal to the address in the Program Counter (PC), plus the value of the displacement in the D field. In this case, the displacement, D is a signed binary integer. Bit 8 is the sign (0 = positive, 1 = negative), and the integer may have any value in the range from -200 to +177 octal (decimal -128 to +127). The address in PC can be visualized as the center of a 256-word page, and any address between the bottom (128 words below the PC) and top (127 words above PC) of the page can be specified by the displacement, D.
10 or 11	<b>Base Register Addressing:</b> In the base register addressing mode the address placed in the Effective Address Register is equal to the address in accumulator register A2 (code 10) or A3 (code 11), plus the value of the displacement in the D field. In this case, the displacement, D is a signed binary integer. Bit 8 is the sign (0 = positive, 1 = negative), and the integer may have any value in the range from -200 octal to +177 octal (decimal -128 to +127). The address in A2 or A3 can be visualized as the center of a 256-word page, and any address between the bottom (128 words below A2 or A3) and top (127 words above A2 or A3) of the page can be specified by the displacement, D.

### 5.4.1.2 Indirect Addressing Operations

When the I field (bit 5) of the Memory Reference Instruction contains a 1, an indirect addressing sequence is required. In this case, the address in the Effective Address Register (determined by the X and D fields) is the memory address from which the effective address is to be fetched.

## 5.4.2 TYPES OF MEMORY REFERENCE INSTRUCTIONS

When the Effective Address Register contains the effective address, E, one of two groups of memory reference instructions is performed, as determined by the operation codes. Refer to Section 5.4.1 for basic memory reference instruction formats and field definitions. Refer to Appendix A, Von Neumann Map of POINT 4 Command Structure, for octal formats of each instruction and to Appendix B, POINT 4 InstructionReference Chart, for octal to symbolic conversion of memory reference instructions.

### 5.4.2.1 Move Data Instructions

When the effective address, E, is in the Effective Address Register, one of two operations is performed, depending on the code in bits 1 and 2 of the OPCODE field. Table 5-2 lists these functions.

**TABLE 5-2. MOVE DATA INSTRUCTIONS**

Bits 1 & 2	OPCODE	Definition
01	LDA	Load Accumulator Instruction: The contents of memory location E are stored in the accumulator specified by the AC field (bits 3 and 4). The contents of E are unaffected, the original contents of the accumulator are lost.
10	STA	Store Accumulator Instruction: The data in the accumulator specified by the AC field is transferred to memory location E. The contents of the accumulator are unaffected, the original contents of E are lost.

### 5.4.2.2 Jump and Modify Memory Instructions

When the effective address, E, is in the Effective Address Register, one of four operations is performed if bits 1 and 2 are both 0. The operation is determined by the code in bits 3 and 4 of the OPCODE field, as shown in Table 5-3.

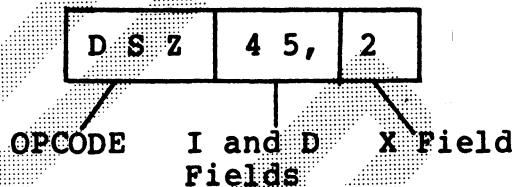
TABLE 5-3. JUMP AND MODIFY MEMORY INSTRUCTIONS

Bits 1-4	OPCODE	Definition
0000	JMP	Jump Instruction: The effective address, E, is transferred from the Effective Address Register to the Program Counter (PC). The next instruction is then fetched from jump address E, and sequential execution is continued from there.
0000	JSR	Jump to Subroutine Instruction: After the effective address, E, has been calculated the address in PC is incremented and the incremented value is stored in accumulator A3. Then, the effective address, E, is transferred from the Effective Address Register to the Program Counter (PC). The next instruction is then fetched from jump address E. Execution of another JMP or JSR instruction that specifies A3 will cause the program to return to the address in A3, plus or minus any desired displacement, D.
0010	ISZ	Increment and Skip if Zero: The contents of effective address E are fetched, incremented, and written back into address E. If the incremented value is equal to zero, PC is incremented by one to skip the next instruction.
0011	DSZ	Decrement and Skip if Zero: The contents of effective address E are fetched, decremented, and written back into address E. If the decremented value is equal to zero, PC is incremented by one to skip the next instruction.

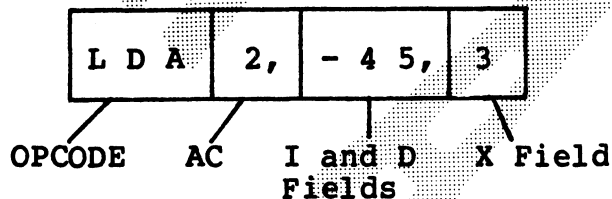
### 5.4.2.3 Assembler Language Conventions and Addressing Examples

The assembler language memory reference instruction consists of the instruction OPCODE mnemonic (STA, LDA, JMP, etc.) followed by symbols that specify the accumulator, the addressing mode and the memory address. The assembler program translates these statements into binary code, which the processor executes. Table 5-1 shows the programming conventions for memory reference instructions.

The format for modify memory and jump instructions requires the instruction mnemonic, and a memory address (including the indirect addressing indicator, the displacement and the indexing indicator). The assembly language instruction will be formatted as follows:



The move data instructions, LDA and STA, also require that an accumulator (A0 - A3) be specified. For example:



Fields that are not specified will be assembled containing 0's. An "@" symbol denotes indirect addressing and places a 1 in bit 5 of the instruction. For example:

```
LDA 1, @ 20
```

specifies indirect, page-zero (X Field = 00) addressing.

Relative addressing is formatted as follows:

```
LDA 0, . + 15
```

The symbol "." indicates X = 01 (relative addressing) and thus "." represents the current value of the program counter.

**TABLE 5-4. ASSEMBLER LANGUAGE CONVENTIONS  
FOR MEMORY REFERENCE INSTRUCTIONS**

Instruction Function	OP CODE Mnemonic	Separator Space or Tab	Accumulator Number	Memory Address		
				I (Indirect)	D (Displ.)	X (Index)
Load Accumulator	LDA		a*			blank
Store Accumulator	STA			blank		1**
Jump	JMP			@	Displacement	2
Jump Subroutine	JSR		None			3
Increment and Skip if Zero	ISZ					
Decrement and Skip if Zero	DSZ					

\* a = 0, 1, 2, 3 representing A0, A1, A2, A3

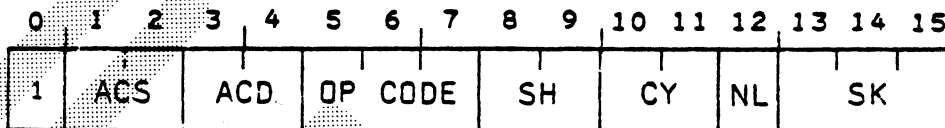
\*\* Instead of "displacement,1" the following sequence may be used: ".+ displacement"

## 5.5 ARITHMETIC AND LOGIC INSTRUCTION GROUP

The eight arithmetic/logic instructions perform binary addition, subtraction, and logical functions on 16-bit operands. These instructions are:

- o Arithmetic: ADD, ADC, INC, SUB, NEG
- o Logic: MOV, COM, AND

All Arithmetic and Logic instructions contain a 1 in bit 0 and have their basic ALU function specified by bits 5-7, as shown below:



**Arithmetic/Logic Instruction Format**

### 5.5.1 ARITHMETIC AND LOGIC PROCESSING

Before describing the eight arithmetic and logic instructions and their auxiliary control fields, it is necessary to describe the organization of the arithmetic/logic processing unit. From the programmer's point of view, the arithmetic/logic processing subsystem is organized as shown in Figure 5-2 and as described in the following subsections.



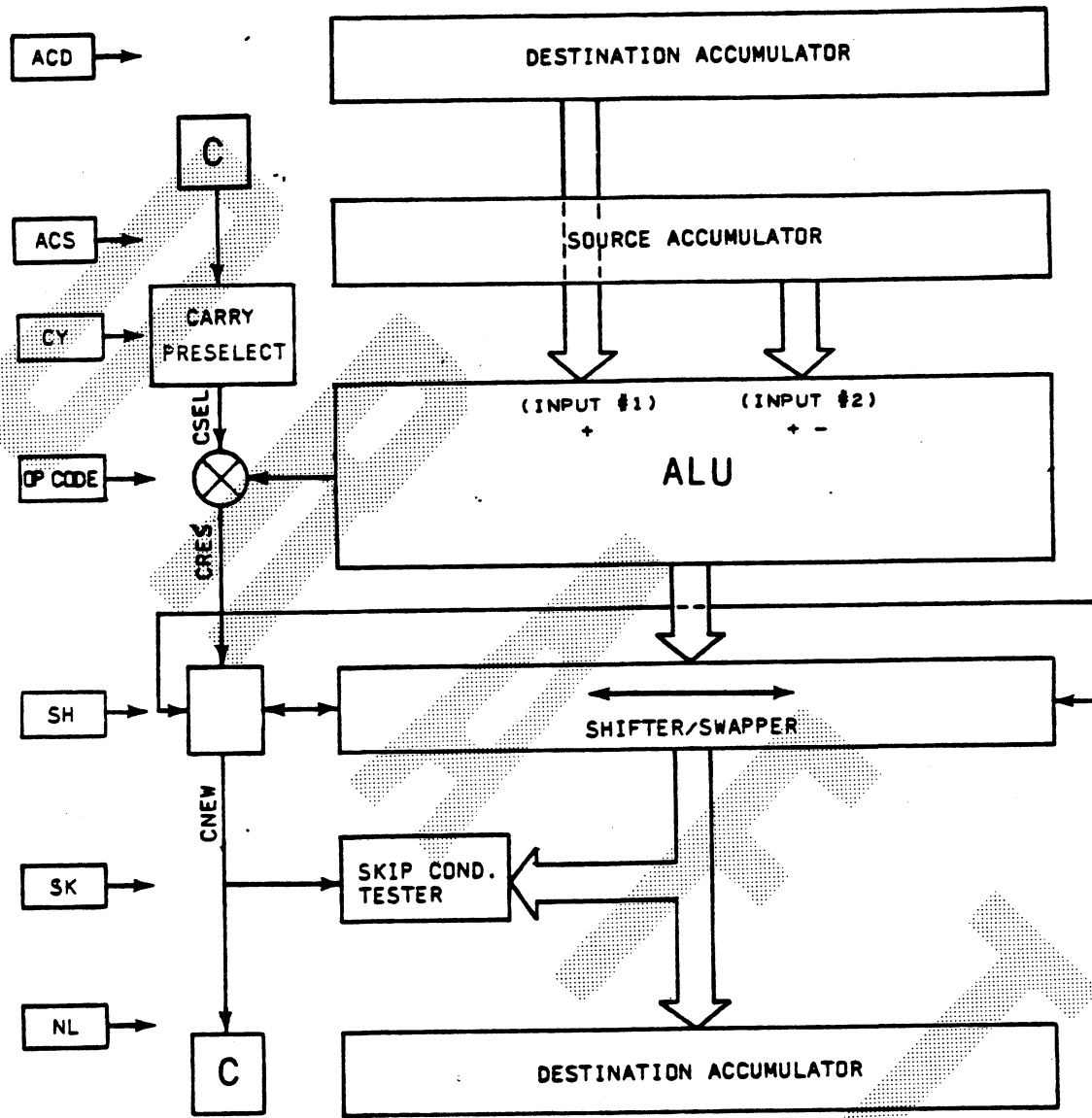


Figure 5-2. Arithmetic/Logic Operations

### 5.5.1.1 Arithmetic/Logic Operations

The heart of the subsystem is the Arithmetic/Logic Unit (ALU) which performs the actual addition, subtraction, or logical operation. It has provision for two inputs:

Input 1: Comes from the accumulator selected by the ACD field and is used only in the operations which require two operands (ADD, SUB, ADC, and AND).

Input 2: Comes from the accumulator selected by the ACS field and is used in all operations.

The ALU performs the arithmetic or logic operation specified by the OPCODE field (bits 5-7). The result of this operation may cause a carry-out to occur from the most significant bit of the ALU. In the case of an operation which adds unsigned integers, a carry-out is equivalent to overflow; however, this is not always true. See Section 5.5.1.2 for a more complete discussion of carry and overflow operation.

If the result of the arithmetic or logic operation involves a carry-out (COUT), the carry preselected by the CY field of the instruction (CSEL) is complemented. The resulting carry (CRES), together with the 16-bit operation result generated by the ALU, is applied as a 17-bit operand to the Shifter, where a shift-left, shift-right or swap may occur as determined by the SH field of the instruction. After shifting, the carry (CNEW) and the 16-bit operation result are loaded into the Carry Flag (C) and the destination accumulator (ACD), unless this is prevented by a 1 in the No-Load (NL) field. In either case, they are tested for a skip condition (i.e., to determine if the next instruction should be skipped) as specified in the SK field of the instruction.

### 5.5.1.2 Overflow and Carry-Out Operations

The 16-bit numbers processed by the ALU may be thought of as unsigned integers between 0 and 64K or as signed integers between -32K and +32K.

<u>Binary Number</u> (in ALU)	<u>Unsigned Interpretation</u>		<u>Signed Interpretation</u>	
	Octal	Dec.	Octal	Dec.
1111111111111111	177777	64K-1	-00001	-1
1111111111111110	177776	64K-2	-00002	-2
1000000000000001	100001	32K+1	-77777	-32K+1
1000000000000000	100000	32K	-100000	-32K
0111111111111111	077777	32K-1	+77777	32K-1
0111111111111110	077776	32K-2	+77776	32K-2
0000000000000001	000001	1	+00001	1
0000000000000000	000000	0	00000	0

When working with either interpretation, there is the possibility of an overflow (answer greater than the maximum number that can be represented) or underflow (less than the minimum). In general, the ALU will produce the correct result if no overflow or underflow occurs, and will produce 64K more than or less than the correct result if there is underflow or overflow, respectively.

There is a relationship between underflow/overflow and the carry-out from the ALU MSB, but the relationship is not a simple one, as shown in the following paragraphs.

#### 1. Unsigned integers:

Decimal:  $0 \leq x < 64K$   
 Octal:  $0 \leq x \leq 177777$

When ADDing two numbers, if the true result is less than 64K, the ALU will produce the correct result and no carry-out will result. If the true result is greater than or equal to 64K, the ALU will produce 64K less than the true result (i.e., the true result truncated to 16 bits), and a carry-out will result. Note that in these cases a carry-out is synonymous with overflow and indicates that the ALU output is not the true result.

SUBtraction is accomplished in the ALU by complementing the subtrahend and adding it to the minuend, with a carry-in. Therefore, when SUBtracting one unsigned integer from another, if the true result is positive or zero, the ALU will produce the true result and will also produce a carry-out. If the true result is negative, the ALU will produce the true result plus 64K (since all numbers are interpreted as

positive), and no carry-out will result. Note that in these cases a carry-out is the opposite of underflow and indicates that the ALU output is the true result.

## 2. Signed Integers

Decimal:  $-32K \leq x < 32K$   
Octal:  $-100000 \leq x \leq 77777$

When ADDing two positive integers (or SUBtracting a negative integer from a positive one), if the true result is less than 32K, the ALU will produce the true result and no carry-out. If the true result is greater than or equal to 32K, the ALU output will appear negative (since the MSB = 1), being 64K less than the true result, and no carry-out will occur. Note that in this case an overflow is not signalled by a carry-out.

When ADDing two integers with opposite signs (or SUBtracting two numbers having the same sign), the ALU will always produce the true result, since the true result must be between -32K and +32K. A carry-out will occur if the result is positive and not if it is negative.

When ADDing two negative numbers (or SUBtracting a positive number from a negative one), if the true result is greater than or equal to -32K, the ALU will produce the true result. If the true result is less than -32K, the ALU output will appear positive (MSB = 0), and will be 64K greater than the true result. In either case a carry-out will always occur.

These relationships are illustrated in Figure 5-3.

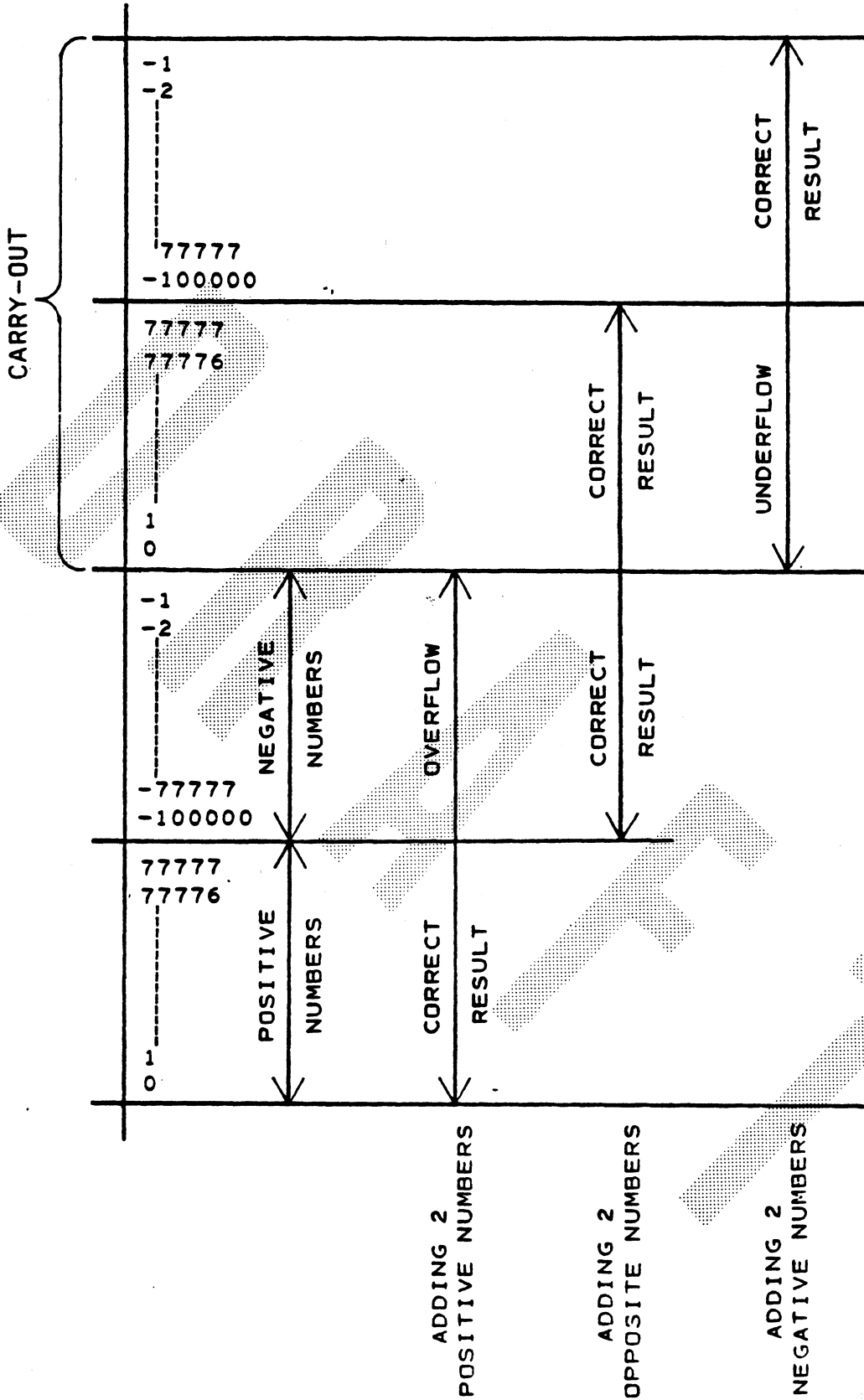


Figure 5-3 Overflow and Carry Operations  
Analysis for Signed Integers

## 5.5.2 ARITHMETIC/LOGIC FUNCTIONS

The OPCODE field (bits 5 through 7) defines one of eight arithmetic/logic operations to be performed by the 16-bit ALU. Table 5-5 defines these operations.

TABLE 5-5. ARITHMETIC LOGIC INSTRUCTIONS

Bits 5-7	OPCODE	Definition
000	COM	Complement: Complement the contents of ACS. Do not modify the preselected carry bit.
001	NEG	Negate: Produce the two's complement of the contents of ACS. If ACS=0, complement the preselected carry bit.
010	MOV	Move: Supply the unmodified contents of ACS. Do not modify the preselected carry bit.
011	INC	Increment: Add 1 to the contents of ACS. If the result is 0, complement the preselected carry bit.
100	ADC	Add Complement: Add the complement of ACS to ACD. Complement the preselected carry bit if ACS is less than ACD.*
101	SUB	Subtract: Subtract ACS from ACD. Complement the preselected Carry bit if ACS is less than or equal to ACD.*
110	ADD	Add: Add the contents of ACS to the contents of ACD. If the unsigned sum is greater than or equal to two to the sixteenth power, complement the preselected carry bit.
111	AND	And: Logically AND the contents of ACS with the contents of ACD. Do not modify the preselected carry bit.

\*Using a 16-bit unsigned integer interpretation.

### 5.5.3 SECONDARY FUNCTIONS

The SH (Shift), CY (Carry), NL (No-Load), and SK (Skip) fields specify secondary operations performed on the ALU result produced by the OPCODE field. These fields are discussed in the sections that follow.

#### 5.5.3.1 Shift Field (SH)

The SH field (bits 8 and 9) determines the shifting action (if any) produced by the Shifter on the result of the calculation produced by the ALU as defined in Table 5-6.

TABLE 5-6. SHIFT FIELD FUNCTIONS

Bits 8-9	Mnemonic	Definition
00	-	No Shift: Do not modify the ALU result. The carry resulting from the ALU operation is unaffected.
01	L	Left Rotate: Shift the result one place to the left, and insert the state of the carry resulting from the ALU (CRES) in the LSB (bit 15) position. Insert the out-shifted MSB (bit 0) into the carry bit (CNEW).
10	R	Right Rotate: Shift the result one place to the right, and insert the state of the carry resulting from the ALU (CRES) into the MSB (bit 0) position. Insert the out-shifted LSB (bit 15) into the carry bit (CNEW).
11	S	Swap: Swap the 8 MSBs of the result with the eight LSBs. The carry resulting from the ALU is unaffected.

### 5.5.3.2 Carry Control Field (CY)

The CY field (bits 10 and 11) specifies the base to be supplied to the ALU for carry calculation, as defined in Table 5-7.

TABLE 5-7. CARRY CONTROL FIELD FUNCTIONS

Bits 10-11	Mnemonic	Definition
00	-	No change: The current state of the carry flag is supplied to the ALU as a base for carry calculation.
01	Z	Zero: The value 0 is supplied to the ALU as a base for carry calculation.
10	O	One: The value 1 is supplied to the ALU as a base for carry calculation.
11	C	Complement: The complement of the current state of the carry flag is supplied to the ALU as a base for carry calculation.

The three logic functions (MOV, COM, AND) supply the values listed above as the carry bit to the Shifter. The five arithmetic functions (ADD, ADC, INC, SUB, NEG) supply the complement of the base value if the ALU operation produces a carry-out of bit 0; otherwise they supply the value listed above.

### 5.5.3.3 No-Load Field (NL)

The NL field (bit 12) determines whether or not the output of the Shifter is stored in ACD and in Carry. If bit 12=0, the Shifter output is stored in ACD and in Carry. If bit 12=1, no storage action occurs.



#### 5.5.3.4 Skip Control Field (SK)

The SK field determines the type of skip test to be performed on the Shifter output. If the selected skip test is affirmative, the next instruction is skipped. The skip tests that can be selected by the SK field (bits 13-15) are defined in Table 5-8.

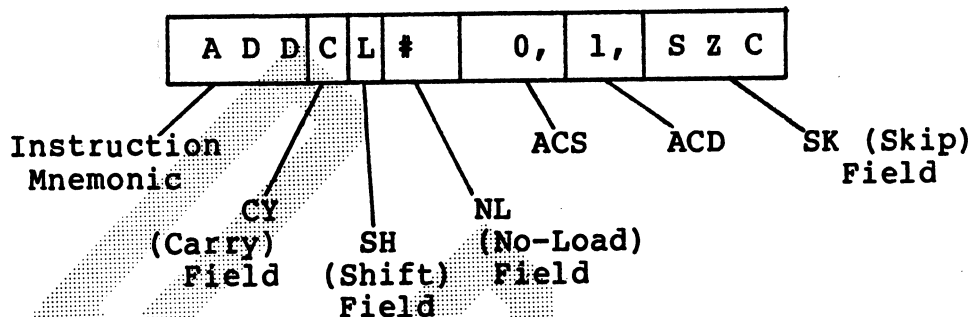
TABLE 5-8. SKIP CONTROL FIELD FUNCTIONS

Bits 13-15	Mnemonic	Definition
000	-	No skip test (never skip)
001	SKP	Skip unconditionally (no skip test required)
010	SZC	Skip if carry bit is zero
011	SNC	Skip if carry bit is nonzero
100	SZR	Skip if result is zero
101	SNR	Skip if result is nonzero
110	SEZ	Skip if either carry bit or result is zero
111	SBN	Skip if both carry bit and result are nonzero

### 5.5.4 ASSEMBLER LANGUAGE CONVENTIONS AND EXAMPLES

The assembler language arithmetic or logic instruction consists of the instruction OPCODE mnemonic (ADD, NEG, COM, etc.) followed by symbols that specify the carry indicator, the shift indicator, the load/no-load indicator, a source and a destination accumulator and the skip conditions. Table 5-9 shows the programming conventions for Arithmetic and Logic Instructions.

