# Gould Multi-Function Processor

# Model 8002

# Technical Manual

June 1987

Publication Order Number: 303-006540-000

GOULD
Electronics

## HISTORY

The Gould Multi-Function Processor Technical Manual, Publication Order Number **303-006540-000**, was printed June, 1987.

This manual contains the following pages:

Title page
Copyright page
iii (iv Blank)
v through xiii (xiv Blank)
1-1 through 1-9 (1-10 Blank)
2-1 through 2-21 (2-22 Blank)
3-1 through 3-13 (3-14 Blank)
4-1 through 4-33 (4-34 Blank)
5-1 through 5-8
6-1 through 6-32
IN-1 through IN-4

# TABLE OF CONTENTS

## TABLE OF CONTENTS (continued)

## TABLE OF CONTENTS (continued)

Chapter Page

## TABLE OF CONTENTS (continued)

## LIST OF ILLUSTRATIONS

## LIST OF TABLES

# CHAPTER 1

# GENERAL DESCRIPTION

## 1.1 Introduction

This technical manual contains the information pertaining to the Multi-Function Processor (MFP). The MFP is manufactured by Gould Inc., Computer Systems Division, Fort Lauderdale, Florida.

The information in this manual is presented in the following order:

- Chapter 1 - General Description
- Chapter 2 - Controls, Indicators, and Connectors
- Chapter 3 - Operation
- Chapter 4 - Programming
- Chapter 5 - Real Time Functions
- Chapter 6 - I/O Support and Interface

The assemblies, logic drawings, and schematics are contained in the Multi-Function Processor Drawings Manual, Publication Order Number 304-006540.

**NOTE**

The acronym MFP, as used throughout this manual, is synonymous with the Multi-Function Processor, unless otherwise noted.

## 1.2 Features

- Two Small Computer System Interface (SCSI) Ports
- Eight Asynchronous Communications Ports
- Line Printer Interface
- Turnkey Panel Interface (optional)
- System Timer (Real-Time Clock/Interval Timer)
- External Interrupt Input and/or Output
- General Purpose Expansion Interface

## 1.3 Physical Description

The Multi-Function Processor consists of the MFP circuit card, the MFP device interface circuit card, and an optional general purpose device interface circuit card. See figure 1-1.

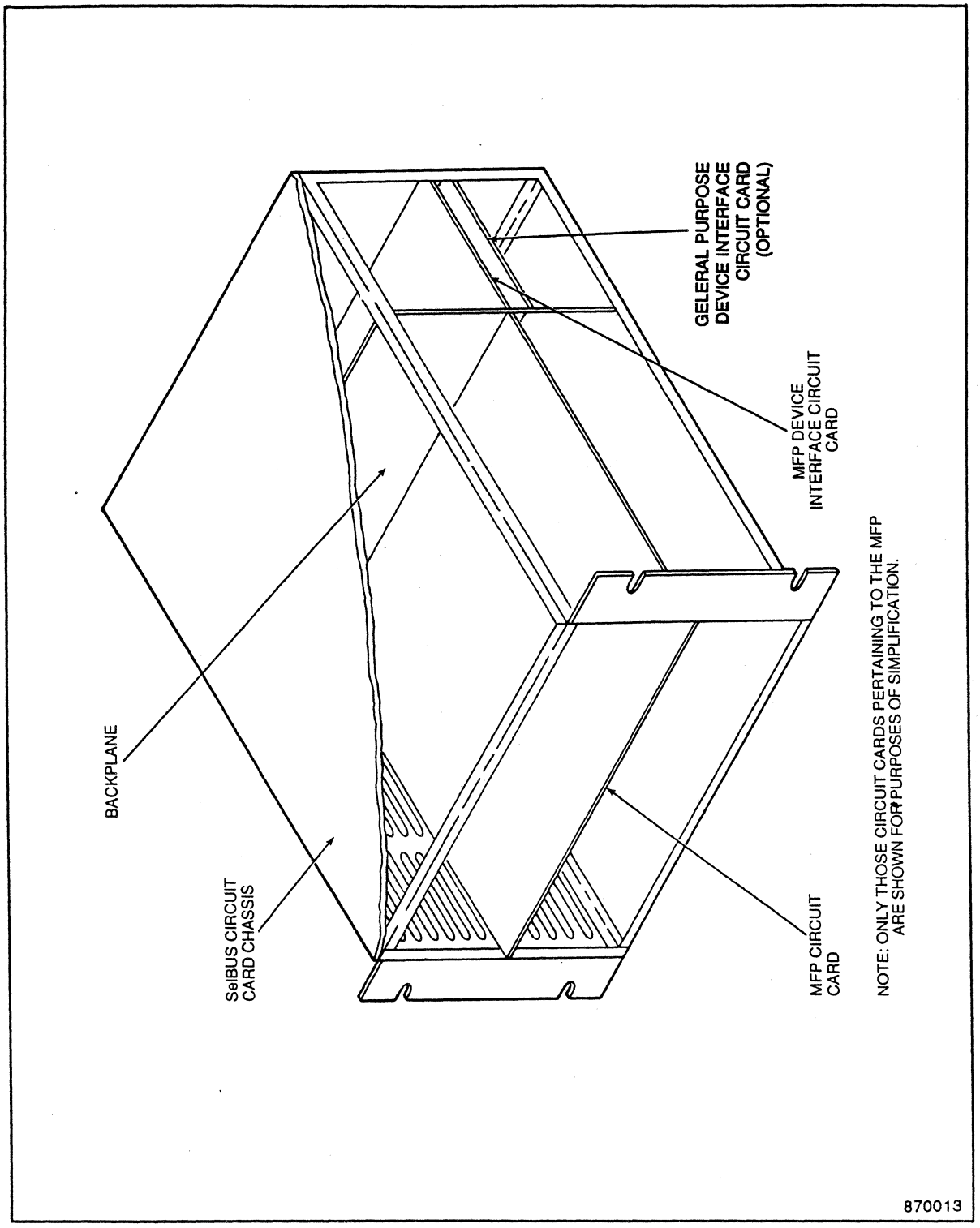



**Figure 1-1. MFP Installed in Typical Card Cage**

### 1.3.1 MFP Circuit Card

The MFP is contained on a single 15 inch wide by 18 inch deep plug-in type circuit card. See figure 1-2.

At one edge of the MFP circuit card are a row of gold fingers which run the full width of the circuit card and provide the electrical interface with the backplane. At the other edge of the MFP circuit card is an on line/off line switch, four light emitting diode (LED) displays, and three connectors. Also located on the MFP circuit card are jumper locations X1 through X9 and connector J4.

The gold fingers are segmented into three groups