The format is as follows:



The CY, SH, NL, and SK fields are specified by adding the appropriate mnemonic symbols. None of these four fields has to be specified, but their symbols must appear in the proper order and place if they are included. Those fields not specified will be assembled containing 0's. For example:

ADDCL    0, 1

performs the following operation: Add A0 to A1 and supply the complement of the Carry flag to the ALU. Shift the 17-bit output to the left, and store it into A1 and the Carry flag.

**TABLE 5-9. ASSEMBLER LANGUAGE CONVENTIONS FOR ARITHMETIC AND LOGIC INSTRUCTIONS**

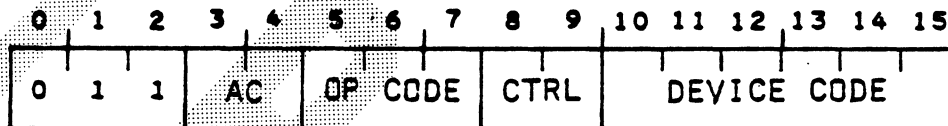
Instruction Function	OP CODE Mnemonic	Optional Secondary Functions		Separator Space or Tab	Accumulators		Optional Skip (SK*)
		CY*	SH*		NL*	ACS	
Add	ADD						blank
Subtract	SUB						SKP
Move	MOV	none	none		0	0	SZC
Increment	INC	Z	L	none	1	1	SNC
Negate	NEG	O	R	#	2	2	SZR
Complement	COM	C	S		3	3	SNR
Add Complement	ADC						SEZ
Logical And	AND						SBN

\* Elimination of a mnemonic symbol for these fields will cause the field to be assembled as all zeros.

## 5.6 INPUT/OUTPUT INSTRUCTION GROUP

The Input/Output Instructions enable the processor to communicate with the peripheral devices on the system and also perform various operations within the processor. I/O instructions transfer data between accumulators and devices, start or reset device operation, or check the status of each device. Each I/O instruction contains a 6-bit device code field, which specifies the particular device for this data transfer. The system allows up to 63 peripheral devices, with each device assigned a unique code from 00 through 76 octal. The 77 octal code denotes a special class of instructions that controls certain CPU functions such as interrupt handling. Use of the 00 code is not recommended, since a device with that code would give a default response to an Interrupt Acknowledge instruction.

All instruction words in this category have the following format:



### **Input/Output Instruction Format**

An instruction in this class is designated by 011 in bits 0-2. The OPCODE and Control (CTRL) fields define the I/O operation to be performed. If a data transfer operation is involved, the AC field (bits 3 and 4) specifies the CPU Accumulator involved in the data transfer (otherwise it has no effect). Bits 10-15 select the device that is to respond to the instruction.

#### **5.6.1 PROGRAMMED I/O INSTRUCTIONS**

Programmed I/O instructions apply to all device codes except code 77. I/O transfer instructions are:

NIO, DIA, DOA

The POINT 4 MARK III programmed input/output system provides for the following functions to be specified:

- o Full 16-bit data transfer:
  - Input channel (DIA)
  - Output channel (DOA)
- o Control Functions:
  - Three control pulses which specify multiplexer, disc or tape transfers, designated by S, C or P, respectively.

These input/output functions are illustrated in the diagram below:

To Be Supplied

Each device interface contains a 6-bit address decoder (bits 10-15). When the processor executes an I/O instruction, it places the specified device code onto the Device Select lines of the I/O Bus. The appropriate device will recognize its own code and thus respond to the I/O instruction. All other devices ignore the instruction.

### 5.6.1.1 I/O Transfer and Device Control Instructions

I/O Transfer instructions move data between the processor and the device interface. The OPCODE field (bits 5-7) of the instruction specifies the type of transfer to take place (Data In, Data Out, No Transfer, etc.). Bits 3 and 4 specify the accumulator that supplies or receives the data and bits 8 and 9 specify a control function which is not used in i/o transfer instructions. The type of transfer is determined by the code in the OPCODE field as illustrated in Table 5-10.

**TABLE 5-10. I/O TRANSFER INSTRUCTIONS**

Bits 5-7	OPCODE	Definition
000	NIO	No data transfer involved. Device control only.
001	DIA	Transfers (reads) the contents of either the status or the data register in the specified device into the CPU accumulator indicated in the AC field. If the device code is even, the contents of the status register will be transferred. If the device code is odd, the contents of the data register will be transferred.
010	DOA	Transfers (writes) the contents of the accumulator indicated by the AC field to either the command or the data register in the specified device. If the device code is even, the contents of the accumulator will be transferred to the command register. If the device code is odd, the contents of the accumulator will be transferred to the data register.

### 5.6.1.2 Assembler Language Conventions and Examples

An assembler language I/O Transfer Statement consists of the instruction mnemonic, an optional control function, an accumulator and octal device code. Table 5-11 shows the programming conventions for Regular Input/Output Instructions.

For Example:

```
D I A      2, 5 0
```

Instruction Mnemonic	Device Code	Accumulator
----------------------	-------------	-------------

This instruction transfers the contents of AZ to the IOCB pointer for device 50 (disc). If AZ=0, DMA transfer will be discontinued, otherwise, DMA transfer will be activated.

The device code may be represented by a device mnemonic. Thus,

```
D I A      D S C
```

is equivalent to the previous example because DSC represents device 50 (disc controller).

A No-I/O (NIO) will not specify an accumulator since no data transfer occurs:

```
N I O      D S C
```

**TABLE 5-11. ASSEMBLY LANGUAGE CONVENTIONS  
FOR INPUT/OUTPUT INSTRUCTIONS**

**U  
R  
F  
T**

To Be Supplied



## 5.6.2 SPECIAL CODE 77 (CPU) INSTRUCTIONS

Certain system functions and interrupt processing control are accomplished via I/O instructions with the octal code 77 in bits 10-15. These instructions do not directly address a particular device and the device code mnemonic is CPU.

CPU instructions have the same general format as regular I/O instructions. The OPCODE field and Control field, however, are interpreted differently.

### OPCODE Field:

- o Addresses all I/O devices simultaneously for certain interrupt control functions.
- o Does nondata transfer functions such as resetting all I/O devices and transferring control to the Virtual Control Panel program MANIP.

### Control pulse:

- o Addresses the Multiplexer
- o Addresses the Disc Controller
- o Addresses the Tape Controller

See the chart of special I/O instructions for details of instruction functions.

The CPU has two flags which can be tested by the I/O Skip instructions:

- o BUSY - ION set (Interrupts are enabled)
- o DONE - Power-failure has been detected (Will cause interrupt if ION set and software will stop operations in order to prevent power failure while performing a disc transfer)

### 5.6.2.1 Special Mnemonics for CPU Instructions

The assembler also recognizes several special mnemonics for CPU instructions. The regular instruction mnemonics and the special mnemonics along with a description of special CPU instructions are listed in Table 5-12.

**TABLE 5-12. SPECIAL CPU I/O INSTRUCTIONS**

Instruction	Special Mnemonic	Definition
NIO CPU		No Action
NIOS CPU	INTEN	Set the processor's Interrupt On (ION) flag. The processor will now respond to interrupt requests from devices, after execution of one more instruction.
NIOC CPU	INTDS	Clear the Interrupt On flag, so that the processor will not respond to interrupt requests.
DOB a, CPU	MSKO a	Set up the Interrupt Disable flags in all devices simultaneously, according to the mask code in accumulator a. A MSKO with a=177777 disables interrupts from all devices. A MSKO with 177776 enables all interrupts. A MSKO with a=177775 enables interrupts as determined by the LSB. The values a=1 or a=0 have special meanings: a=1 disables all DMA a=0 enables DMA to controllers
DICC 0, CPU	IORST	Clears the processor ION flag; clears any pending interrupts; clears DMA mode (MSKO with LSB of accumulator set to 1); clears MSKOUT (MSKO with LSB set to 0). Also sends IOBRST pulse to Peripheral Interface Board which a) clears all disc control registers, b) clears all tape control registers, c) but does not change multiplexer communications controller set up or the IOCB pointers in accumulators A13, A14, and A15.
DOC 0, CPU	HALT	Transfers control to the processor Virtual Control Panel, thus enabling use of MANIP commands for front panel operations.

Note that the special mnemonic does not allow the programmer to specify the S and C functions. For example,

**MSKO 3**

when executed, MSKO3 sets the interrupt status based on the value in accumulator 3. To activate the disc controller it would be necessary to use:

**DOBC 3,CPU**

The instruction IORST, however, assumes the C function. All I/O device flags are reset and the ION flag is cleared.

### **5.6.2.2 Control Field Uses**

The control field is used in conjunction with DIB and DOB instructions with device code 77 to control interrupt polling and DMA transfers. Table 5-13 describes the functions of the control codes (S, C and P) when used with DIB and DOB instructions.

**TABLE 5-13. CONTROL FIELD DEFINITIONS FOR  
I/O INSTRUCTIONS WITH DEVICE CODE 77**

OPCODE Bits 5-7	OPCODE Mnemonic	Control Bits 8-9	Control Mnemonic	Definition
011	DIB None	00 None	None	None
011	DIB	01	S	Reads pointer for multiplexer IOCB to determine if an interrupt is pending (MSB=1).
011	DIB	10	C	Reads pointer for disc controller IOCB to determine if an interrupt is pending (MSB=1).
011	DIB	11	P	Reads pointer for tape controller IOCB to determine if an interrupt is pending (MSB=1).
100	DOB None	00 None	None	None
100	DOB	01	S	Sets pointer to multiplexer IOCB but does not activate multiplexer. DOA instructions may be issued to ports to be activated.
100	DOB	10	C	Sets pointer to disc controller IOCB and activates disc controller. The controller then performs automatic block transfer based on information in IOCB. If a DOBC is issued with accumulator set to zero, automatic block transfer is shut off and/or pending interrupt bit for disc controller is reset.
100	DOB	11	P	Sets pointer to tape IOCB and activates tape controller. The controller then performs automatic block transfer based on information in IOCB. If a DOBP is issued with accumulator set to zero, automatic block transfer is shut off and/or pending interrupt bit for tape controller is reset.

### 5.6.2.3 Skip Instructions

A value of 111 in bits 5-7 signifies a conditional skip instruction. The function field in this case indicates which processor flag (Interrupt On or Power Fail) will be tested, as shown in Table 5-14.

**TABLE 5-14. I/O SKIP INSTRUCTIONS**

Bits 8 & 9	Instruction	Definition
00	SKPBN CPU	Skip next instruction if Interrupt On is nonzero.
01	SKPBZ CPU	Skip next instruction if Interrupt On is zero.
10	SKPDN CPU	Skip next instruction if the Power Failure flag is nonzero.
	11SKPDZ CPU	Skip next instruction if the Power Failure flag is zero.

### 5.6.2.4 Assembler Language Conventions and Examples

CPU instructions are usually written using the special mnemonics shown in Section 2.6.2. However they may also be written in the same manner as regular I/O instructions, specifying the instruction mnemonic, optional control function, optional accumulator, and a device code of 77 octal (mnemonic CPU). For example:

```
NIOS CPU
```

sets the ION flag in the processor.

## 5.7 INSTRUCTION EXECUTION TIMES

One of the outstanding features of the POINT 4 Computer is the substantial reduction in instruction time over execution time in Comparable mini-Computers. Table 5-15 gives instruction execution times for the POINT 4 Computer.

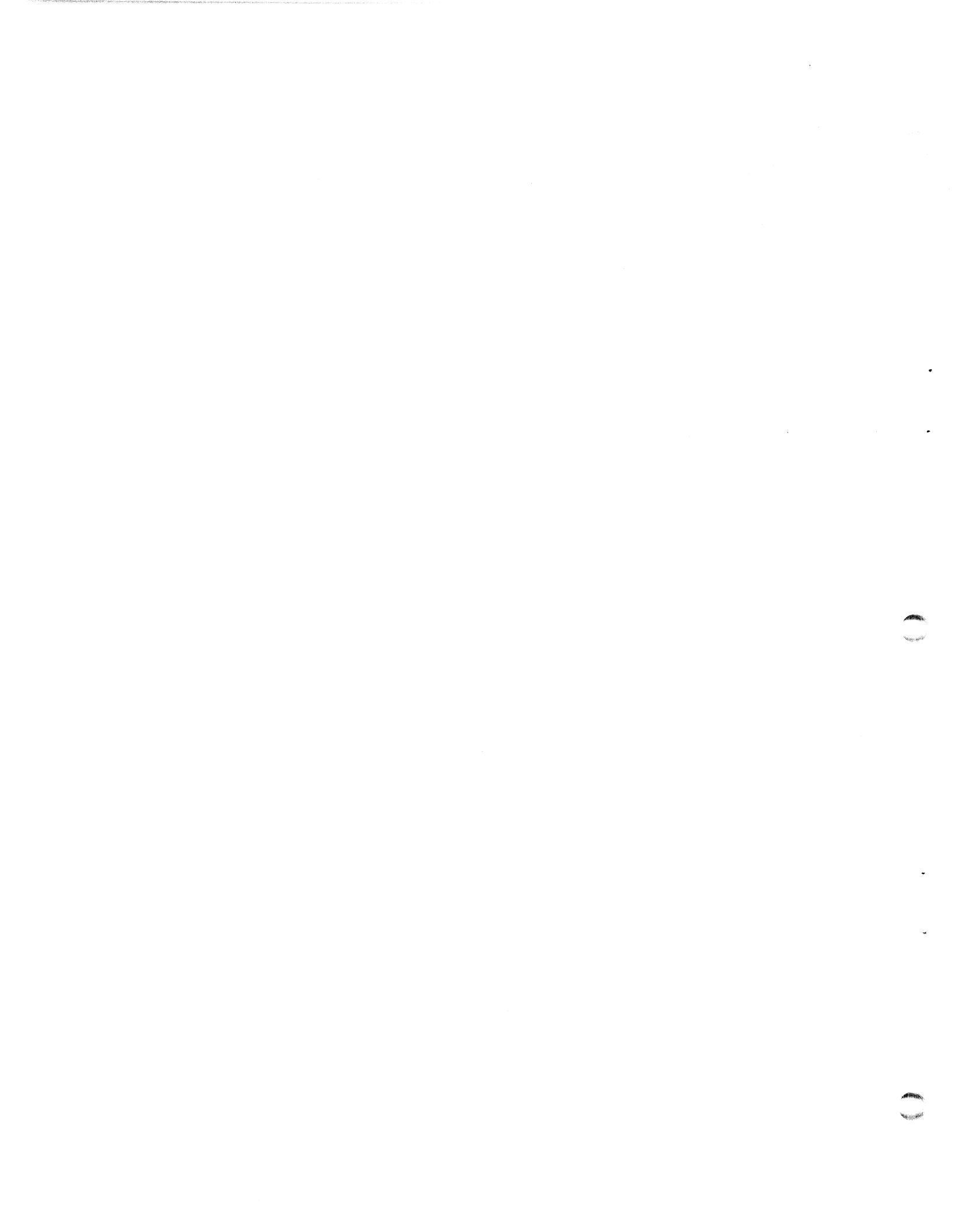
**TABLE 5-15. INSTRUCTION EXECUTION TIMES**

Instruction Category	Instruction (Generic Types)	Execution Times (nanoseconds)
MEMORY REFERENCE	Load or Store Accumulator (LDA, STA)	1400
	Increment or Decrement if Zero (ISZ, DSZ)	1800
	Jump (JMP)	800
	Jump to Subroutine (JSR)	800
	For Indirect Addressing add	600
ARITHMETIC/ LOGIC	Arithmetic/Logic Instructions (COM, NEG, MOV, INC, ADC, SUB, ADD, AND)	600
	For skip add	0
	For swap add	200
INPUT/ OUTPUT	Input (DIA)	2600
	Output (DOA)	2200
	I/O Skips (SKFBN, SKPBZ, SKPDN, SKPDZ) - CPU only	2000
DATA CHANNEL TRANSFERS	Disc Transfers:	
	Input (16 bits)	1200
	Output (16 bits)	1200
	MUX Transfers (includes automatic vectoring, special character test, and buffer-end test):	
	Input (8 bits)	9600
	Output (8 bits)	8400
	Tape Transfers:	
Input (8 bits)	5600	
Output (8 bits)	5600	

These times are exclusive of three types of overhead:

1. A 600-nanosecond refresh cycle takes place once every 16 microseconds - this adds about 4% overhead.
2. Arithmetic/Logic instructions on RAM page boundaries (2 least significant digits of address = 76 or 77) take an extra 200 nanoseconds - this results in approximately 1/2% overhead.







## Section 6

### PERIPHERAL DEVICE HANDLING

---

#### 6.1 PERIPHERAL INTERFACE BOARD

The Peripheral Interface Board (PIB) is a peripheral controller that interfaces the MARK III CPU to three groups of peripherals: Asynchronous Communications Module; Disc Drive Module; and Tape Drive Module. PIB interfaces to the MARK III CPU via the MUXI-BUS, providing a high-speed DMA transfer between peripherals and the CPU.

The PIB has no on-board micro engine. The CPU multiplexes its intelligence to the controlling modules on the PIB as needed.

PIB features include:

- o Separate Software Programmed I/O Control
- o DMA Data Transfer
- o Simultaneous Operation of Three Groups of Peripherals
- o Ease of Software Interface

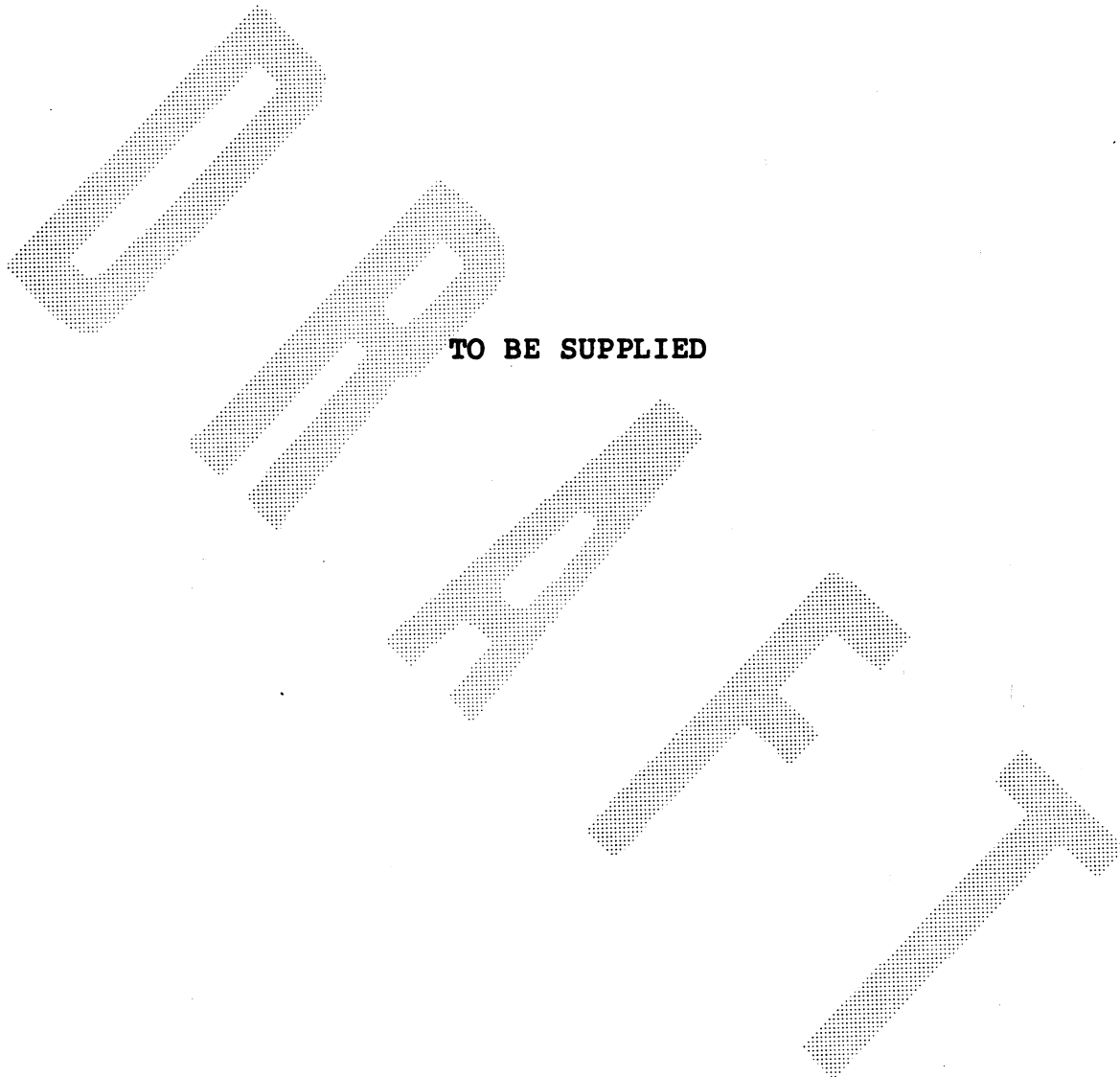
#### 6.1.1 SYSTEM DESCRIPTION

The PIB contains three main controlling modules:

1. Asynchronous Communications Module
2. CMD/SMD Disc Drive Module
3. Tape Drive Module

Utilizing these modules, the MARK III system can control up to three different types of peripherals simultaneously.

Figure 6-1 shows the basic System Block Diagram for the PIB. Communications is via the MUXI-BUS DMA facility, utilizing I/O Control Blocks (IOCBs) for access control. The PIB includes device control and instruction decode logic. This allows the CPU to access each module individually.



**Figure 6-1. Basic System Block Diagram**

## 6.1.2 EQUIPMENT CHARACTERISTICS

### 6.1.2.1 Performance Characteristics

#### COMMUNICATIONS CONTROL

Transmission Type: Asynchronous

Line Types: Half or Full Duplex

Line Interface: RS-232C

Number of Ports: Four

#### Modes of Operation:

##### Programmed I/O

Port 0 - Device Code 10/11

Port 1 - Device Code 12/13

Port 2 - Device Code 14/15

Port 3 - Device Code 16/17

DMA Operation - Device Code 77

Line Printer: May be connected to any port

#### Baud Rates (Hardware Strappable):

110, 150, 300, 600, 1200, 2400, 4800, 9600

#### DISC DRIVE INTERFACE

Drives per Controller: Two

Drives Type: SMD/CMD drives supporting the sector mark interface signal

DMA Transfer Rate: 1.25 Megabytes per second

#### Sector Size:

Header - 4 words

Data - 256 words

#### TAPE DRIVE INTERFACE

Designed to be compatible with industry standard streamer tape units.

## INPUT/OUTPUT INSTRUCTIONS

Input - DIA

Output - DOA

Device Codes:

10-17 = MUX Ports 0-3

50-55 = Disc

60-62 = Tape

DMA (Device Code 77):

DOBS/DIBS = MUX

DOBC/DIBC = Disc

DOBP/DIEP = Tape

CPU FUNCTIONS (Device Code 77):

SKPBN = Skip if Interrupts Enabled

SKPDN = Skip if Power Fail Detected

SKPBZ/SKPDZ = Opposite of SKPBN, SKPDN

### 6.1.2.2 Equipment Specifications

#### Power Requirements

+5V dc .....5.7 A

-5V dc .....4 mA

-12V dc.....29.0 mA

#### Environmental Requirements

Temperature:

0 to +50C operating

-20 to +100C non-operating

Relative Humidity: 5 to 90% noncondensing

Dimensions: 14.5" x 12" (36.8. cm x 30.5cm)

### 6.1.3 PIB INTERNAL ARCHITECTURE

Figure 6-2 shows the internal architecture of the Peripheral Interface Board.

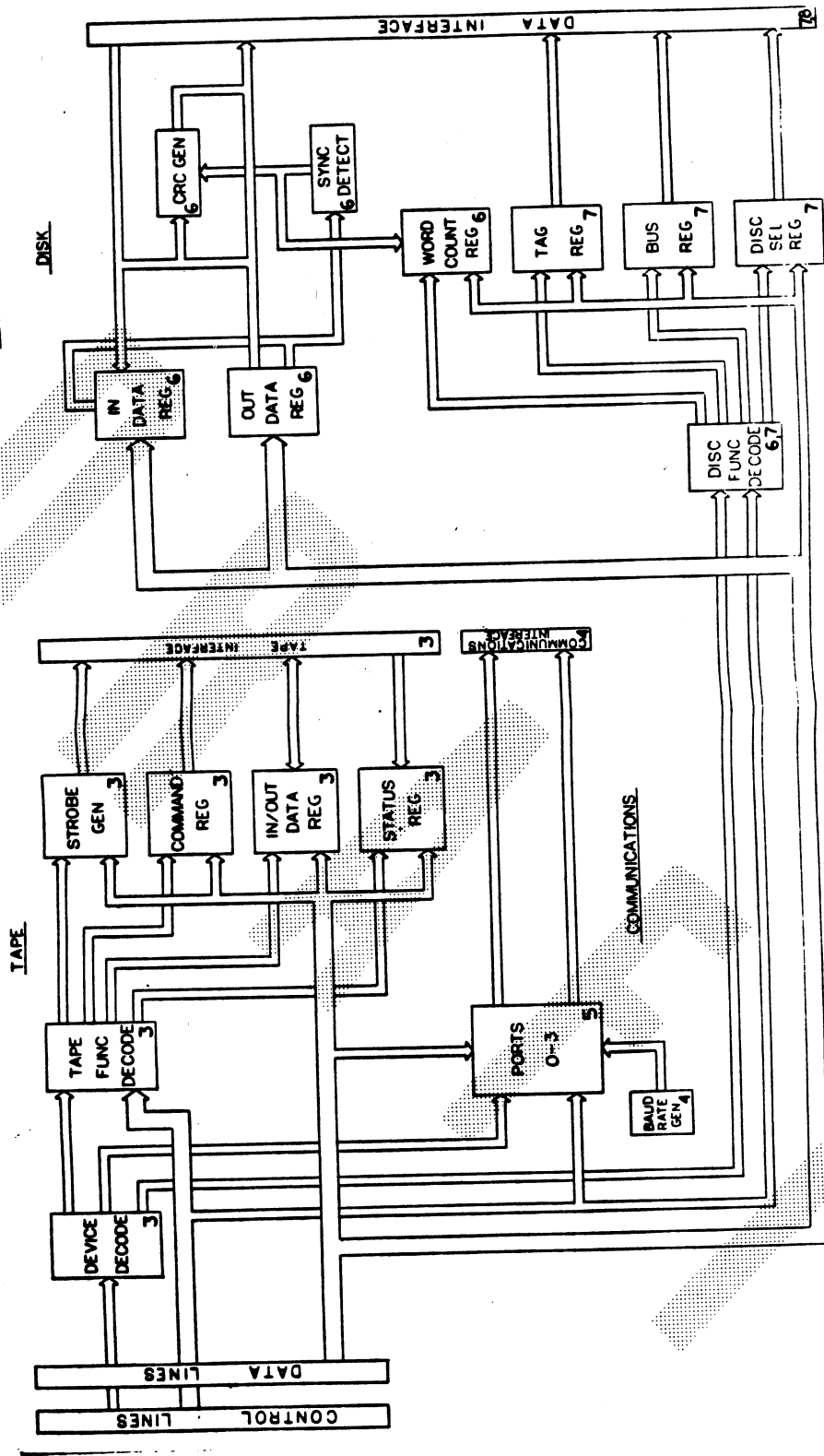


Figure 6-2. PIB Internal Architecture

## **6.2 INPUT/OUTPUT INTERFACE HANDLING**

This section provides information to the user about the basic operating principles and programming methods for the input/output devices that are compatible with the POINT 4 MARK III computer. There are two types of I/O devices:

- o Those that transfer data via I/O programmed instructions only
- o Those that use Direct Memory Access (DMA) for input/output transfers

The following subsections outline the interrupt handling and priority scheme, programmed transfer handling and DMA channel transfer handling.

### **6.2.1 PROGRAM INTERRUPT AND PRIORITY SCHEME**

Many input/output devices require service within a short time after they request it, but they need service infrequently relative to the processor speed and only a small amount of time is required to service them. Failure to service within the specified time (which varies among devices) causes operation of the device below its maximum speed and can result in loss of information.

The use of interrupts in the current program sequence facilitates concurrent operation of the main program and a servicing of number of peripheral devices. The program interrupt scheme allows an I/O device to gain control of the processor. When an interrupt occurs, the processor suspends normal program execution and starts a device service routine. When the routine is completed, the processor returns to the interrupted program.

### 6.2.1.1 Interrupt Sequence

When a device needs service, it sets its Interrupt Request flag. The processor begins servicing interrupts if all four of the following conditions exist:

- o The processor has just completed an instruction fetch or a data channel transfer
- o At least one device has a pending Interrupt Request
- o Interrupts are enabled (i.e., ION is set)
- o No device is waiting for a DMA transfer

The processor responds to the interrupt request by storing the value of the program counter into memory location 0 and jumping to the instruction addressed by memory location 1. Location 1 must contain the address of the interrupt handling routine. Interrupts are disabled at the start of the interrupt service cycle and must be re-enabled by the software at the end of the interrupt service.

The POINT 4 MARK III has only priority level for interrupts. The Mask Out (MSKO) instruction is used, however, to Mask Out interrupts and thus allow use of programming routines using INTDS/INTEN instructions. The single priority level is reflected in the least significant bit (LSB) of the accumulator specified in the MSKO instruction. The least significant bit of the accumulator can be interpreted as follows:

Accumulator Value	Function
ODD	Masks out (disables) all interrupts
EVEN	Masks in (enables) all interrupts
= 1	Disables DMA to controllers
= 0	Enables DMA to controllers

### 6.2.1.2 Programming Polling and Interrupts

Pending interrupts are flagged in the Most Significant Bit (MSB) of the pointer to the device controller Input/Output control Block (IOCB). The DIB instruction with a device code of 77 (CPU) is used to read the IOCB pointers (accumulators 13-15). There are three forms of DIB instructions, one for each of the device controllers. The control field of the I/O instruction is used to designate multiplexer (S), disc controller (C) or tape controller (P). The resulting I/O instructions are:

- o DIBS (a),77    Read MUX Pointer
- o DISC (a),77    Read Disc Pointer
- o DIBP (a),77    Read Tape Pointer

Two software procedures may be used to service I/O devices using the DIB instruction. They are:

1. Disable interrupts using an INTDS or a MSKO instruction. The DIB with S, C or P must be used to constantly poll the IOCB pointers to check the MSB for = 1. Upon detecting a pending interrupt, the processor stores the value of the program counter into memory location 0 and jumps to a device service routine. After servicing of the first pending interrupt encountered the other IOCB pointers should be checked, since more than one interrupt may be pending at one time.
2. Enable interrupts using an INTEN instruction. The system will now be in automatic transfer mode. When an interrupt is generated by a device requesting service, the processor will reset the interrupt request flag and jump to a software interrupt handling routine. There the DIB instruction is used to read the IOCB pointers to check the MSB for = 1. The controller causing the interrupt will be serviced immediately as described in Subsection 6.2.1.1. Note that if more than one controller is requesting service, each should be serviced at this time. If not, a pending interrupt will be ignored.

In the automatic transfer mode, interrupts may also be generated by a controller when either the block transfer is finished or when a status change or error has occurred. In either case the controller will set the interrupt request flag and the processor will reset the interrupt request flag and jump to a software handling routine, such as described above.

When using either method, the software must first reset the pending interrupt bit in the IOCB pointer and then turn off automatic transfer mode, if required (usually not done for the multiplexer). Next the software must jump to an interrupt servicing routine.

The IOCB for each multiplexer sort contains an Operation Complete bit (Bit 0 of Word 0 or 1). The Operation Complete Bit is set by the multiplexer when an interrupt is requested. This bit is used by the software to confirm that the pending interrupt was caused



by the IOCB for this device. The Operation Complete Bit is essential to multiplexer operation, since there is only one pending interrupt flag for the four multiplexer ports. Each port is identified by an IOCB. Bit 0 of Word 0 (input operations) and Word 1 (output operations) designates operation complete status for that port. When software detects an interrupt pending for the multiplexer, the software must scan the four IOCBs for multiplexer ports to determine which port needs service (Bit 0 set to 1). There may be more than one port completing transfer at any one time, care must be taken to scan all Operation Complete bits, even if one set Operation Complete bit has been detected. The Operation Complete bit must be reset to 0 by the software in order to prevent redundancy in servicing of I/O ports.

In addition to an Operation Complete bit (Bit 0, Word 7), the disc controller IOCB also contains a Summary Error bit in the termination status (Bit 1, Word 7) which is used to flag the existence of an error condition in disc transfer. Like the Operation Complete bit, the Summary Error bit is set by the disc controller when an error has been detected and is read by the software to determine the cause of the interrupt. This bit must be reset to 0 by software in order to prevent redundancy in interrupt servicing.

Figure 6-3 is a flowchart showing the sequence of steps that must be taken into account when programming polling and interrupt processing procedures. Note that interrupts other than those caused by the Peripheral Interface Board (i.e., power-fail interrupt) are not covered in this flowchart. Figure 6-3 commences with the polling of the interrupt pending bit. Arrival at this point in the software procedure may have been from a constant polling sequence or from an interrupt that caused a jump to a poll routine.

Figures 6-4 and 6-5 are flowcharts of parts of I/O service routines for the multiplexer and for disc or tape. The part of the service routines shown only take into account testing of Operation Complete and Summary Error bits. The remainder of the service routine is determined by the programmer.

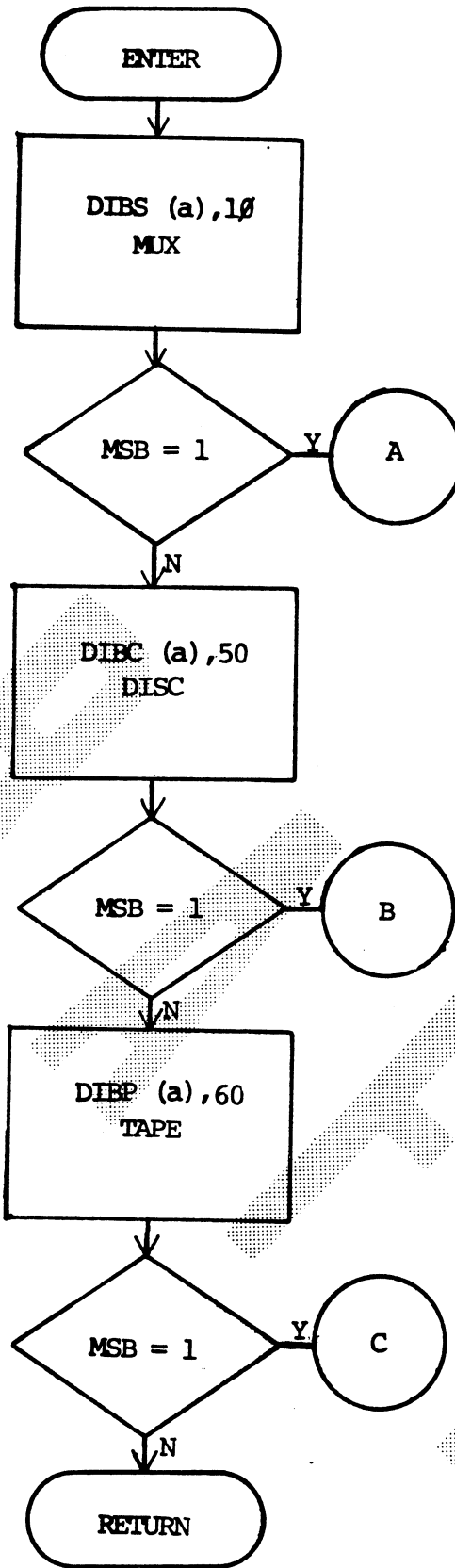


Figure 6-3. Software Polling/Interrupt Handling Flowchart (1 of 2)

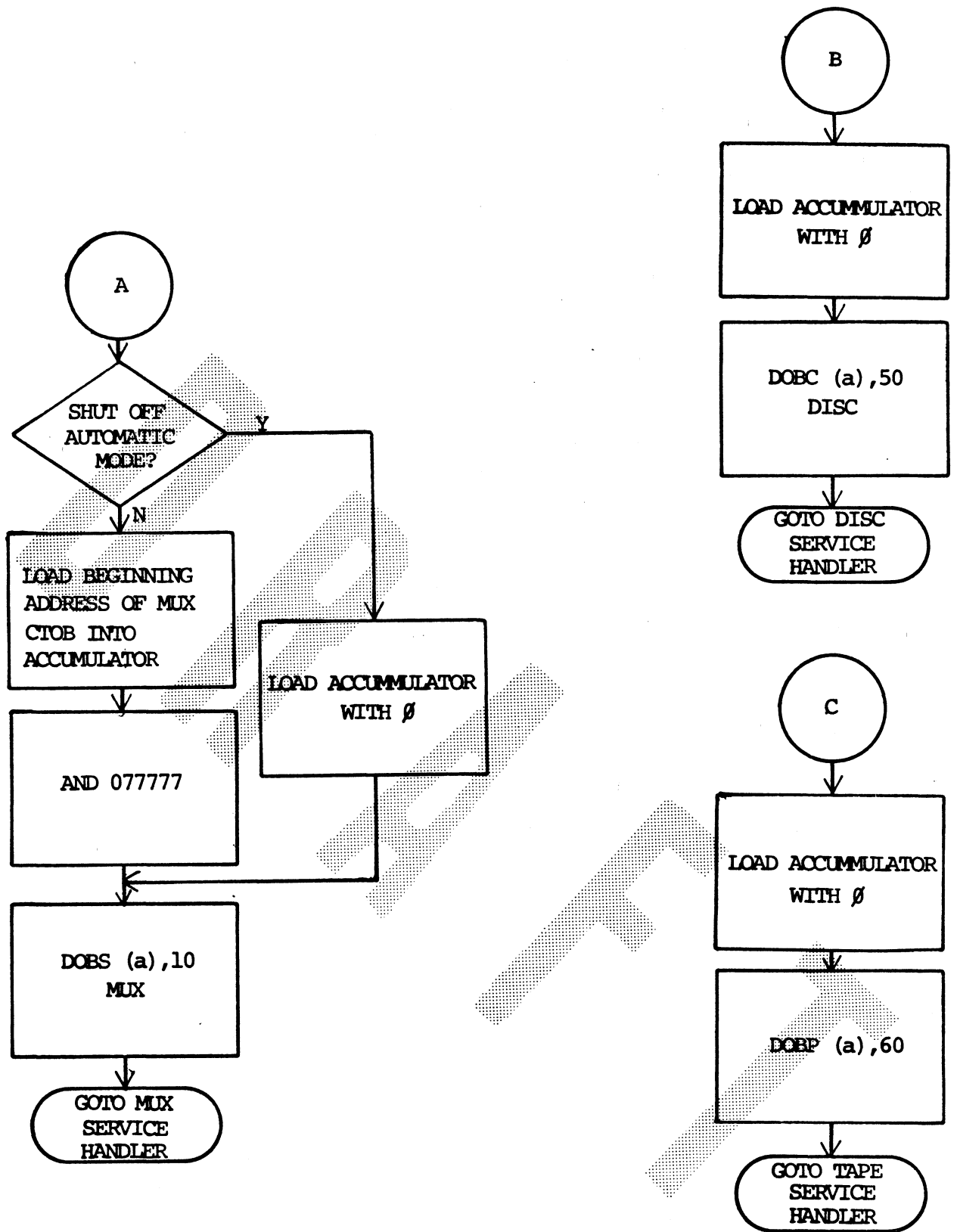
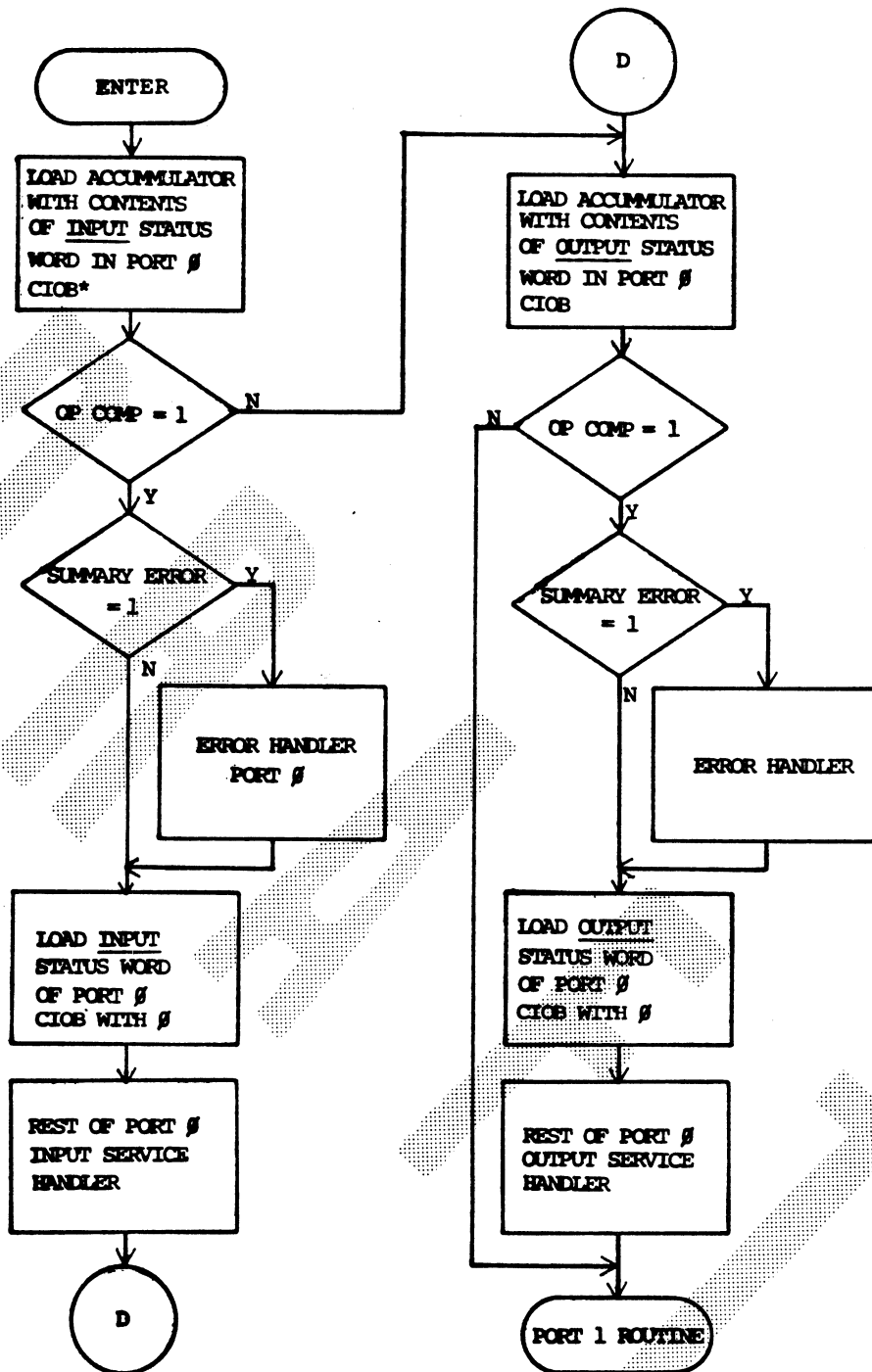
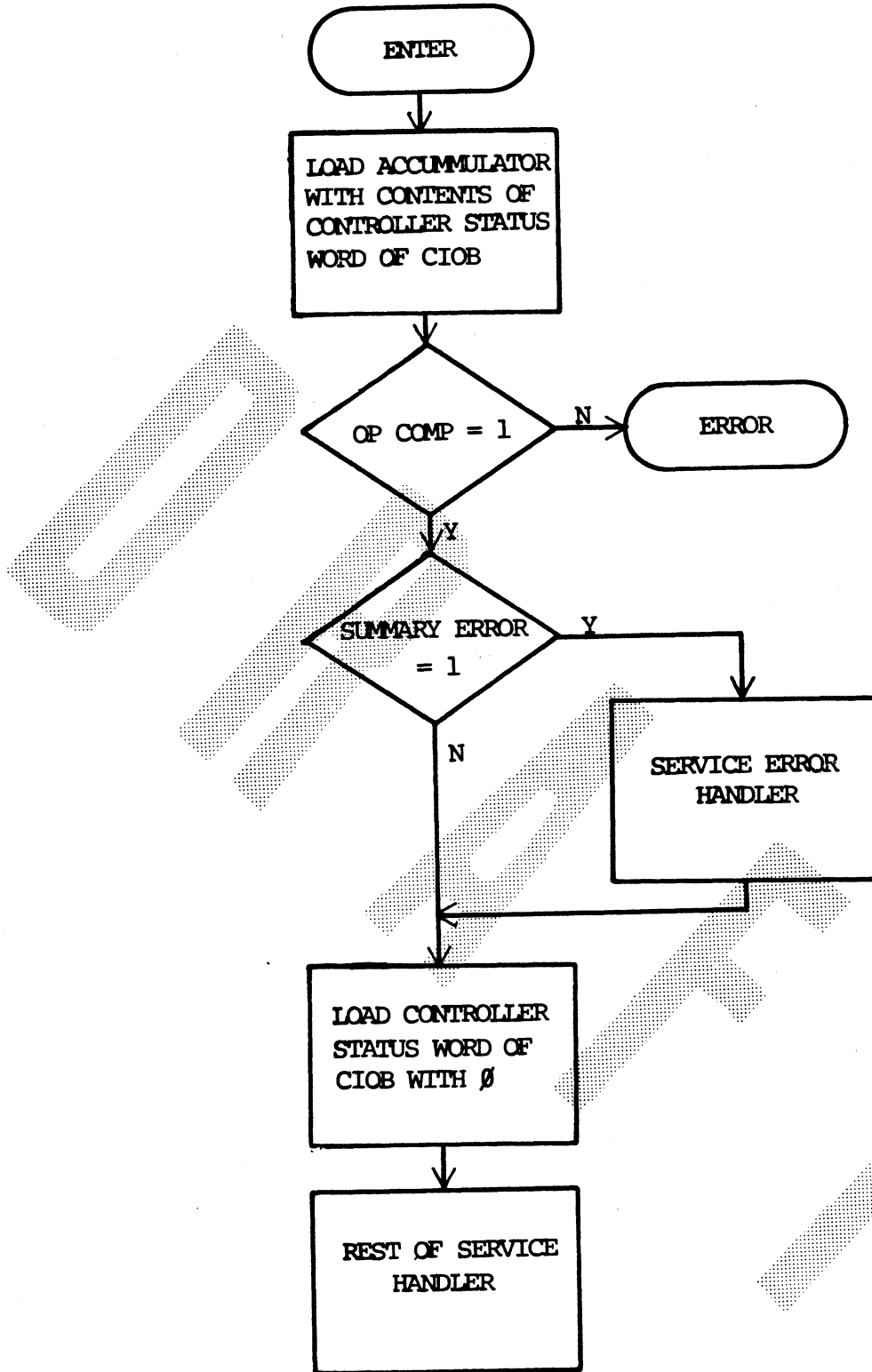


Figure 6-3. Software Polling/Interrupt Handling Flowchart (2 of 2)



\*PORTS 1, 2, +3 HANDLED  
IN SAME MANNER AS PORT β

Figure 6-4. I/O Service Routine Multiplexer Handling Procedures



**Figure 6-5. I/O Service Routine Disc  
Tape Handling Procedures**

## 6.2.2 PROGRAMMED TRANSFERS

For programmed input/output the program directly controls the data transfer between the CPU and the I/O device. Programmed I/O is used by the multiplexer in byte mode of communications. As discussed in Section 5.6, on Input/Output Instructions, each data word is transferred between an accumulator specified in the instruction and an I/O device register.

Programmed I/O uses the DIA and DOA instructions to program transfers of data and status information between the CPU and peripheral devices. The two programmed I/O instructions are used as follows for I/O programming:

**DIA Transfers** (reads) the contents of either the status or the data register in the specified device into the CPU accumulator. If the device code is even, the contents of the status register will be transferred. If the device code is odd, the contents of the data register will be transferred.

**DOA Transfer** (writes) the contents of the CPU accumulator to either the command or the data register in the specified device. If the device code is even, the contents of the accumulator will be transferred to the command register. If the device code is odd, the contents of the accumulator will be transferred to the data register.

See Subsection 6.3.2 for a detailed description of Byte Mode Operations.

### 6.2.2.1 Master Terminal Interface

The CRT I/O controller has separate interface logic for input and output. The input logic interfaces to the keyboard, some Teletypes, a paper tape reader. The output logic interfaces to the video display. When the CRT is connected to the computer, a character entered on the keyboard for input to the computer must be "echoed" back to the output interface logic on the terminal in order to appear on the screen.

All alphanumeric and control characters are represented by standard ASCII code (see Appendix C) consisting of eight bits, the most significant of which is usually an even parity bit.

The following are programming conventions for handling CRT terminals:

**Instruction Formats:** The data transfer output instruction transmits bits 8-15 from the specified accumulator to the output interface register. The input instruction loads the input interface register into bits 8-15 and resets bits 0-7 of the specified accumulator.

**Terminal Data Word Output:** To transfer a data character from an accumulator to the terminal data register, the following instruction is used:

DOA (a),11

(where (a) stands for any of the four accumulators).

**Terminal Command Output:** To transfer a command from an accumulator to the Port 0 terminal, the following instruction is used:

DOA (a),10

This instruction will cause bits 8-15 of the accumulator to the command register of Port 0. bit 0 of the accumulator is also used for enabling/disabling the Real Time Clock.

**Terminal Data Word Input:** When a key is pressed on the keyboard, the data word is placed in the data register. If interrupts are enabled an interrupt is requested. The program then reads the data word from the data register with a

DIA (a),11

**Terminal Status Input:** A change in terminal status will cause an interrupt request. if interrupts are enabled. The program must read the status of Port 0 from the Status register into bits 8-15 of the specified accumulator using a:

DIA (a),10

### 6.2.3 DIRECT MEMORY ACCESS TRANSFERS

Mass storage devices such as tape drives, and discs can transfer blocks of data at high speeds directly into memory, without requiring programmed I/O instructions for each word transferred, by using the DMA (direct memory access). DMA device interface logic contains both conventional device registers and special data channel logic.

The program initiates a DMA transfer by supplying certain parameters to the device registers and starting the device. The device automatically transfers one or more data words to or from memory. When finished with the DMA transfer, the device generates an interrupt if so enabled. At the start of each instruction cycle, the processor checks to see if a device is requesting DMA service. If a device is requesting service, the DMA transfer is performed before going on with the instruction.

The time a device must wait for DMA access depends on when its request is made within an instruction and how many other devices are also requesting access. Once the processor reaches a point at which it can pause to handle transfers, a given device must wait until the processor checks each IOCB pointers for an interrupt request flag (bit 0) set to 1. The first device encountered with this flag set will be serviced, then all other pointers will be checked and serviced if required. An exception is made if Power-Fail has been sensed, in which case the DMA is allowed only every other cycle - the alternate cycle being used for Power-Fail Interrupt processing. The first device serviced never waits longer than the time required for the processor to finish the instruction that is being performed when the request is synchronized. However, indirect addressing can extend this beyond the normal instruction execution time.

See Subsection 6.2.1.2 for Interrupt Handling Procedures.



### 6.2.3.1 Enabling/Disabling DMA Transfers

The Mask Out (MSKO) instruction is used to enable and disable DMA Transfers. In order to enable/disable DMA Special values must be loaded into the CPU accumulator specified in the MSKO instruction. These values are:

Accumulator Value	Function
1	Disables DMA to controllers
0	Enables DMA to controllers

An alternate way of expressing the MSKO instruction is to use a DOB instruction with a device code of 77 octal as follows:

Instruction	Accumulator Value	Function
DOB (a),77 octal	1	Disables DMA to controllers
DOB (a),77 octal	0	Enables DMA to controllers

Alternate methods of disabling DMA Transfer include:

- o Issue a IDRST instruction (clears the processor's ION flag)
- o Press the RESET button (gives control to the Virtual Control Panel-MANIP)
- o Power-up the system (gives control to the Virtual Control Panel-MANIP)
- o Issue a HALT instruction (gives control to the Virtual Control Panel-MANIP)

DMA should be disabled the minimum time possible. The longer the period of time disabled, the greater the risk of having an overrun or data in the multiplexer or of having the tape streamer come to a stop.

### 6.2.3.2 Initiating DMA Transfer

Three configurations of the DOB instructions are used to initiate DMA transfers. They are:

DOBS (a),77 octal Set Multiplexer IOCB  
DOBC (a),77 octal Set Disc IOCB  
DOBP (a),77 octal Set Tape IOCB

Accumulator (a) contains the memory address of the Input/Output Control Block (IOCB) for the device specified. Accumulators 13 through 15 are used for IOCB pointers as follows:

Accumulator	Device Controller IOCB
15	Disc
14	Tape
13	Multiplexer

If these IOCB pointers contain a value of 0, the device controller is not being used in the system. The most significant bit (MSB) of this accumulator is set to 1 when there is an interrupt pending. For DMA transfers the MSB must be set to 0 in order to avoid erroneous interrupt generation.

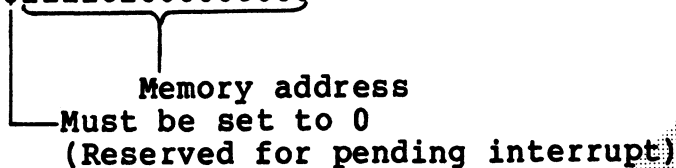
When issued to the disc or tape controllers, the DOB instruction signals the corresponding controller. The controller responds by performing an automatic block transfer of data as specified in the IOCB for that device. This data transfer may be one byte or an entire 256 byte block of data, depending on the starting and ending parameters supplied in the IOCB. An example for disc DMA operation would be:

IOCB Address: 075000 octal

Instruction: DOBC A1,50

Accumulator 1 Value:

A1 = 0111101000000000

  
Memory address  
Must be set to 0  
(Reserved for pending interrupt)

When issued to the multiplexer, the DOBS instruction sets the software pointer to the multiplexer IOCB but does not initiate any action on the part of the multiplexer. When the DOBS is complete, the software routine must issue DOA instructions to the port whose IOCB address was in the accumulator of the DOBS instruction. The DOA instruction will initiate DMA transfer between the CPU and the peripheral device.

### 6.2.3.3 Turning Off Automatic Block Transfer Mode For Disc or Tape

The DOB instructions with device code 77 may also be used to turn off automatic block transfer mode to disc and tape and/or reset the pending interrupt bit in the IOCB pointer. Both of these functions must be performed by the software for disc and tape transfers. To turn off automatic block transfer mode and reset the pending interrupt bit for disc or tape issue a DOBC or DOBP with the accumulator set to zero.

Instruction: DOBC A1, 50

Device Code: 5 = DISC

Accumulator 1 Value:

A1 = 0000000000000000

Turns off automatic transfer mode  
Resets pending interrupt

#### NOTE

It is important that the Most significant bit (MSB) be set to 0. If a DOBC or DOBP is issued with a value of 1 in the MSB, it will initiate processing of an interrupt which may or may not be pending for the controller whose IOCB address as in the accumulator.

### 6.2.3.4 Turning Off Automatic Block Transfer Mode For Multiplexer

More caution must be taken when using a DOBS to turn off automatic block transfer mode to the multiplexer and/or reset the pending interrupt bit for the multiplexer. If the accumulator contains a value of 0 when the DOBS is issued, the automatic mode is turned off for all ports, regardless of whether a port is in the process of performing a block transfer.

For a majority of operations the DOBS is used only to reset the pending interrupt bit. In this case, the contents of the accumulator specified in the DOBS instruction should be set to the address of the multiplexer IOCB pointer with the most significant bit set to 0.

**6.3 MUX GENERAL DESCRIPTION**

**TO BE SUPPLIED**



### 6.3.1 MUX HARDWARE CONFIGURATION

The POINT 4 MARK III combines four communications ports. These ports are RS-232 compatible, and can operate in either half or full-duplex mode. The baud rate for these ports is hardware-strappable.

#### 6.3.1.1 Baud-Rate Selection

The following baud rates may be selected for MARK III communications ports:

110  
150  
300  
600  
1200  
2400  
4800  
9600

Table 6-1 shows jumpering for each baud rate. Table 6-2 indicates port assignments for baud-rate headers.

**TABLE 6-1. BAUD RATE JUMPERING**

Baud Rate	Jumper
110	16 to 1
150	15 to 2
300	14 to 3
600	13 to 4
1200	12 to 5
2400	11 to 6
4800	10 to 7
9600	9 to 8

**TABLE 6-2. PORT ASSIGNMENTS FOR BAUD RATE HEADERS**

Header No.	Port No.
J16	0
J15	1
J14	2
J13	3

### 6.3.1.2 MARK III Cabling Connections

Cabling connections for MARK III interface with a CRT and line printer are indicated in Figures 6-6 and 6-7.

Figure 6-6 shows MARK III Asynchronous MUX/CRT cable connections.

Figure 6-7 shows MARK III Asynchronous MUX/line printer cable connections.

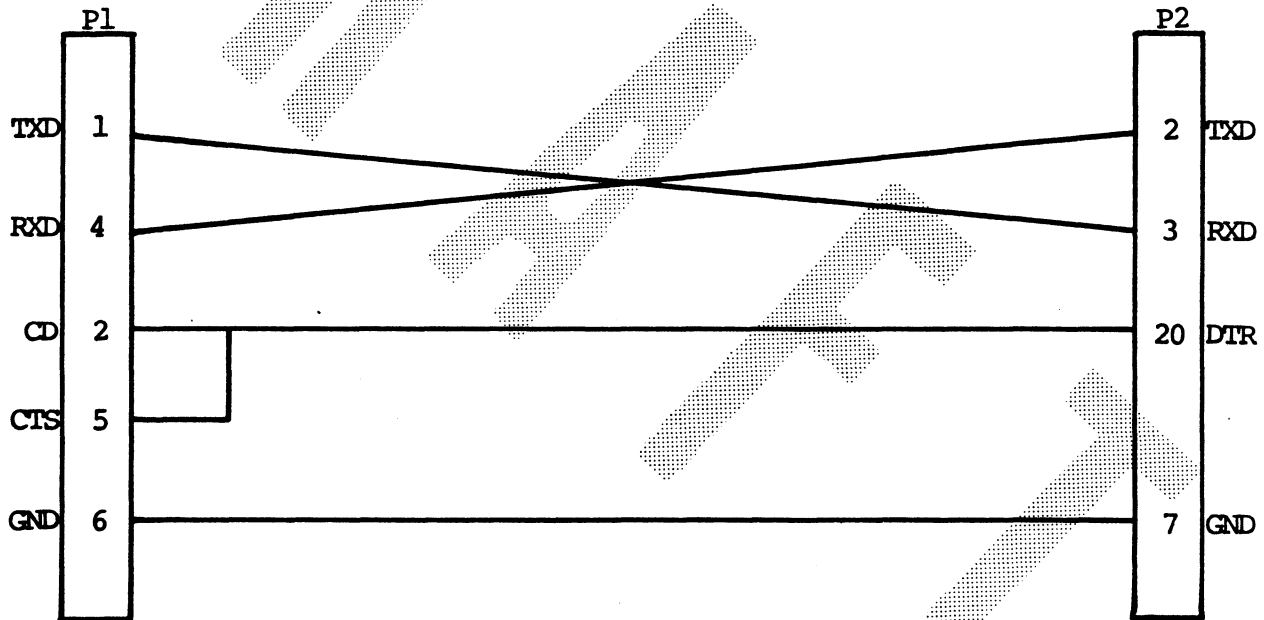
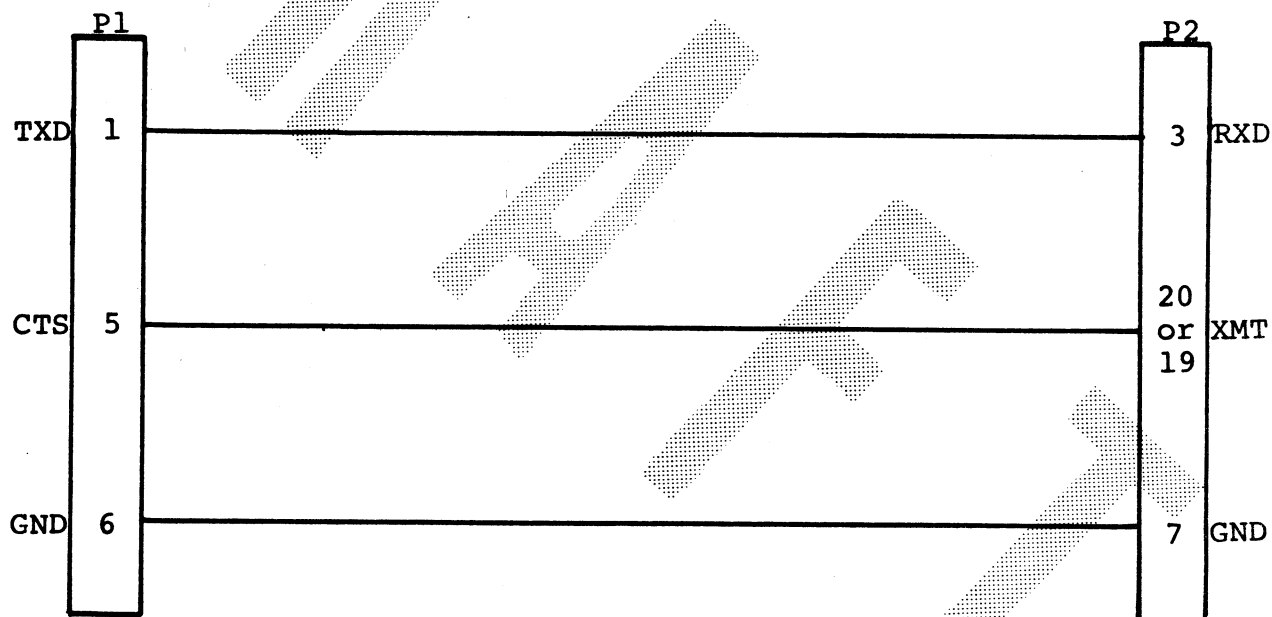


Figure 6-6. MARK III Asynchronous MUX/CRT Cable Connector Wiring



**Figure 6-7. MARK III Asynchronous MUX/Line Printer Cable Connector Wiring**

### 6.3.2 COMMUNICATIONS BYTE MODE

This section outlines programming protocols for the MUX controller on the Peripheral Interface Board (PIB), via the Communications Byte Mode and the Communications Auto Mode. Communications Byte Mode can operate in both DMA channel and Programmed I/O modes. Communications Auto Mode operates only in DMA channel mode.

Byte Mode topics include: Programming Communications Controller IC 6850; Operational Mode modifications; Command and Status Registers; and Software Polling.

Auto Mode topics include: MUX I/O Control BLOCK (IOCB); Control Word Definitions; Termination Status descriptions; and Initialization procedures.

The Communications Controller may be used in the byte mode. Command, status and data transfer are accomplished through programmed I/O, utilizing DOA and DIA instructions. A list of valid DOA and DIA device codes is shown in Table 6-3.

**TABLE 6-3. DOA, DIA DEVICE CODES**

Device Code (Octal)	Read (DIA)	Write (DOA)
	PORT 0	
10	Status Register	Command Register
11	Data Register	Data Register
	Port 1	
12	Status	Command
13	Data	Data
	Port 2	
14	Status	Command
15	Data	Data
	Port 3	
16	Status	Command
17	Data	Data



### 6.3.2.1 Communications Controller IC 6850

IC 6850 is an asynchronous communications chip with four internal registers. Two registers are read-only; two are write-only. The 6850 is totally programmed; in byte-mode operation, programming is directed by the software. An example of the 6850 programming process is provided below, using Port 0 and its associated device codes ( $10_8$  and  $11_8$ ).

### 6.3.2.2 Command Register

The following device code is used for the command register (CR):

DOA  $10_8$

Only the Least Significant Byte of the accumulator is used. The Most Significant Byte (Bits 0-7) is not used, and must be set to zero (0). Figure 6-8 shows the Command Register on Port 0. Bits, settings, and their functions are shown in Table 6-4.

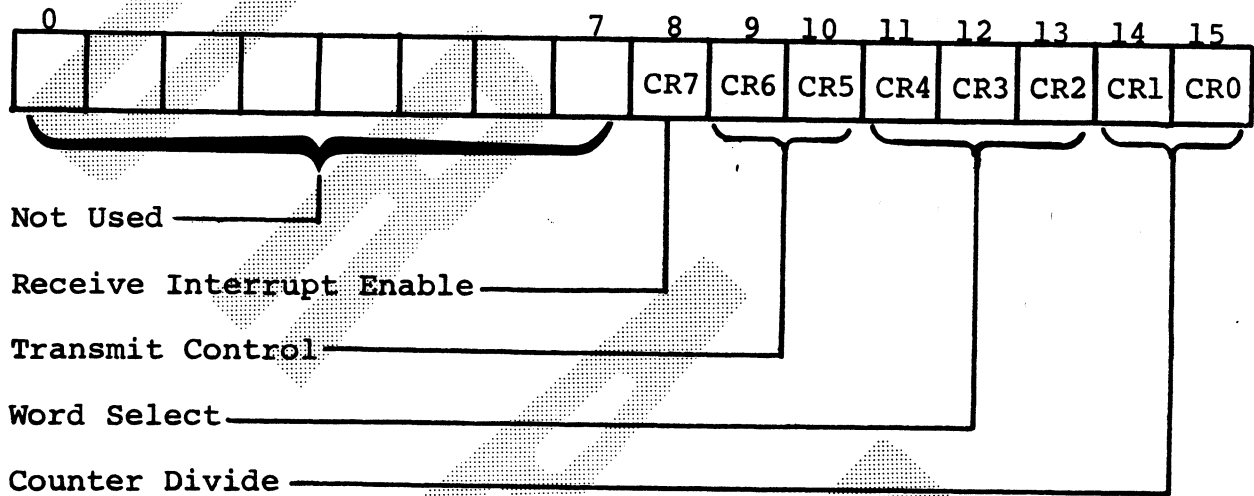


Figure 6-8. Command Register on Port 0

**TABLE 6-4. COMMAND REGISTER BIT FUNCTIONS**

Bit			Function
CR1	CR0 (LSB)		Counter Divide
0	0		Not Used
0	1		Divide-by-16
1	0		Not Used
1	1		Master Reset
CR4	CR3	CR2	Word Select
0	0	0	7 Bits + Even Parity + 2 Stop Bits
0	0	1	7 Bits + Odd Parity + 2 Stop Bits
0	1	0	7 Bits + Even Parity + 1 Stop Bit
0	1	1	7 Bits + Odd Parity + 1 Stop Bit
1	0	0	8 bits + 2 Stop Bits
1	0	1	8 bits + 1 Stop Bit
1	1	0	8 Bits + Even Parity + 1 Stop Bit
1	1	1	8 Bits + Odd Parity + 1 Stop Bit
CR6*	CR5		Transmit Control
0	0		Transmit Interrupts Disabled
0	1		Transmit Interrupts Enabled
	CR7 (MSB)		Receiver Interrupts
	0		Receiver Interrupts Disabled
	1		Receiver Interrupts Enabled

\*Transmit Control is not used; must be set to zero.

### 6.3.2.3 Standard Mode of Operation

During power-up initialization, the 6850 is reset, then programmed to a standard mode of operation. The POINT 4 MARK III is delivered with a standard mode of operation established for all ports. This standard mode is as follows:

1. Divide-by-16 Clock
2. 7 Bits + Even Parity + 1 Stop Bit
3. RTS = Low, Transmitting Interrupt Disabled
4. Receiving Interrupt Disabled

### 6.3.2.4 Operational Mode Modifications

For certain devices, it may be necessary to change the mode of operation set in the 6850 IC. This may be done by issuing the following two instructions:

1. Master Reset

DOA (a),  $10_8$  (a) =  $3_8$

2. Redefine Command Register

DOA (a),  $10_8$  (a) = 00000000X0XXXX01

User Defined

If there is a need to change the mode of operation, two important points should be considered:

1. IC 6850 uses a Divide-by-16 clock. If CR1 and CR0 are not set to Master Reset (1,1) or Divide-by-16 (0,1), erroneous data will result.
2. During the byte mode of systems operation, interrupts may be utilized. If interrupts are used, CR5 and CR7 must be programmed accordingly. If interrupts are not used, the status register may be polled to determine if the transmitter is empty, or if the receiver is full.

### 6.3.2.5 Status Register

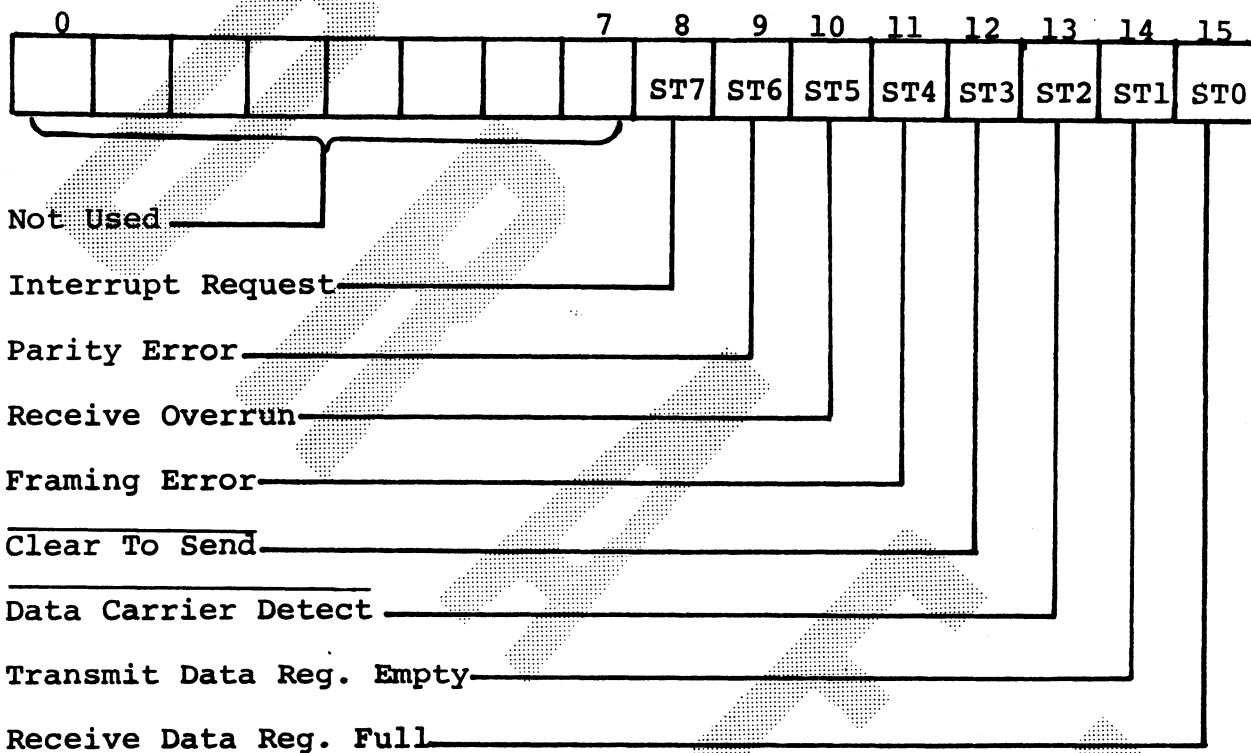
Software utilizes the Status Register during a polling sequence to determine the following:

1. Receive data register full
2. Transmit data register empty
3. Error condition

The following instruction is used to read the status register in Port 0:

DIA 10<sub>8</sub>

The status (ST) will be the Least Significant Byte of the accumulator. The Most Significant Byte is not used. Figure 6-9 shows the Status Register on Port 0. Bits, instructions, and operational descriptions are shown in Table 6-5.



**Figure 6-9. Status Register on Port 0**

**TABLE 6-5. STATUS REGISTER BIT OPERATIONS**

Bit	Instruction	Operation
15 (ST0)	Receive Data Register Full (RDRF)	Indicates that received data has been transferred to the Receive Data Register. Cleared by reading the Receive Data Register (DIA 11g) or by a Master Reset. Cleared state indicates that the Receive Data Register is not current or Data Carrier Detect is high.
14 (ST1)	Transmit Data Register Empty (TDRE)	Set high when Transmit Data Register contents have been transferred, and new data may be entered. Low state indicates either that Transmit Data Register is full, or a high on Clear to Send is inhibiting data transfer.
13 (ST2)	Data Carrier Detect (DCD)	Set high when a loss of carrier occurs. Generation depends upon port connection to Data Terminal Equipment (DTE). In a point-to-point connection, generated from the Data Terminal Ready (DTR-) of the DTE. In a modem connection, generated from the modem's Carrier Detect.  Inhibits the Receive Data Register Full status bit. May be reset by reading the status register and the data register, or by Master Reset. If it remains high after attempts to clear, DCD- will remain high, and follow the DCD- input.
12 (ST3)	Clear to Send (CTS)	The PIB does not use CTS- input; it is always tied low. Only set in the status register when a hardware malfunction occurs. If it goes high, it inhibits the Transmit Data Register Empty status bit.
11 (ST4)	Framing Error (FE)	When set high, indicates that received character is improperly framed by a start and stop bit. Indicates error as long as character is available.
10 (ST5)	Receiver Overrun (OVRN)	Error flag which indicates that one or more characters in the data stream have been lost, due to failure to read data in the Receive Data Register. Flag does not occur until a valid character prior to the Overrun has been read. Reset after reading the data from the Receive Data Register, or by a Master Reset.

**TABLE 6-5. STATUS REGISTER BIT OPERATIONS (Cont)**

Bit	Instruction	Operation
9 (ST6)	Parity Error (PE)	Present as long as a data character is in the Receive Data Register. If no parity is selected, both transmitter parity generator output and receiver parity check results are inhibited.
8 (ST7)	Interrupt Request (IRQ)	In byte mode, transmit and receive interrupts should be disabled to prevent use of this bit.
7	Secondary RTS (Port 3 Only)	When a line printer is connected to Port 3, indicates the status of the secondary RTS line. 1 = EIA positive (greater than or equal to +5V) 0 = EIA negative (less than or equal to -5V or open)



### 6.3.2.6 Software Polling

Figure 6-10 indicates the programming stages required to poll devices, in order to ensure that the data register is empty, and may receive another byte of data.

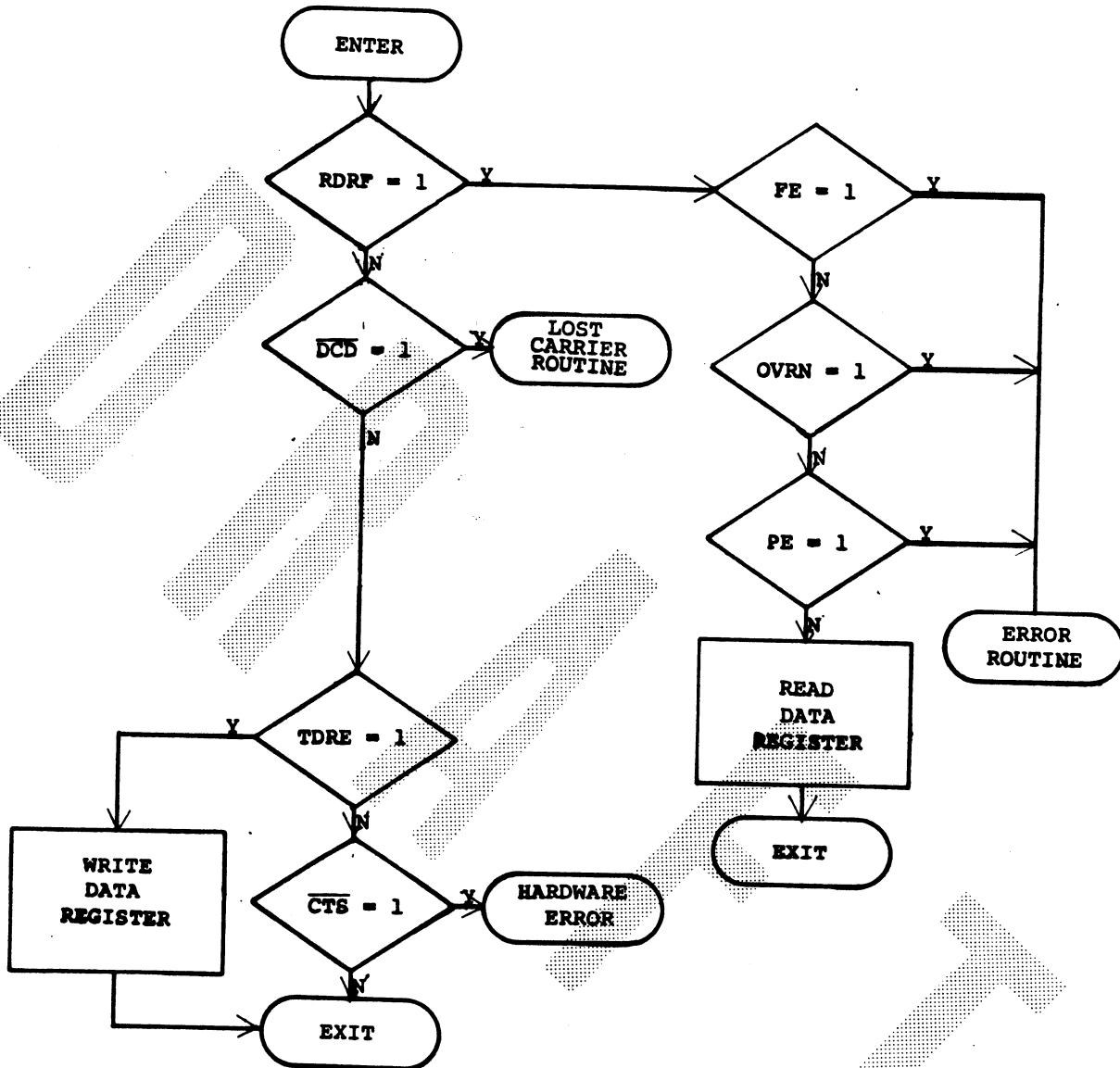


Figure 6-10. Software Polling Flowchart For Output Operations

### 6.3.3 COMMUNICATIONS AUTO MODE

#### 6.3.3.1 MUX I/O Control Block

The MUX Controller supplies software control over four communications ports and their parameters through programmed I/O. Software control includes enabling and disabling the receiver and transmitter, setting character length, determining parity tThe MUX I/electing the number of stop bits. A Real Time Clock on Port 0 is also enabled or disabled under software control.

Each port has an I/O Control Block (IOCB); the MUX Controller utilizes four IOCBs in the PIB.. The MUX I/O Control Blocks (IOCB) contain information which MUX uses for automatic data transfer. They do not contain data for selecting port parameters or enabling and disabling ports. All IOCBs must be contiguous in main memory, offset from one another by a multiple of 40 octal. Each IOCB contains an input and output section for each port.

#### 6.3.3.2 Control Word Definitions

Each IOCB consists of eight words. Six words are used for MUX control: 0, 1, 4, 5, 6, and 7. The three even-numbered words (0, 4, 6) control input; the three odd-numbered words (1, 5, 7) control output. Two words (2, 3) are not used.

The MUX IOCB is shown in Figure 6-11.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	IOC	In Mode	SC	Error Status			CAR	Input Character								
1	OOC	Out Mode	SC	0	0	0	RTC	Output Character								
2	Not Used															
3	Not Used															
4	Input Byte Pointer														Bin	
5	Output Byte Pointer														Bin	
6	Last Input Byte Pointer														Bin	
7	Last Output Byte Pointer														Bin	

Figure 6-11. MUX IOCB

### 6.3.3.3 Input Operations

The three even-numbered words of each IOCB (0, 4, 6) control input. Word 0 is the Input Control Word (ICW); word 4 is the Input Byte Pointer; word 6 is the Last Input Byte Pointer.

#### 6.3.3.3.1 INPUT CONTROL WORD

The components of word 0 are shown in Figure 6-12.

Table 6-6 defines the Input Control Word, indicating each bit, the appropriate instruction, and the resultant operation.

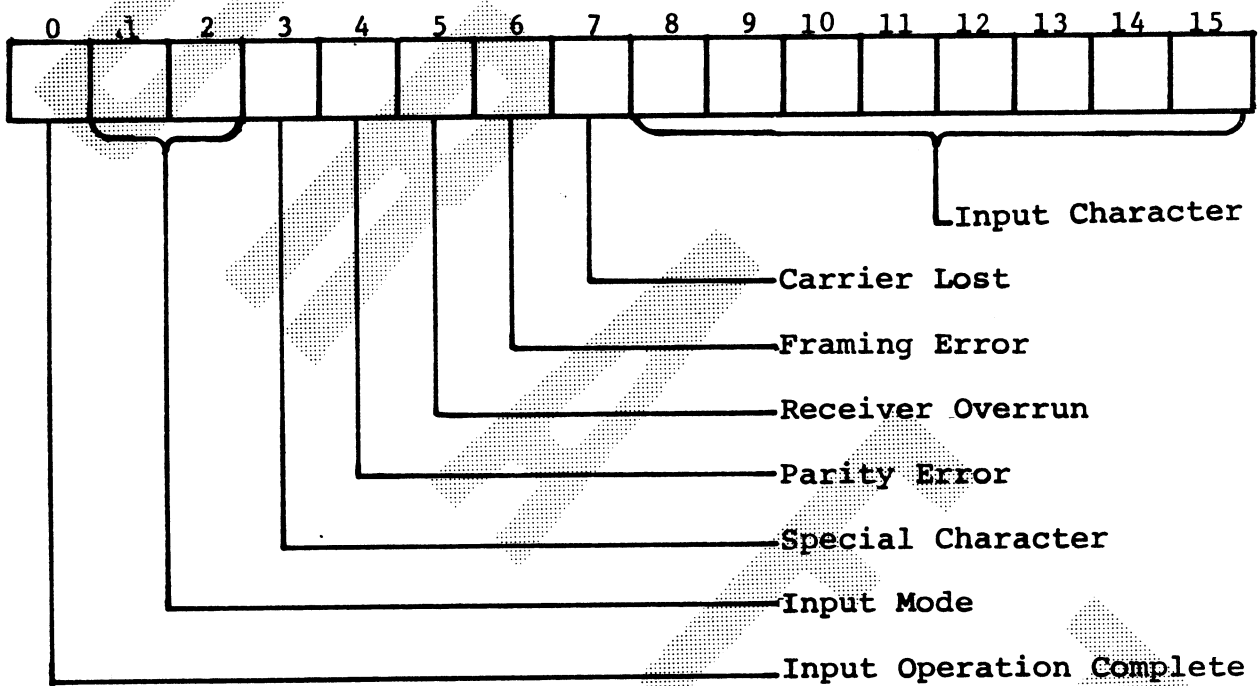


Figure 6-12. Input Control Word - Word 0

**TABLE 6-6. INPUT CONTROL WORD DEFINITIONS  
(WORD 0 OF EACH CONTROL BLOCK)**

Bit	Instruction	Operation
0	IOC (Input Operation Complete)	<p>Set to one (1) if any of the following conditions occurs:</p> <ol style="list-style-type: none"> <li>1. Input buffer is full</li> <li>2. The receiver detects:               <ol style="list-style-type: none"> <li>a. Parity error (if parity is not inhibited)</li> <li>b. Framing error (break detect)</li> <li>c. Receiver overrun (DMA is disabled for an extended period, or another controller uses DMA cycle for a similar period)</li> </ol> </li> <li>3. IOCB is in Single Character Input Mode, and an incoming character is received and stored in Bits 8-15 of Input Control Word.</li> <li>4. Special Interrupt Request is enabled and a special character is received (&lt;40 octal).</li> <li>5. A loss of Carrier occurs (Carrier Detect goes high) during a receive operation.</li> </ol> <p>At same time that IOC is set to one (1), controller sets Interrupt Pending Bit and Interrupt Request Flag.</p> <p>IOC must be cleared by MUX Interrupt Service Program. There is approximately one (1) character time to accomplish this.</p> <p>If another input character is received while in single input mode, it overwrites character in Bits 8-15.</p>
1 & 2	Input Mode	<p>Single Character Input Mode - each incoming character is placed in Bits 8-15 of ICW. IOC is set, interrupt pending bit is set, and an interrupt generated.</p>

**TABLE 6-6. INPUT CONTROL WORD DEFINITIONS (Cont)  
(WORD 0 OF EACH CONTROL BLOCK)**

Bit	Instruction	Operation
0	1	Not used (illegal)
1	0	Automatic Input - incoming characters are placed in input buffer defined by Input Byte Pointer (IBP) and Last Input Byte (LIB), if no interrupt conditions are encountered.
1	1	Automatic Input with Echo - same as Automatic input, except that each character placed in input buffer is also automatically echoed (output). Note that any character producing an interrupt (other than buffer full) is not automatically echoed.
NOTE		
When using Automatic Input with Echo mode, software must ensure that transmitter interrupt section is disabled for a specified port. If interrupt is not disabled, controller will echo character and transmit characters from output buffer. With transmitter interrupt disabled, controller will ignore output section of CIOB for specified port.		
3	Special Character	If this bit is set to one (1), controller examines each incoming character to determine if it is a Special Character (<40 octal). If the incoming character is a Special Character, controller stores the character in Bits 8-15 of Input Control Word, sets pending interrupt bit, and generates an interrupt request. The character will not be stored in input buffer. In addition, controller sets most significant bit of all incoming characters to one (1).
4	Parity Error	Set to one (1) when incoming character has a parity error. When this condition occurs. IOC Bit is set to one (1), and stores character in ICW.



**TABLE 6-6. INPUT CONTROL WORD DEFINITIONS (Cont)  
(WORD 0 OF EACH CONTROL BLOCK)**

Bit	Instruction	Operation
5	Receiver Overrun	Set to one (1) when controller cannot service receiving section of specified port before a character is over-written. This can occur if DMA is disabled for an extended period, or if another controller uses an excessive number of DMA cycles. In this condition, IOC Bit is set to one (1), and data in Receive Data Register is stored in ICW.
6	Framing Error	Set when an incoming character has a framing error (an improper number of start and stop bits). Sets IOC Bit to one (1) and stores incoming character in ICW.
7	Carrier Detect	Set to one (1) when a loss of Carrier occurs (Carrier Detect goes high.) When this condition occurs, receiver section is disabled. Sets IOC bit to one (1) and stores data in Receive Data Register in ICW.
8-15	Input Character	<p>Each incoming character which causes IOC Bit to be set is stored in these bits, which include:</p> <ol style="list-style-type: none"> <li>1. The last character stored in input buffer, causing an input buffer-full condition.</li> </ol> <p style="text-align: center;"><b>EXCEPTION</b></p> <p>If an input buffer ends in mid-word, byte placed in ICW is right half of last buffer word, and not last incoming character. Software must check for buffer full before it checks for Special Character.</p> <ol style="list-style-type: none"> <li>2. An incoming character which is overrun, or which has caused a parity or framing error.</li> <li>3. An incoming character which is a Special Character.</li> <li>4. A character in Receive Data Register of 68B50 when Carrier Detect goes high.</li> </ol>

### 6.3.3.3.2 INPUT TERMINATION STATUS

When an input operation is completed, the appropriate status is written into the ICW by the controller.

When the status is written, the Input Operation Complete Bit (IOC) is set to one (1) The incoming character which completed the operation is written in the input character of Word 0. The controller then sets the Pending Interrupt Bit and the Interrupt Request Flag.

When the interrupt occurs, the software has approximately one character time to determine which port's input operation is complete, and to perform one of the following services:

1. Disables the receiver section of the corresponding port through programmed I/O.

#### NOTE

If the receiver section is disabled, the controller will not transfer any incoming characters to a buffer. But it will cause any incoming characters to be overrun if the receiver section is disabled for an excessive period of time.

2. Re-initializes the CIOB after interpreting the status word by creating a new input memory buffer address and resetting the IOC Bit.

If neither service is performed, a MUX overrun may occur.

### 6.3.3.3.3 INPUT BYTE POINTER

Word 4 of the IOCB is the Input Byte Pointer (IBP). IBP and LIB (Last Input Byte) are only used in the automatic Input mode. IBP must be set by the program to one (1) less than the first byte address of the automatic input buffer. Each time the MUX stores an incoming byte, it will increment IBP. The MUX uses the most significant bit (bit 15) as the Byte Indicator (BIN). The byte will be stored in the left half of the word addressed if BIN = 1; it will be stored in the right half if BIN = 0. IBP always points to the last input byte stored. If IBP > LIB, an interrupt will be generated by the first incoming character, which will be stored at IBP.

### 6.3.3.3.4 LAST INPUT BYTE (LIB)

Word 6 of the IOCB is the Last Input Byte (LIB). It is set up by the program to the last byte address of the auto input buffer. The MUX will generate an Input Done interrupt when a byte is stored at this address. IBP will then be equal to LIB.

### 6.3.3.4 Output Operations

The three odd-numbered words of each IOCB (1,5,7) control output. Word 1 is the Output Control Word (OCW); word 5 is the Output Byte Pointer; word 7 is the Last Output Byte Pointer.

#### 6.3.3.4.1 OUTPUT CONTROL WORD

The components of word 1 are shown in Figure 6-13.

Table 6-7 defines the Output Control Word, indicating each bit, the appropriate instruction, and the resultant operation.

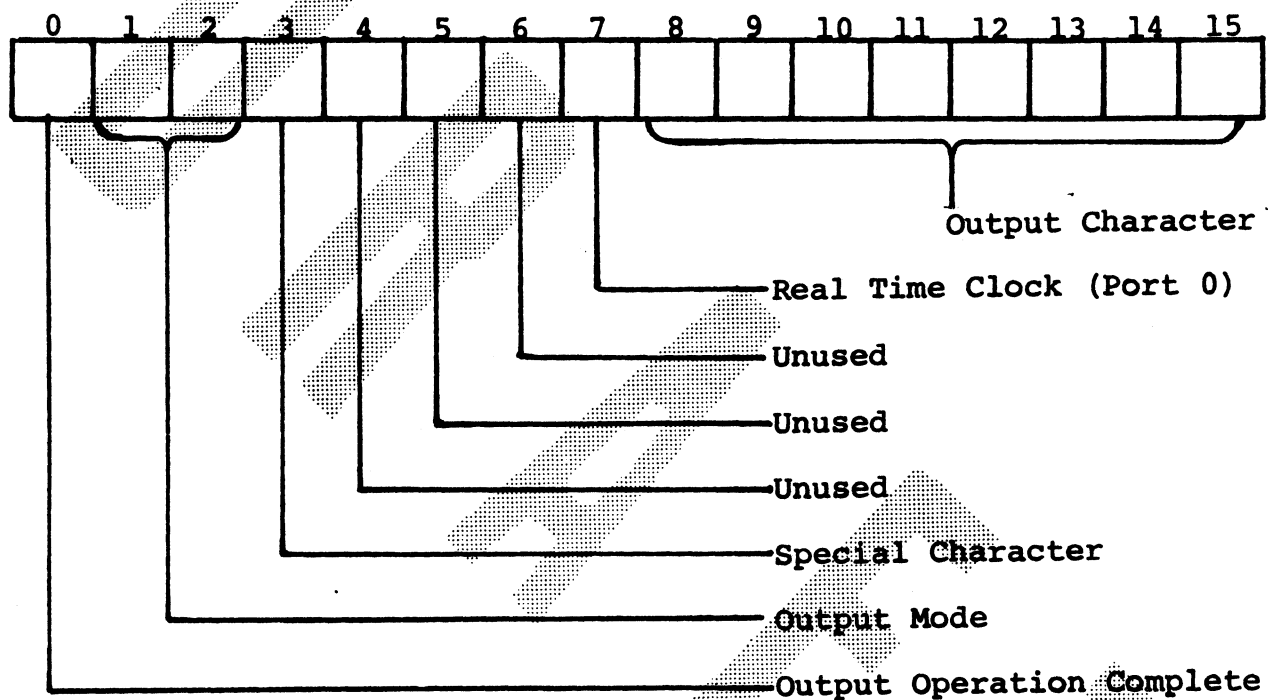


Figure 6-13. Output Control Word - Word 1

**TABLE 6-7. OUTPUT CONTROL WORD DEFINITIONS  
(WORD 1 OF EACH CONTROL BLOCK)**

Bit	Instruction	Operation
0	OOC (Output Operation Complete)	<p>Set to one (1) if the operation has been terminated due to one of the following conditions:</p> <ol style="list-style-type: none"> <li>1. If MUX is in Single Character mode when the output byte is read for transmission to the port. If the output control words are reloaded within one-to-two character times, uninterrupted output will result.</li> <li>2. If MUX is in Automatic Buffer Output mode when the last byte of the automatic buffer is read for transmission. If the output control words are reloaded within one-to-two character times, uninterrupted output will result.</li> <li>3. If MUX is in Automatic Output with Special Character mode, and if the transmitted character is a Special Control Character. Automatic output is terminated after transmission of the Special Character.</li> </ol> <p>The character which completed the operation is stored in the OCW. The controller sets the Pending Interrupt Bit and the Interrupt Request Flag.</p> <p>The OOC Bit must be cleared by the MUX Interrupt Service Program. If the CIOB is initialized within one-to-two character times, uninterrupted transmission will result. When the last byte of the output buffer is transferred, OOC is the only bit set.</p>

**TABLE 6-7. OUTPUT CONTROL WORD DEFINITIONS (Contd)  
(WORD 1 OF EACH CONTROL BLOCK)**

Bit	Instruction	Operation
1 & 2	Output Mode	Written by the program and read by MUX to determine the type of output.
0		Not used (Illegal)
1		Automatic Buffer output - characters are automatically transmitted from the output buffer, as defined by the Output Byte Pointer (OBP) and the Last Output Byte Pointer (LOBP).
1 0		Not Used (Illegal)
1 1		Single Character Output - the data byte in Bits 8 - 15 of the OCW is transmitted, and the OOC Bit is set.
3	Special Character	Used only in the Automatic Output Mode. If this bit is set to one (1), MUX tests each outgoing character to determine if it is a Special Character (>40 octal). If it is, MUX transmits the character, sets the C Bit in the Output Termination Word to one (1), and stores the Special Character in Bits 8 - 15 of the OCW. It then sets the Pending Interrupt Bit and generates an Interrupt Request. Automatic output is terminated.
4 - 6		Unused. Should be set to zero.
7	Real Time Clock	Used only on Port 0.  Set to one (1) if the software has enabled the Real Time Clock and the controller has been interrupted by the clock. (The Real Time Clock interrupts once each 10 msec.)  The Interrupt Pending bit is set, and an interrupt request is generated.  Does not set the OOC bit, or terminate automatic operation.

**TABLE 6-7. OUTPUT CONTROL WORD DEFINITIONS (Contd)  
(WORD 1 OF EACH CONTROL BLOCK)**

Bit	Instruction	Operation
8-15	Output Character	<p>In single mode, this byte is sent to the port for transmission.</p> <p>In automatic mode, each outgoing character which causes the OOC Bit to be set is stored in this byte, when the following conditions occur:</p> <ol style="list-style-type: none"> <li>1. An outgoing character is a Special Character, and the Special Character enable bit is set.</li> <li>2. The outgoing character is the last character in the output buffer.</li> </ol>



#### 6.3.3.4.2 OUTPUT TERMINATION STATUS

When an output operation is completed, the appropriate status is written into the OCW by the controller.

When the status is written, the Output Operation Complete Bit (OOC) is set to one (1). The outgoing character which completed the operation is written in Bit 8 - 15 of Word 1. The controller then sets the Pending Interrupt Bit and the Interrupt Request Flag.

When the interrupt occurs, the software has approximately one character time to determine which port's output operation is complete, and to perform one of the following services:

1. Disables the transmitter section of the corresponding port through programmed I/O.

#### CAUTION

Exercise caution when using this method, because the transmitter double-buffers characters. Software may disable the transmitter interrupt, but must not issue a MASTER RESET until the buffered characters have been transmitted.

2. Re-initializes the CIOB after interpreting the status word by creating a new input memory buuffer address and resetting the IOC Bit. This must be effected within two character times.

If neither function is completed, MUX will be inhibited from performing further output transfers, because MUX looks at the OOC Bit in the status word before performing any output transfer. If OOC is set to one (1), MUX will not perform an output transfer. MUX will attempt to perform an output transfer after each executed instruction, which will create a system slow-down.

#### 6.3.3.4.3 OUTPUT BYTE POINTER

Word 5 of the IOCB is the Output Byte Pointer (OBP). OBP and LOB (Last Output Byte) are used only in the Automatic Output mode. OBP must be set up by the program to one (1) less than the first byte address of the automatic output buffer. Each time the MUX is ready for an output byte, it will increment OBP. The MUX fetches the byte for transmission from the appropriate half of the word address given by the most significant 15 bits of the OBP (word address) in conjunction with the Byte Indicator (BIN). OBP always points to the last byte transmitted.

#### 6.3.3.4.4 LAST OUTPUT BYTE

Word 7 of the IOCB is the Last Output Byte (LOB). Set by the program to the last byte address of the automatic output buffer. When the byte at that address is picked up for transmission, the MUX will generate an Output Done interrupt (OBP will then be equal to LOB). An automatic buffer may contain as little as one byte. In this case, initially  $(LOB) = (OBP) + 1$ . If initially  $(LOB) < (OBP) + 1$ , the MUX will transmit one byte from  $(OBP) + 1$ , set ODN and produce an interrupt.

### 6.3.3.5 Initialization Procedures

This section covers set-up procedures for the MUX controller, and includes the following steps: setting the pointer; initialization guidelines; enabling and disabling ports and defining port parameters; the printer port; and enabling and disabling transmitters and receivers.

#### 6.3.3.5.1 SETTING THE POINTER

All port IOCBs must be contiguous in main memory. Software only sets up IOCBs for ports that are to be enabled, but must reserve space for all port IOCBs in the contiguous memory block.

1000	PORT 0
8	
1040	PORT 1
8	
1100	PORT 2
8	
1140	PORT 3
8	

When MUX is in Automatic Transfer mode, the controller must be informed of the CIOB block starting address by issuing a DOBS instruction with the contents of the accumulator containing the address. See the instruction below.

DOBS (a),77<sub>8</sub>

(a) = 1000<sub>8</sub>

When the controller receives a DOBS, the memory address is stored in a register. No other activity occurs (unlike the Tape and Disc Controller) because software has control over enabling and disabling of each port through programmed I/O.

### 6.3.3.5.2 INITIALIZATION GUIDELINES

Four basic enabling and disabling considerations are listed below:

1. If an input or output section of a CIOB has not been previously set up by software, the corresponding receiver or transmitter of the port should not be enabled.
2. If a port is to operate in Automatic Echo mode for incoming character, the transmitter section of that port must be disabled.
3. Software may disable the receiver section of a port after the input operation is complete. However, there is a risk of having an overrun condition on any further incoming characters.
4. When an output operation is complete, there is approximately one character time for software to either update the IOCB for a new transmission, or disable the transmitter with a DOA instruction.

### 6.3.3.5.3 ENABLING/DISABLING PORTS AND PORT PARAMETERS

After software has set the IOCBs and issued a DOBS-,10 instruction, corresponding ports must be enabled. A list of instructions used for enabling and disabling ports follows:

DOA (a),10 <sub>8</sub>	PORT 0
DOA (a),12 <sub>8</sub>	PORT 1
DOA (a),14 <sub>8</sub>	PORT 2
DOA (a),16 <sub>8</sub>	PORT 3

The accumulator contains the necessary information for port programming. In the programming process, the following three steps are required:

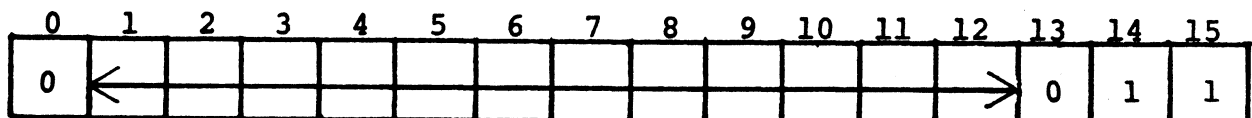
1. When the system powers up, the following parameters are standard.
  - a. Receiver interrupts disabled
  - b. Transmitter interrupts disabled
  - c. Word length = 7 bits
  - d. Even parity
  - e. 1 stop bit

Software can change any of these parameters using a DOA instruction.

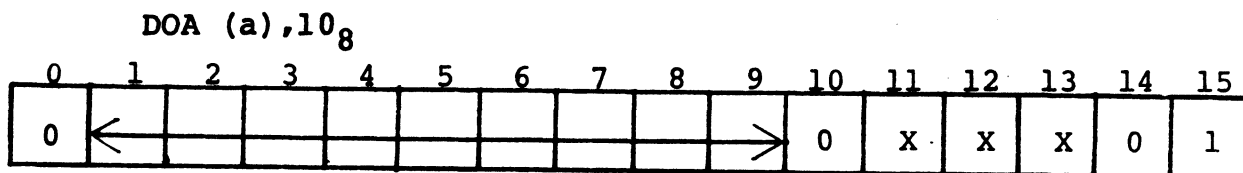
2. Software may re-initialize the port at any time by issuing a Master Reset DOA. It should be used only to initialize the MUX, because receive, transmit and status data are lost. The Master Reset should never be used once the port has been initialized and continuous operations are to be performed. The Master Reset DOA instruction and a port re-initialization example are shown below.

DOA (a),10<sub>8</sub>

(MASTER RESET FIGURE HERE)



3. Software can now program the word length, parity, and stop-bit parameters, using another DOA instruction. The parameter instruction and parameter example are shown below.



Port Characteristics  
Which User Can Define

Bits 11, 12, and 13 should be set to the same values as defined during initialization.

These steps should be followed for all ports when first initializing MUX. The steps should never be used after initialization.

NOTE

Port 3 contains an additional status line for use with a line printer. Any status change on this line is handled by the controller, and is totally transparent to the software.

6.3.3.5.4 ENABLING/DISABLING TRANSMITTERS AND RECEIVERS

The receiver and transmitter sections for each port operate independently, and must be enabled or disabled independently. The instruction and Enable/Disable example are shown in Figure 6-14.

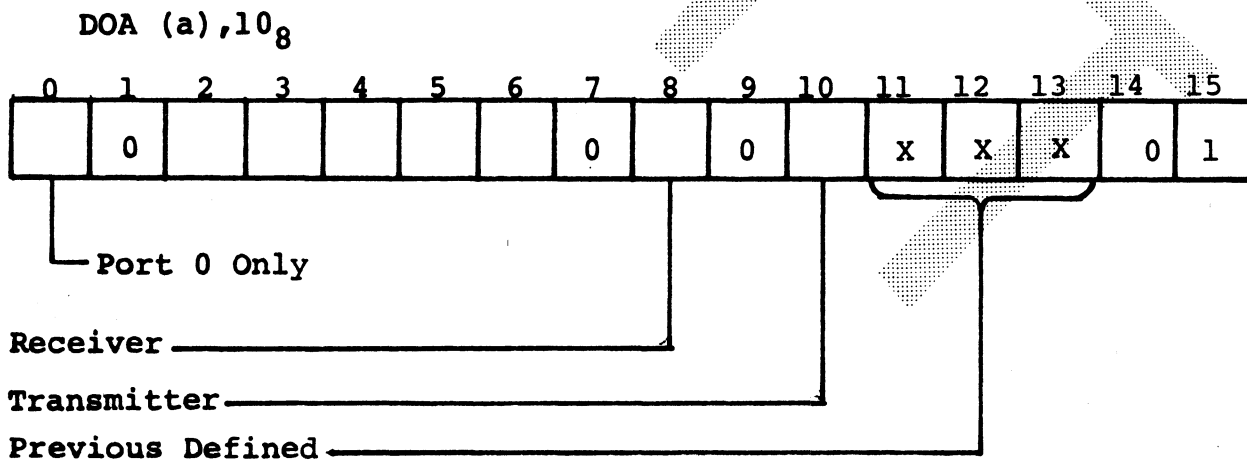


Figure 6-14. Transmitter/Receiver Example

Each DOA instruction updates the port command register. If previous values have been defined (such as port parameters), they should be used on all of the following DOA instructions for the specified port as indicated in Table 6-8.

**TABLE 6-8. ENABLING/DISABLING OPERATION**

Bit	Name	Operation
0	Real-Time Clock Enable (Port 0 Only)	0 = Disable Real-Time Clock 1 = Enable Real-Time Clock
8	Receiver	0 = Disable Receiver Section 1 = Enable Receiver Section
10	Transmitter	0 = Disable Transmitter Section 1 = Enable Transmitter Section
11-13	Port Parameters	Set to Values Defined During Initialization

The MUX controller uses an Attention Flag for servicing ports in the Automatic mode. Bits 8 and 10 of the accumulator allow a port's receiver or transmitter to generate the Attention Flag.

#### 6.3.3.5.5 POLLING MUX INTERRUPT

The following instruction is used to poll the MUX Interrupt Pending bit.

DIBS (a),77<sub>8</sub>

This instruction causes the controller to transfer the contents of the MUX accumulator (Interrupt Pending bit + IOCB memory address) into acc. Software may then test the Most Significant Bit (Interrupt Pending bit).

The Interrupt Pending bit is reset by resetting the Most Significant bit in acc and issuing the following instruction:

DOBS (a),77<sub>8</sub>

#### 6.3.3.6 Deactivating MUX

The following instruction is used to deactivate the MUX controller. The accumulator must be equal to 0.

DOBS (a),77<sub>8</sub>

This instruction causes MUX to ignore all ports.



#### 6.4 SMD/CMD DISC INTERFACE

The POINT 4 MARK III SMD/CMD disc controller module is designed to handle up to two drives. The module provides high speed, direct-memory access between the disc drives and the CPU. The following are features of the drive:

- o Handles SMD/CMD drives that support sector mark
- o DMA transfer rates up to 1.25 megabytes per second
- o Complete software control of sector addressing
- o Full interrupt capabilities
- o Read - Verify data operations
- o Status and error reporting on completion of operation

#### 6.4.1 PERFORMANCE CHARACTERISTICS

Drives per Controller: Two

Drive Type: SMD/CMD drives supporting the sector mark interface signal

DMA Transfer Rate: 1.25 Megabytes per second

Sector Size: Header - 4 words  
Data - 256 words

Error Detection: Yes

Controller Device Code: 50

Drive Port Assignments:

Port A-J8

Port B-J7

I/O Instructions:

Input - DIA

OUTPUT - DOA

Device Codes -

10-17 = MUX Ports 0-3

50-55 = Disc

60-62 = Tape

DMA (Device Code 77) -

DOBS/DIBS = MUX

DOBC/DIBC = Disc

DOBP/DIBP = Tape

CPU Functions (Device Code 77) -

SKPBN = Skip if Interrupts Enabled

SKPDN = SKIP if Power Fail Detected

SKPBZ/SKPDZ = Opposite of SKPBN, SKPDN

#### 6.4.2 DRIVE REQUIREMENTS

For proper disc interface, the controller requires that the disc drive be equipped with the following items:

- o Power Supply
- o Phase-lock Data Separator
- o Sector Mark Circuitry
- o Daisy Chain Control Bus Cabling
- o Ribbon Cable Interface

#### 6.4.3 MULTI-DRIVE CONNECTION

The disc controller will support two drives which need not be of the same type (SMD or CMD) or of the same manufacturer. The two drives must be connected in daisy chain fashion as illustrated in Figure 6-15.

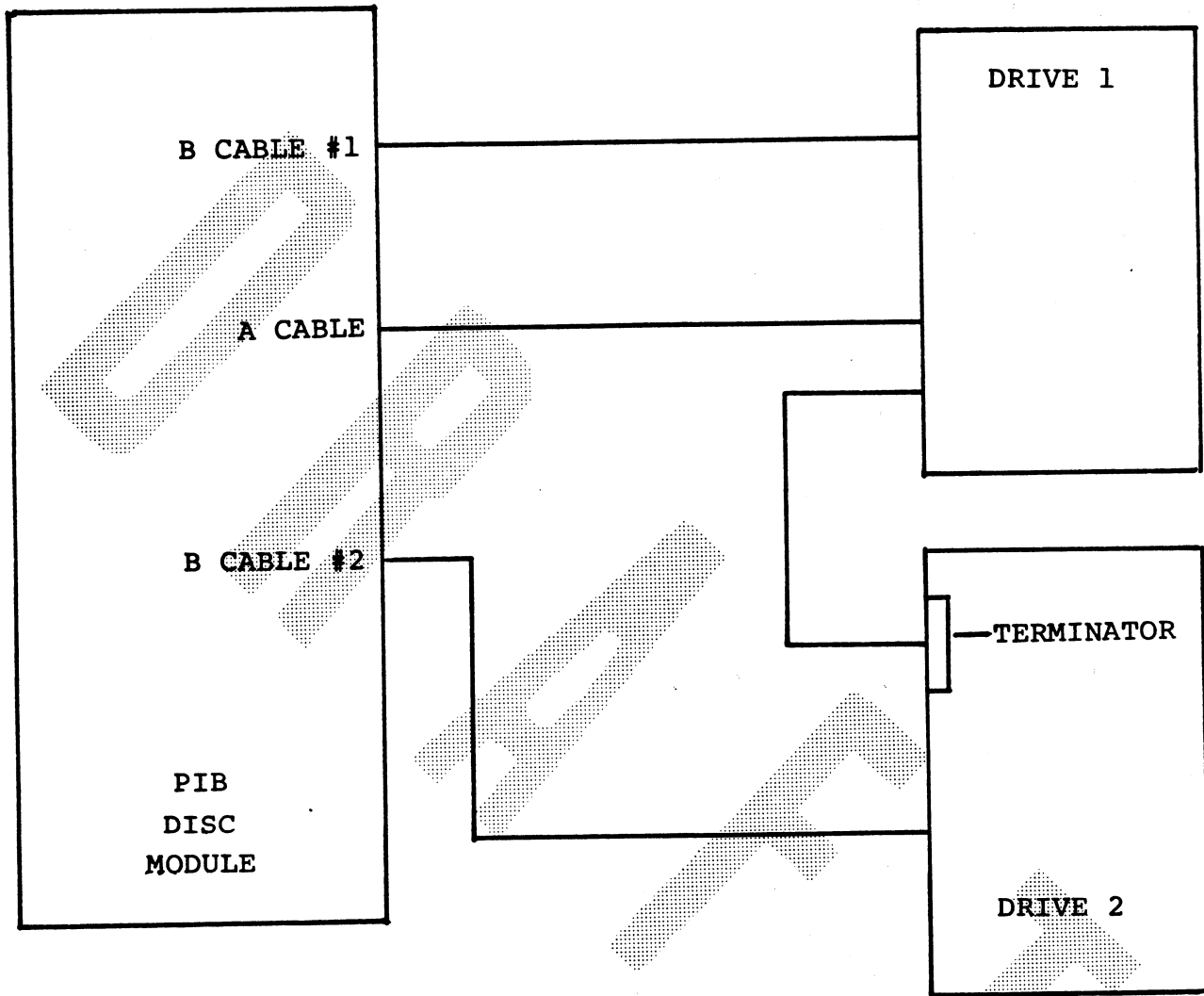


Figure 6-15. Daisy Chain Drive Connection

**6.4.4 OPERATION**

**TO BE SUPPLIED**

**SECRET**

#### 6.4.4.1 Sector Format

The sector is divided into four fields, address, address CRC, data and data CRC. Each sector has a total of 262 words. Their contents are defined as follows:

- I. Address field (4 words)
  - A. Current Sector Address (2 words)
    - 1. Current Cylinder (12 bits)
    - 2. Current Sector (8 bits)
    - 3. Current Head (5 bits)
  - B. Next Sector Address (2 words)
    - 1. Next Cylinder (12 bits)
    - 2. Next Sector (8 bits)
    - 3. Next Head (5 bits)
- II. Address Cyclic Redundancy Code (1 word)
- III. Data Field (256 words)
- IV. Data Cyclic Redundancy Code (1 word)

Sector addressing is established when the pack is formatted. Formatting software determines both the number of sectors per track and how the sectors are linked (see Subsection ).

#### **6.4.4.2 Sector Verification**

Before performing any read or write operation, the controller must verify that the correct sector has been located. The current address in the header is compared with the cylinder, head and sector address required for the read/write operation. If an address match is not obtained, or a cyclic redundancy code (CRC) error is detected while reading the sector address header, the controller will terminate the operation. A No IO Compare error will be reported to the processor.

#### **6.4.4.3 Data Transfer**

The disc controller read, write and read verify commands will cause one sector (256 words) to be transferred per operation. Read regardless commands are used by the software for error recovery and identification of bad tracks. For a read regardless, the number of sectors transferred is dependent on the sector count in the IOCB. Each sector transferred will contain the entire 262 word sector content, including header and CRC fields.

#### **6.4.4.4 File Linking**

When alternate track addressing is required, the location of the alternate cylinder will be specified in the address header of the desired sector. The controller will select the same head and sector on the alternate cylinder and overwrite the cylinder select information in the IOCB with the alternate cylinder address.

#### **6.4.4.5 Data Verification**

The read verify command allows checking of written data without performing a data transfer. If a cyclic redundancy code (CRC) error is detected, it will be reported to the processor.

#### **6.4.4.6 Error Checking and Status Reporting**

The controller recognizes two types of errors: disc interface errors and disc data errors. Disc interface errors are associated with disc or DMA interface, including such errors as drive fault, seek errors and format error. Data errors occur when the CRC does not match in either the address or data fields.

If any error is detected, the drive status will be reported to the CPU in a termination status word at the end of the IOCB. The termination status word (word 7 of the IOCB) is transferred at the end of each operation, whether successful or not.

#### **6.4.4.7 Interrupt Operation**

The disc controller generates an interrupt after completion of an operation whether successful or not. For description of interrupt servicing see Subsection 6.2.1.2 on Programming of Polling and Interrupts.

#### **6.4.4.8 Initiating an Operation**

Before a disc operation can be initiated, a software routine must check the controller to verify that a previous operation is not still in process. To do this use a DIBC (a),77 octal instruction and check the contents of the accumulator for a 0 value. If a non-zero value is in the accumulator, the controller is busy and a new operation must not be initiated.

If the controller is idle (0 value in the accumulator) a new operation may be initiated. To start the controller on a new operation, use a DOBC (a),77 octal instruction with the contents of the accumulator equal to the starting memory address of the IOCB.



#### **6.4.4.9 Deactivating The Controller**

When an operation is complete, the controller will set to 1 the most significant bit (MSB) in the IOCB pointer for the disc. The controller also writes a termination status word into the IOCB and generates an interrupt (if interrupts are enabled). The IOCB pointer is read by the DIBC instruction to determine if the MSB (pending interrupt bit) is set to 1, indicating the need to service an interrupt. The disc handling routine issues a DOBC (a),77 octal with the accumulator equal to 0. This instruction resets the pending interrupt bit and puts the controller in an idle state.

#### **6.4.4.10 Seek Control**

Software routines must control the task of seeking to a cylinder. The unit and sector select information in the IOCB is used to issue a series of TAG2 - TAG1 - TAG2 commands to the disc drive

#### **6.4.4.11 Seek Error Recovery**

A seek error will occur if one of the following condition exists:

- o The drive was unable to complete the seek within 500 milliseconds
- o The carriage on the drive has moved outside the recording field
- o The carriage has received an illegal track address

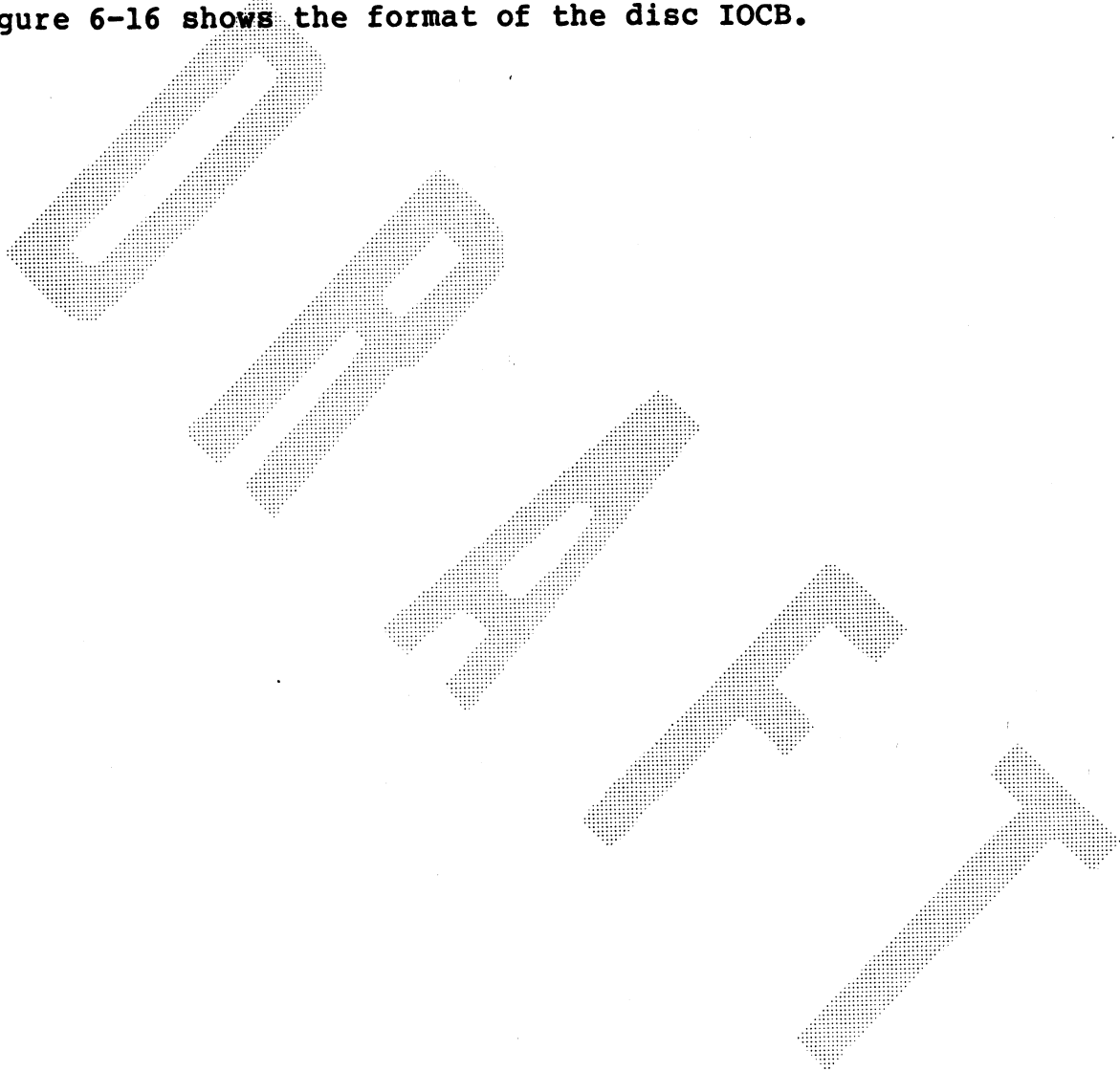
Upon detection of a seek error the controller will terminate the operation and will write the termination status word into the IOCB. The IOCB flags a seek error and generates an interrupt (if interrupt are enabled).

The software must clear the seek error. The seek error is cleared by issuing a Return to Zero command and a TAG3 command to the disc drive.

#### 6.4.5 THE INPUT/OUTPUT CONTROL BLOCK (IOCB)

The Input/Output Control Block (IOCB) for the disc controller consists of 7 words located in Main Memory and reserved for holding information about a disc transfer. The seven words of the IOCB are loaded with information by the software before the operation begins.

An additional word must be reserved for the termination status word immediately following the IOCB. Upon completion of an operation (successful or not), the controller writes status information on the drive into the Termination Status Word. Figure 6-16 shows the format of the disc IOCB.



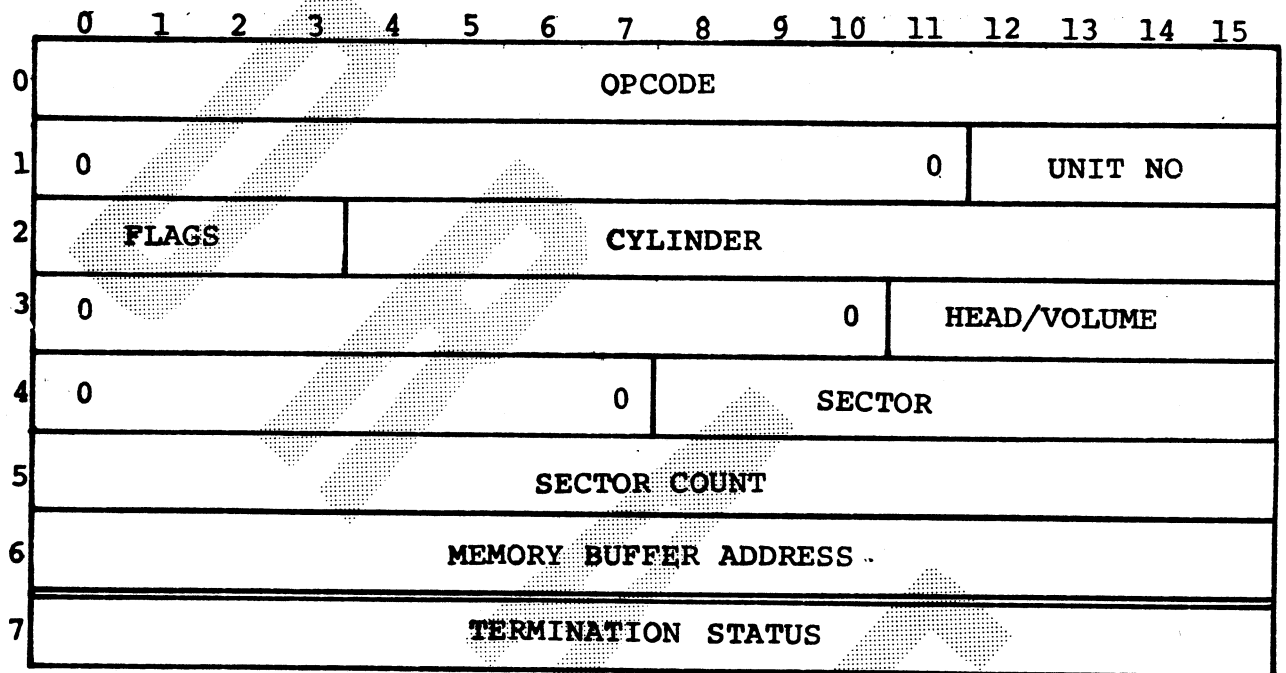


Figure 6-16. Disc IOCB Format

### 6.4.5.1 Opcode (Word 0)

Word 0 of the IOCB is used to define five operations which the disc may perform. The following is the format of word 0 of the disc IOCB:

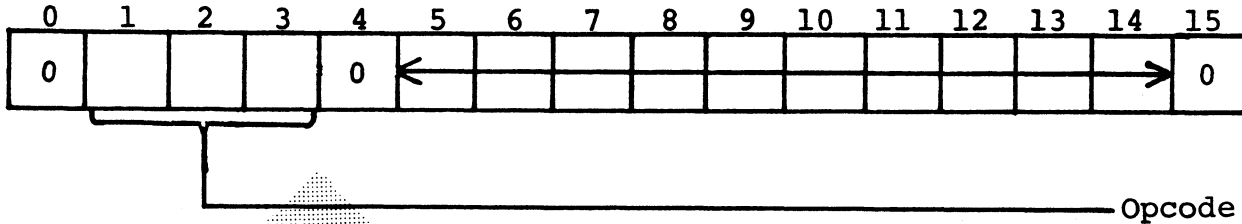


Table 6-9 defines the bit usage in word 0.

**TABLE 6-9. DISC IOCB WORD 4 DEFINITION**

Bits	Definition
0	Not used - Must be set to 0
1-3	Opcode - defines the following operations depending on their octal setting:
Opcode Word (Octal)	Operation
60000	Format
20000	Write Data
10000	Read Data
50000	Read Regardless
00000	Read Verify
4-15	Not used - Must be set to 0

Note that unused opcodes are reserved and should not be used. If used, no operation will take place and an error will be written into the Termination Status Word.

### 6.4.5.2 Unit Select (Word 1)

Word 1 is used by the software to specify the drive number on which to perform the operation. The following is the format of word 1 of the disc IOCB:

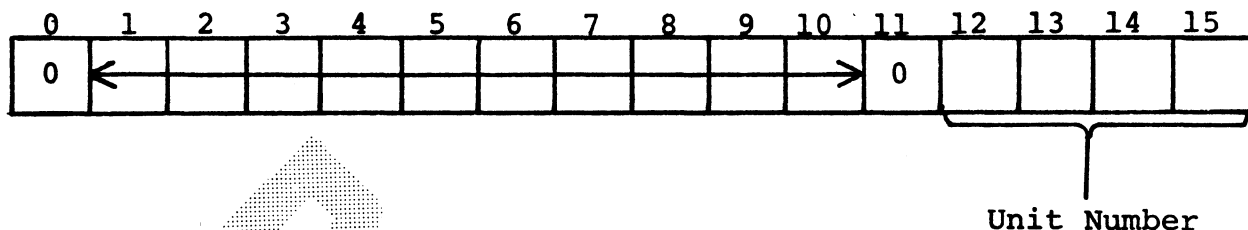


Table 6-10 defines the bit usage in word 1.

**TABLE 6-10. DISC IOCB WORD 1 DEFINITION**

Bits	Definition
0-11	Not used - Must be set to zero
12-15	Represents the binary equivalent of the drive number

### 6.4.5.3 Cylinder Select (Word 2)

Word 2 of the IOCB is used to specify the starting cylinder address for each operation. Procession software uses this word to select a cylinder address in the disc drive. The controller uses this word to compare with the current cylinder address in the sector address header for address verification. The following is the format of word 2 of the disc IOCB:

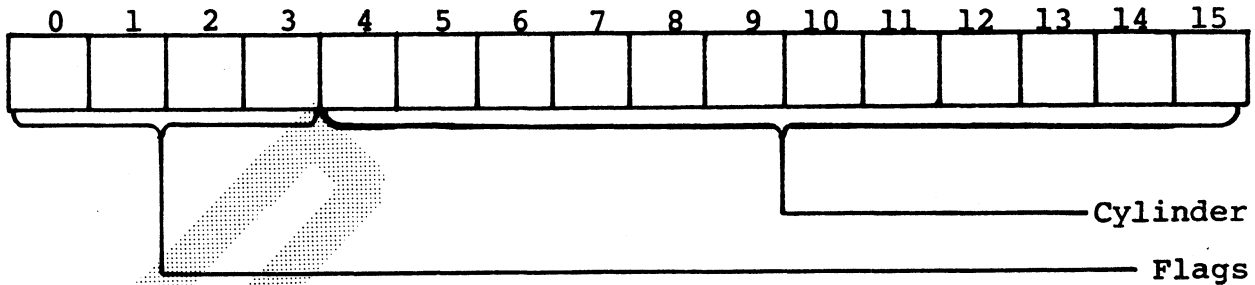


Table 6-11 defines the bit usage in word 2.

**TABLE 6-11. DISC IOCB WORD 2 DEFINITIONS**

Bits	Definition
0-3	File protect flags
4-15	Cylinder address

If the file protect flags were set for this cylinder by the software at format time, they should also be set in the IOCB cylinder select word. All sixteen bits of the cylinder select word are compared with the current cylinder address found in the sector header. If there is no match, a No IO compare error will occur and the operation will not be performed.

#### 6.4.5.4 Head Select (Word 3)

The head select word is used by processor software to issue the head/volume number to the drive for a seek and by the controller during sector verification. The following is the format of the head select word:

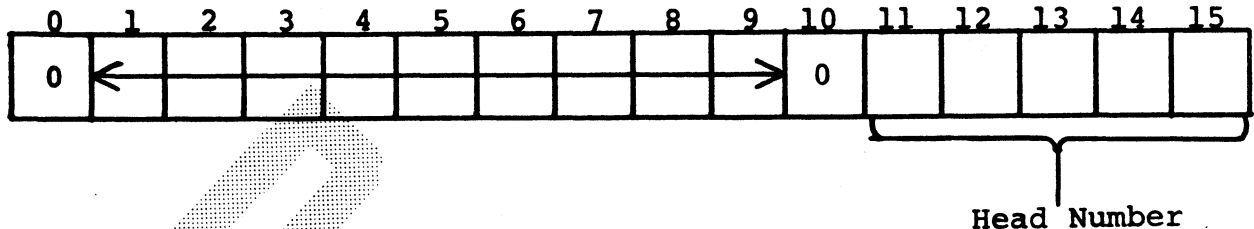


Table 6-12 defines the bit usage in word 3.

**TABLE 6-12. DISC IOCB WORD 3 DEFINITION**

Bits	Definition
0-10	Not used - Must be set to 0
11-15	Used for head/volume number as follows:
<u>11*12 13 14 15</u>	<u>CMD Drives</u>
0 0 0 0 0	Head number 0; Removable
1 0 0 0 0	Head number 1; Fixed
1 0 0 0 1	Head number 2; Fixed
1 0 0 1 0	Head number 3; Fixed
1 0 0 1 1	Head number 4; Fixed
1 0 1 0 0	Head number 5; Fixed
<u>11 12 13 14 15</u>	<u>SMD Drives</u>
0 0 0 0 0	Head number 0
0 0 0 0 1	Head number 1
0 0 0 1 0	Head number 2
0 0 0 1 1	Head number 3
0 0 1 0 0	Head number 4
0 0 1 0 1	Head number 5

\*Bit 11 is the volume bit. It is used by CMD drives only.

#### 6.4.5.5 Sector Select (Word 4)

The sector word is used to specify the starting for the next operation. The controller compares the sector select word with the current sector address in the sector header. If the sector address does not match, the sector will not be accessed by the controller. The following is the format for the sector select word:

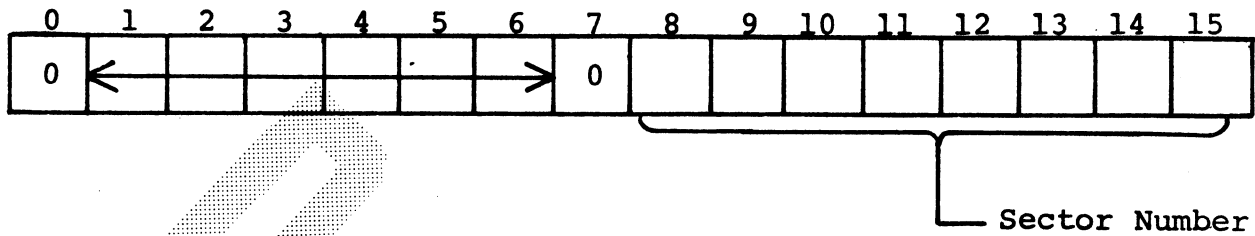


Table 6-13 defines the bit usage in word 4.

**TABLE 6-13. DISC IOCB WORD 4 DEFINITION**

Bits	Definition
0-7	Not defined - Must be set to 0
8-15	Specifies the sector address



#### 6.4.5.6 Sector Count (Word 5)

The sector count word is only used for Read Regardless and Format operations. The format for IOCB word 5 is shown below:

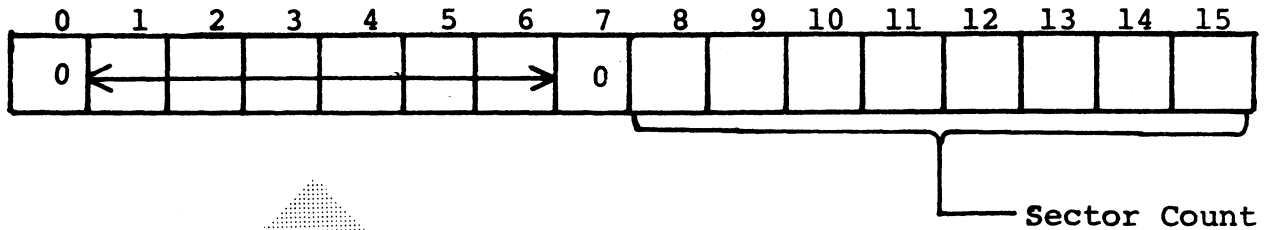


Table 6-14 defines the bit usage in word 5.

**TABLE 6-14. DISC IOCB WORD 5 DEFINITION**

**Bits Definition**

0-7 Not used - Must be set to 0

8-15 Specifies the binary equivalent of the number of sectors to be performed in the operation.

OR

Specifies that the maximum operation that can be performed for either read Regardless or Format commands is one track. In other words, the sector count should never be greater than the sector capacity of the track.

### 6.4.5.7 Memory Address (Word 6)

The Memory address word is used to specify the starting address for the data buffer. The data buffer consists of a block of data either to be taken from the Memory buffer and written on the disc (Write Data) or read from the disc and placed into the buffer (Read Data). Addressability is up to 32K words. The format for the IOCB word 6 is shown below:

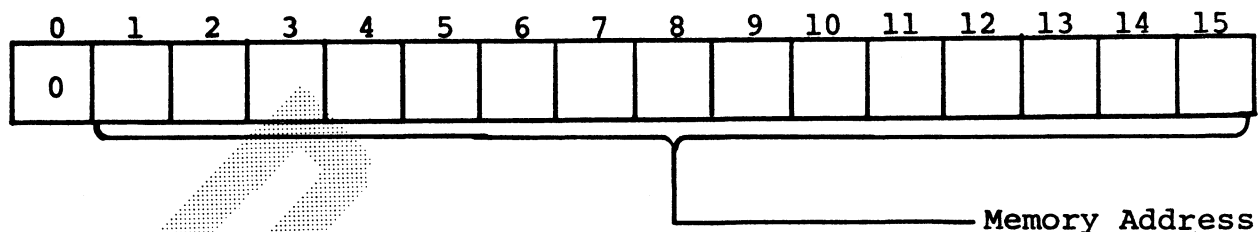


Table 6-15 defines the bit usage in word 6.

**TABLE 6-15. DISC IOCB WORD 6 DEFINITION**

Bits	Definition
0	Not used - Must be set to 0 since there are only 32K words of memory in the POINT 4 MARK III
1-15	Starting address for the memory buffer.

### 6.4.5.8 Termination Status (Word 7)

Upon completion of an operation, the controller will write a termination status into the last word of the IOCB. The termination status provides information on the result of the operation, controller status, and drive status to the processor. One word must be reserved at the end of the disc IOCB for the Termination Status Word.

The controller termination status word reflects the general status of the controller and disc drive at the time of termination of the operation. Disc related status information refers only to the drive selected by the Unit Select Word in IOCB involved. The following shows the format of the IOCB word 7:

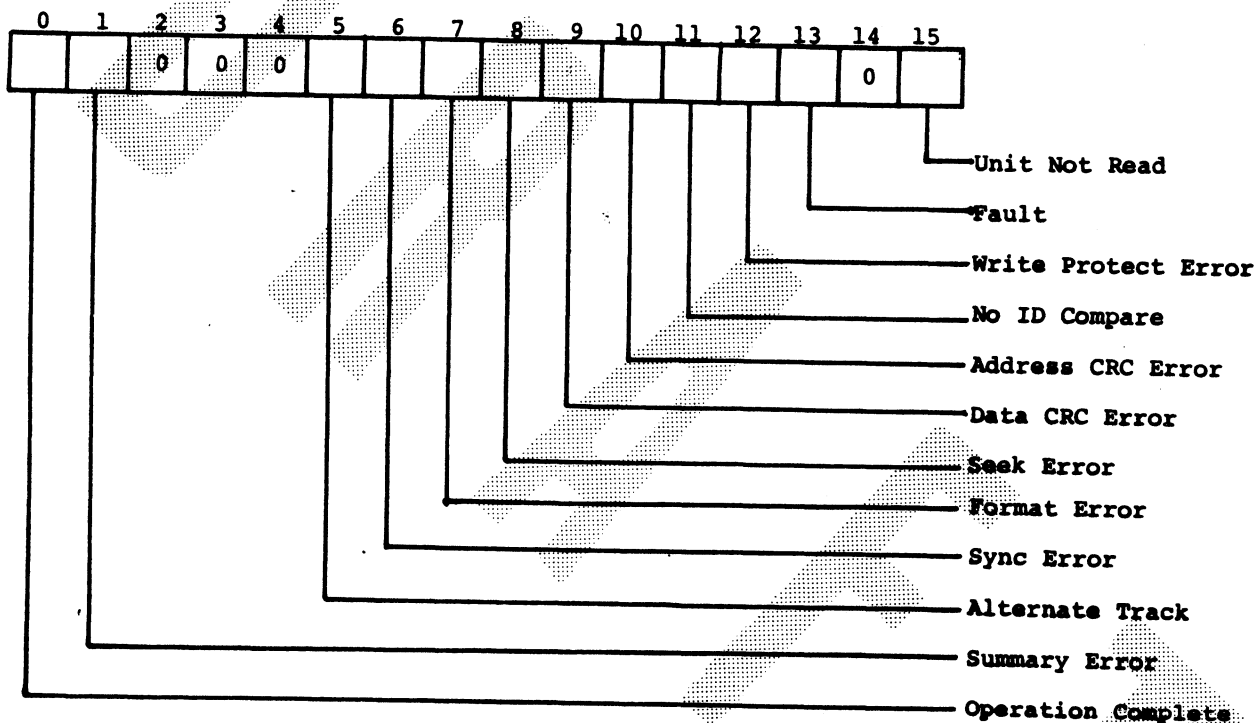


Table 6-16 defines the bit usage in word 7.

**TABLE 6-16. DISC TERMINATION STATUS (WORD 7) DEFINITION**

Bit	Definition
0	Operation Complete - Set to 1 upon completion of an operation, successful or not. It provides a means for software to determine when a valid Termination Status Word has been written. The operation complete bit must be cleared by software. For an error-free operation, this will be the only bit set in the Termination Status Word.
1	Summary Error - Indicates that one or more of the controller error bits are set. Operation complete does not set this bit.
2-4	Not Used - Must be set to 0.
5	Alternate Track - Set to 1 when the controller detects a Non-Zero value in the next cylinder part of the sector address header, during a sector verification.  The controller overwrites the cylinder and head number words in the IOCB with the value found in the sector header next cylinder and next head fields. The controller will also set the alternate track bit to 1, the operation complete bit to 1, and the summary error bit to 1. The pending bit will be set to 1 and an interrupt generated, if interrupts are enabled.  When software detects this bit set, a new seek should be issued based on the new information in the IOCB.
6	Sync. Error - Indicates that the controller failed to synchronize with data (could not find any sectors on the track).
7	Format Error - Indicates that during a format, either a track overrun occurred or a CRC error was detected; or that a data miscompare had occurred while verifying the track.
8	Seek Error - Indicates that a seek error occurred on the selected drive.
9	Data CRC Error - Indicates that a CRC error occurred while reading the data field.

**TABLE 6-16. DISC TERMINATION STATUS (WORD 7) DEFINITION (Cont)**

Bit	Definition
10	Address CRC Error - Indicates that a CRC error occurred while reading the sector address header that the controller was trying to access.
11	No IO Compare - Indicates that the controller could not verify the sector address it was attempting to access. In effect this means that there was no sector address match with any of the sectors on that track.
12	Write Protect Error - Indicates that a write or format operation was attempted on a drive that was write protected.
13	Fault - Indicates that a fault has occurred in the selected drive. A drive fault means that one or more of the following has occurred: <ul style="list-style-type: none"><li>o Power fail</li><li>o Illegal head select</li><li>o Write fault</li><li>o Writing or reading while off cylinder</li><li>o Write and read gate are on simultaneously</li></ul>
14	Not used - Must be set to 0
15	Unit Not Ready - Indicates that the unit selected in the IOCB WORD 1 is not installed, is not up to speed, the heads are not engaged or a drive fault exists.

#### 6.4.6 INPUT/OUTPUT INSTRUCTIONS

The software plays a key role in controlling certain initialization functions of the controller such as seek, fault clear and return to zero. The DIA and DOA instructions are used to perform these functions. The method of controlling data transfer between the processor and the disc using DIA and DOA instructions is called Programmed I/O.

The DOBC and DIBC instructions form a controller command instruction group. These instructions are used for activating/deactivating the controller and for sensing the interrupt pending bit and idle condition.

U  
P  
R  
T  
T

### 6.4.6.1 Programmed I/O Input

The DIA instruction is used to test the status of the drive. Software can use the status to detect the following:

- o Unit Ready
- o Drive Fault
- o Write Protect
- o On Cylinder
- o Drive Select
- o Seek Error

The instruction format is:

DIA (a),50 octal

where (a) is the processor accumulator and 50 is the device code. The contents of the accumulator should be arranged as follows:

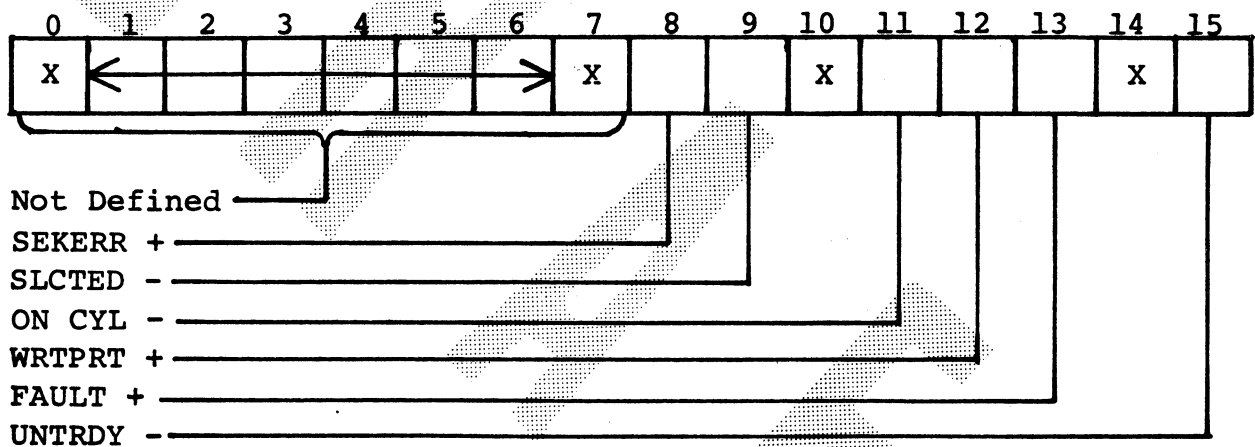


Table 6-17 gives a definition of the function of each bit in the accumulator.

**TABLE 6-17. DIA INSTRUCTION ACCUMULATOR BITS**

Bits	Function
0-7	Not defined
8	<p>Seek Error+: When set to 1, this bit indicates that a seek error has occurred. A seek error can occur from the following:</p> <ul style="list-style-type: none"><li>o A seek could not be completed within 500 milliseconds</li><li>o An illegal track address was received by the drive</li><li>o The carriage has moved outside the recording field</li></ul> <p>A seek error can only be cleared by issuing a Return to Zero command.</p>
9	<p>Selected-: When set to 0, this bit indicates that the drive has been selected. When set to 1, this bit indicates that the drive has not been selected.</p>
10	Not defined
11	<p>On Cylinder-: When set to 0, this bit indicates that the heads are on cylinder. When set to 1, this bit indicates that the heads are not on cylinder.</p>
12	<p>Write Protect+: When set to 1, this bit indicates that the drive that has been selected may only be read from, <u>not</u> written to (i.e., the drive is write protected).</p>
13	<p>Fault+: When set to 1, this bit indicates that a fault in the selected drive has occurred. A fault may be caused by one or more of the following:</p> <ul style="list-style-type: none"><li>o Power fail</li><li>o Illegal head select</li><li>o Write fault</li><li>o Writing or reading while off cylinder</li><li>o Write and read on simultaneously</li></ul>
14	Not defined
15	<p>Unit Not Ready-: When this bit is set to 0, the unit selected is ready. When this bit is set to 1, the unit selected is not installed, is not up to speed, has heads which are not engaged, or a drive fault exists.</p>



### 6.4.6.2 Programmed I/O Output

The DOA instruction with a device code of 50 is used to output the unit number of the drive to be selected. The contents of the accumulator should contain the unit number in word 1 of the IOCB. The format of the instruction is:

DOA (a),50 octal

where (a) is the processor accumulator and 50 is the device code.

The DOA instruction with a device code of 52 is used to output the different TAG pulses. TAG pulses are sent to the drive to indicate that there is valid data in either the bus register, or unit register or both. Each TAG has a different meaning. Their meanings are described in Table 6-18, on the DOA instruction accumulator bits. The contents of the accumulator should be arranged as follows:

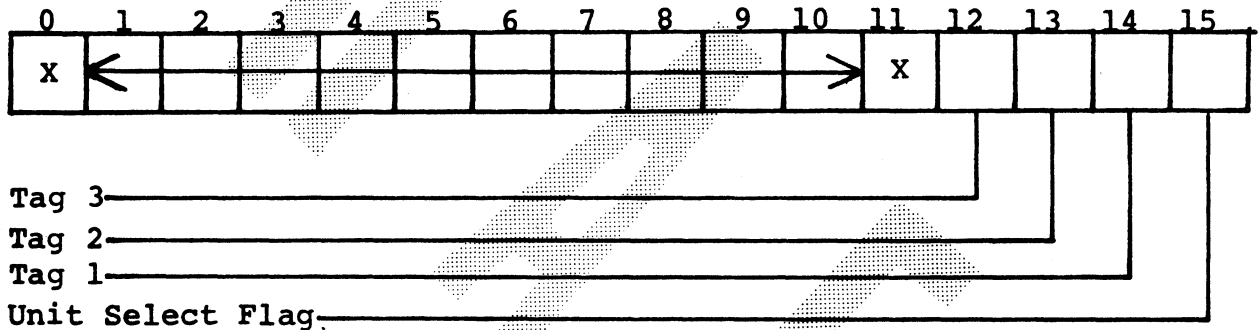


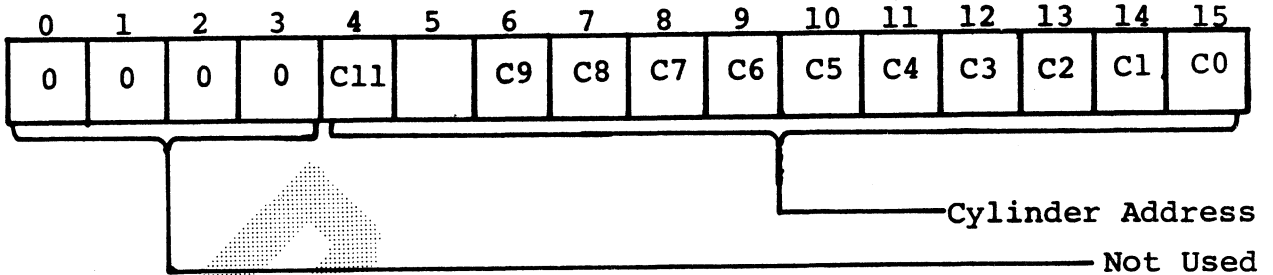
Table 6-18 gives a definition of the function of each bit in the accumulator.

**TABLE 6-18. DOA INSTRUCTION ACCUMULATOR BITS**

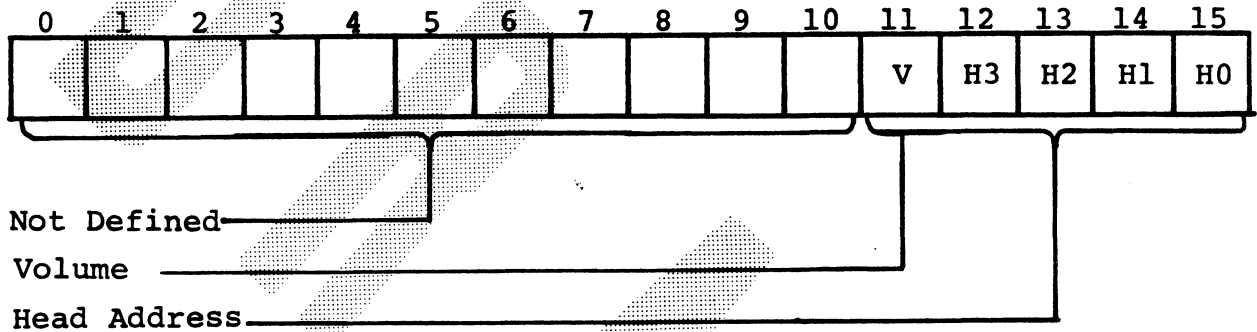
Bit	Function
0-11	Not defined
12	TAG 3: When set to 1, indicates to the drive that the Bus register contains valid drive control information.
13	TAG 2: When set to 1, indicates to the drive that the Bus register contains a valid Volume/Head address.
14	TAG 1: When set to 1, indicates to the drive that the Bus register contains a valid cylinder address.
15	Unit Select TAG: When set to 1, indicates to the drive that there is valid data in the Unit Select Register. Used in conjunction with the Unit Select Register to select the drive, the TAG signal must be held high through the entire operation.

The DOA instruction with a device code of 51 octal is used to load the Bus Register with TAG 1, 2 and 3 information. The Bus Register format is defined as illustrated below for each TAG number:

**TAG 1**



**TAG 2**



**TAG 3**

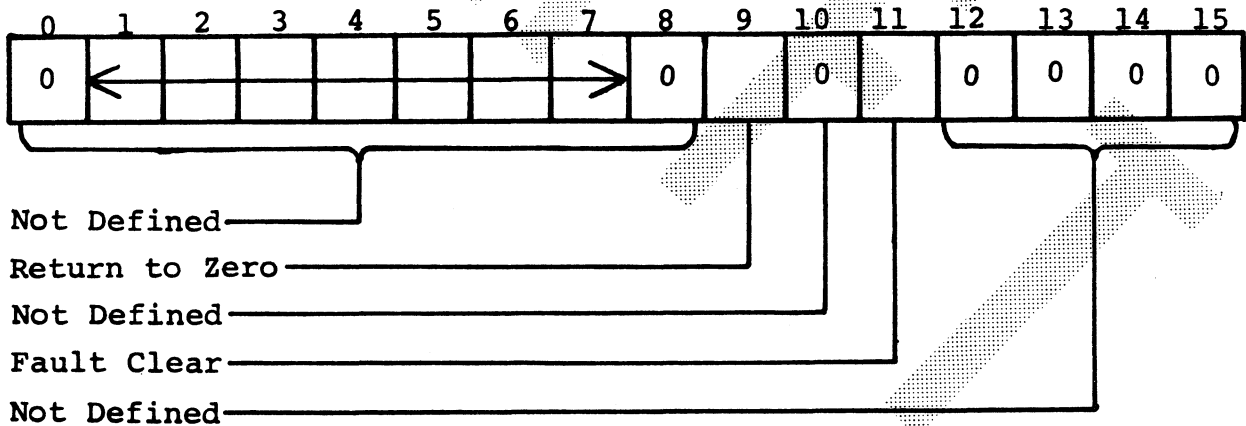


Table 6-19 gives definitions of the TAG 3 bit functions.

**TABLE 6-19. TAG 3 BIT FUNCTIONS**

Bit	Function
0-8	Not defined
9	Return to Zero: When this bit is set to 1 and a TAG 3 is issued, the track will seek to track 0, reset the head register, select the cartridge volume (CMD only) and reset the Seek Error bit in the Termination Status word of the IOCB.
10	Not defined
11	Fault Clear: If a drive fault no longer exists, the drive fault may be cleared by setting this bit and then issuing a TAG 3.
12-15	Not defined

### 6.4.6.3 Controller Command Input

The processor has a dedicated register that is reserved as a pointer to the IOCB for the disc. This register may be written into or read from by using the DOBC or DIBC instructions respectively. The C control code is used to specify command instructions directed at the disc controller, rather than the Multiplexer (S) or tape controller (P). The format of input command instruction is:

DIBC (a),77 octal

where (a) is the accumulator into which the value in the disc IOCB pointer is loaded, and 77 octal is a device code for the CPU.

The DIBC instruction may be used for two purposes.

1. Idle state sensing - The disc controller must be sensed for the idle state before a DOBC instruction to activate the controller and start an operation. This prevents starting a new operation while another is still in progress.

The idle state is defined as a zero value in the accumulator after executing a DIBC instruction to read the disc IOCB pointer. For example:

DIBC (a),77 octal

where (a) = 0 after instruction execution.

2. Polling the Interrupt Pending Bit - When the disc controller has completed an operation, successful or not, it sets the interrupt pending bit in the disc IOCB pointer. The interrupt pending bit is the Most significant bit in the disc IOCB pointer.

The DIBC instruction can be used to read the disc IOCB pointer to check for the interrupt sending bit set to 1. This is used for polling for interrupts.

#### 6.4.6.4 Controller Command Input

The DOBC instruction is used for activating and deactivating the disc controller. The format of the controller command output instruction is:

DOBC (a),77 octal

where (a) is a general purpose accumulator and 77 octal is the device code for the CPU. This instruction is used for two purposes:

1. **Activating the disc controller** - The disc controller is activated when a DOBC instruction is issued with the contents of the accumulator containing the Memory address of the first IOCB word and the Most significant bit set to 0. When activated the controller will go to a busy state and start the operation specified in the IOCB.
2. **Deactivating The Controller** - When the disc controller completes an operation and generates an interrupt, it must be deactivated by the software. The DOBC instruction with the accumulator set to zero is used to deactivate the controller. This resets the pending interrupt and places the disc controller in an idle state.

#### **6.4.7 WRITE DATA OPERATIONS**

A write data command causes the controller to transfer data from the processor memory buffer to the drive. When a DOBC instruction is issued the controller will respond by reading the sector address headers in search of a match to the address in word 2, 3 and 4 of the IOCB. When the controller finds a sector address header that compares with the IOCB address fields, it will start reading the data from the memory data buffer and writing it to the disc at the sector located. This procedure continues until 256 data words are written to the sector.

#### **6.4.8 READ DATA OPERATION**

The Read Data operation is used to read data from the disc and transfer it into the specified processor memory buffer. When a DOBC instruction is issued, the controller responds by searching for the sector address specified in words 2, 3 and 4 of the IOCB. When the controller finds a sector address header that compares with the IOCB address fields, it starts reading data from the disc and writing it to the memory buffer. After the sector is completely read (256 words), the CRC is checked. If the CRC is not correct, the data CRC error bit will be set to 1 in the termination status word.

#### **6.4.9 READ VERIFY OPERATION**

The Read Verify operation is used to verify that data written on the disc during a Write Data operation was written without error. When a DOBC instruction is issued the controller responds by searching for the sector address specific in the IOCB. When the controller finds a sector address header that compares with the IOCB address fields, it reads the data in the sector without transferring it to memory. The data is checked for accuracy and any errors reported in a termination status word.

#### **6.4.10 READ REGARDLESS OPERATION**

The read regardless operation is used by the software for error recovery and the identification of bad tracks. When the disc controller is activated by a DOBC instruction, it will start transfer of data from the disc to the memory buffer upon detection of an Index Mark. The number of sectors transferred is specified in the IOCB sector count. The maximum sector count should not be greater than the sector capacity of one track. The largest read cycle that a Read Regardless command can perform is one track. There is no sector verification involved and the controller will transfer all 262 words of the sector including the address header (1 word), the address CRC (1 word), the data field (256 words) and the data CRC (1 word).



#### 6.4.11 FORMATTING OPERATION

The disc is formatted by software under software control when a DOBC with a format OPCODE, the controller searches for the first index mark and begin reading address headers from processor memory. Each sector is written onto the disc, allowing partitioning in the manner best suited for the particular application.

Before initiating a format operation the software must build a sector table in memory. This table is used as a source of sector header information when formatting begins. The starting address for this table is located in word 6 of the disc IOCB. The sector table consists of a series of sector blocks, one for each sector to be formatted. All sector blocks must be stored in a contiguous block in processor memory, in the same sequential order that they are to be written to disc. The sector addresses, however, do not have to be in sequential order. Figure 6-17 shows the sector table as built in memory.

TO BE SUPPLIED

Figure 6-17. Sector Table in Memory

### 6.4.11.1 Memory Sector Block Description

The sector table blocks are a four word block containing all of the information that is to be written into the sector address header for each sector. Figure 6-18 shows the content of block in the sector table in memory.

Table 6-20 defines the field in the memory sector blocks.

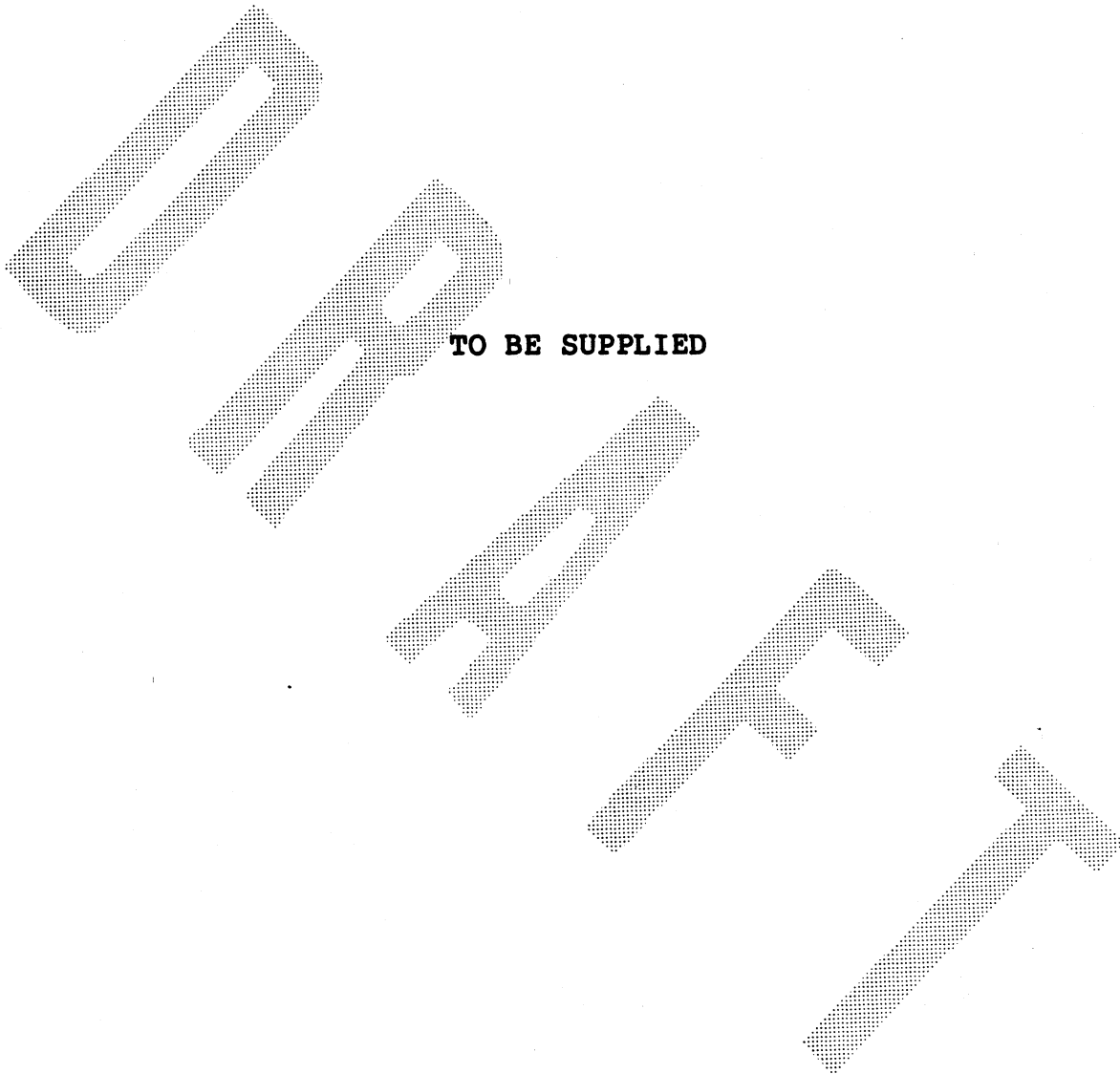
Figure 6-19 is an illustration of the sector format.



**Figure 6-18. Content of Each Block in The Sector Block**

**TABLE 6-20. MEMORY SECTOR BLOCK FIELDS**

Word	Bits	Function
0	0-3	May be used to define file protect flags.
0	4-15	Define the current cylinder address of the sector address header.
1	0-7	Define the current sector address of the sector address header.
1	8-10	Not used - Must be set to 0
1	11-15	Define the current head address of the sector address header.
2	0-3	May be used as file protect flags.
2	4-15	Define the next cylinder address of the sector address header. When initially formatting a track this word must be set to zero. If the track being formatted was bad, the software may update this word with an alternate cylinder address. A non-zero value in this word notifies the controller that this track is bad, sets the alternate track bit to 1 in the termination status word and terminates the operation. The next cylinder word of each sector block in the sector table (every sector on the track) should contain the same value, since the entire track will be flagged bad.
3	0-7	Not used with controller - Must be set to 0
3	8-10	Not used with controller - Must be set to 0
3	11-15	Not used with controller - Must be set to 0



**Figure 6-19. Sector Format**

### 6.4.11.2 Formatting Procedure

The format command will format one track per operation. The software must insure that the sector table contains all sectors to be written on a track. If there are fewer sectors specified than are available on the track, the remaining sectors will be zeroed out.

The software must also build the IOCB for the format operation. The IOCB cylinder and head select words must match the current cylinder address and current head address of the first sector block in the sector table. The IOCB memory word (word 6) must contain the address of the first word in the sector table. The IOCB sector count (word 5) must be set to the total number of sectors to be formatted on the track (i.e., sector blocks in the sector table). Figure 6-20 is an example of a format IOCB and sector block.

When activated by a DOBC instruction, the controller waits for a signal that an Index Mark has been located on the drive. When this signal is received the controller will start writing on the sector with the address header specified by the sector block. Next the data field will have a worst case pattern written into it.

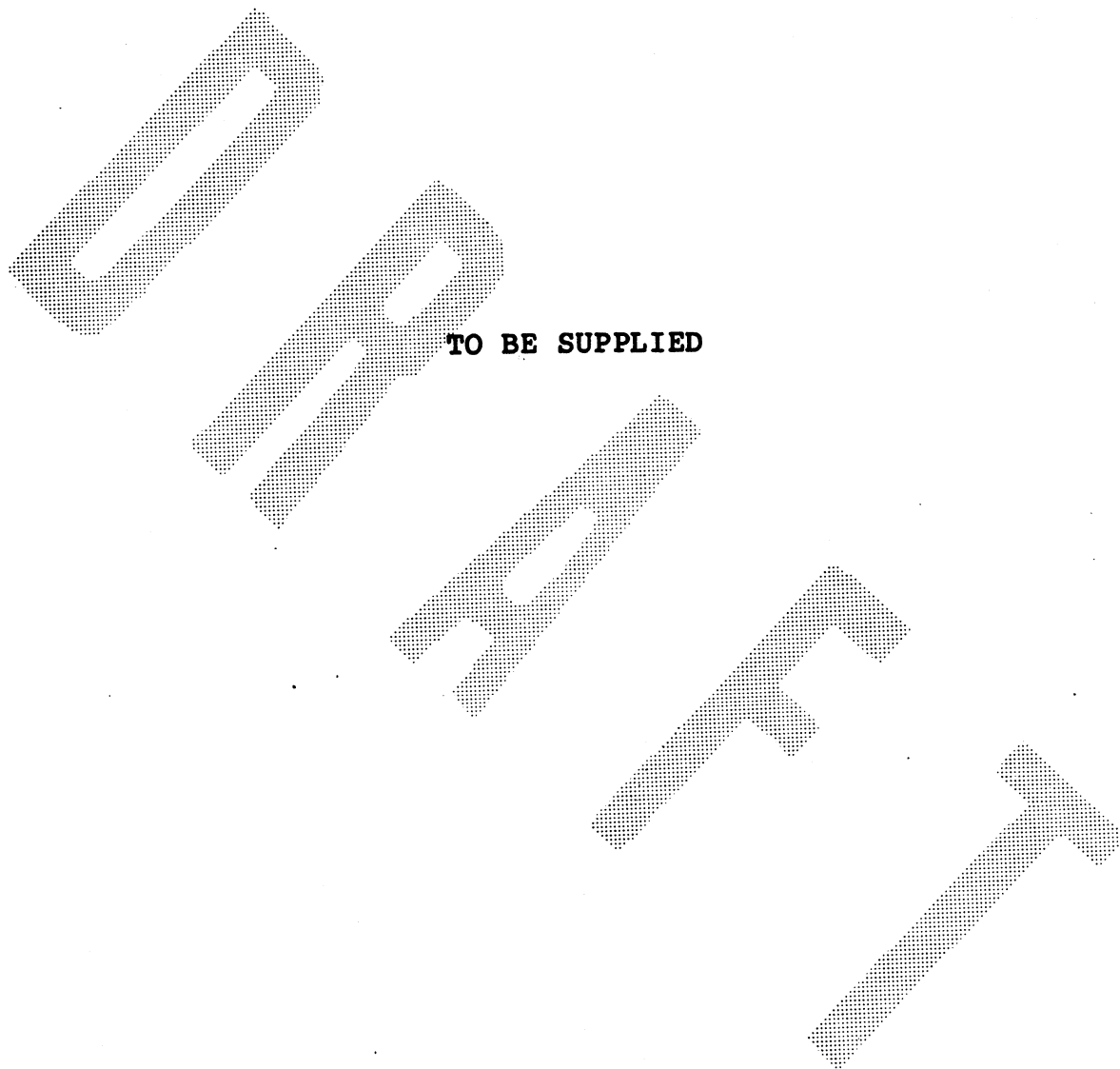
The controller will continue to write sector address header, data field, CRC characters and inter-sector gaps until one of two events occur:

1. All available capacity on the current track is used. In this case the format operation will abort and a format error will be reported in the termination status word.
2. The entire track has been formatted. The controller recognizes that the track has been formatted when the sector count in the IOCB goes to zero.

When this happens the controller will fill the remainder of the track with zeroes, set the operation complete bit in the termination status word, set the interrupt pending bit and generate an interrupt, if interrupts are enabled.

When the format operation is complete, the software should issue a read verify for each sector. This will insure that there are no CRC errors in the address fields or data fields.

It is also advisable for software to perform a Read Regardless operation. This will read the address header field, address CRC, the data field and data CRC into main memory. The sector count in the Read Regardless operation should equal the number of sectors written to the track. After the Read Regardless is performed, software may compare the address header information to the information in the sector table. This insures that the address information was written to disc properly.



TO BE SUPPLIED

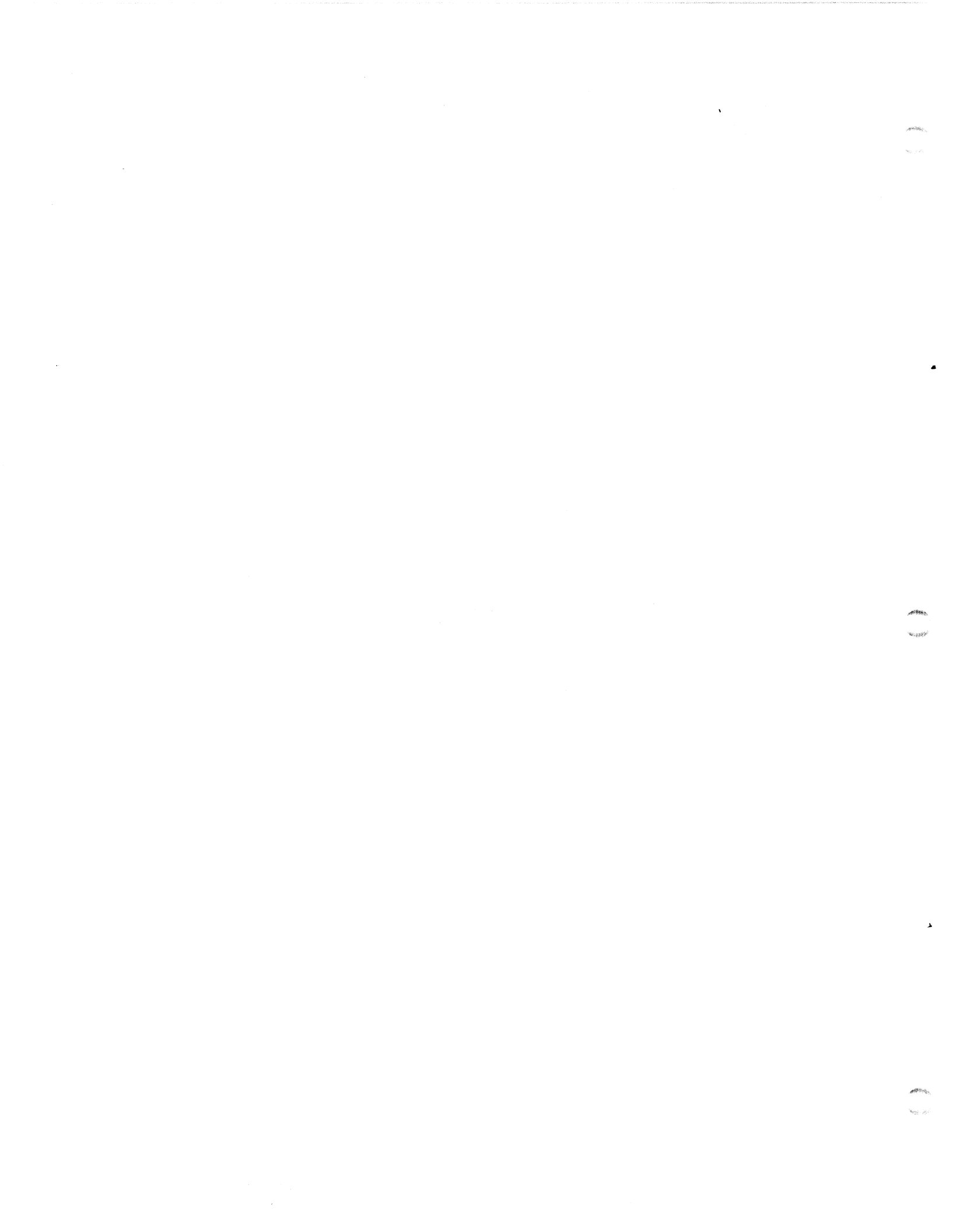
**Figure 6-20. Format IOCB Example**

**6.4.11.3 Format Considerations**

TO BE SUPPLIED

**6.4.12 PROGRAMMING CONSIDERATIONS**

TO BE SUPPLIED





**COMMENT SHEET**

MANUAL TITLE POINT 4 MARK III Computer User Manual

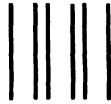
PUBLICATION NO. HM-080-0019 REVISION 01

FROM: NAME/COMPANY: \_\_\_\_\_

BUSINESS ADDRESS: \_\_\_\_\_

CITY/STATE/ZIP: \_\_\_\_\_

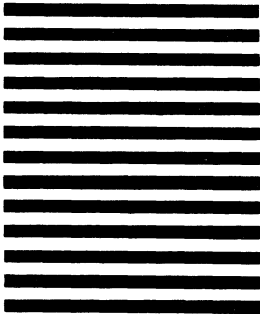
COMMENTS: Your evaluation of this manual will be appreciated by POINT 4 Data Corporation. Notation of any errors, suggested additions or deletions, or general comments may be made below. Please include page number references where appropriate.



**BUSINESS REPLY MAIL**  
FIRST CLASS      PERMIT NO. 5755      SANTA ANA, CA.

POSTAGE WILL BE PAID BY ADDRESSEE:

NO POSTAGE  
NECESSARY  
IF MAILED IN  
UNITED STATES



**POINT 4 Data Corporation**  
**PUBLICATIONS DEPARTMENT**  
2569 McCabe Way  
Irvine, CA 92714

CUT ON THIS LINE



**POINT 4 DATA CORPORATION**

2569 McCabe Way / Irvine, California 92714 / (714) 754-4114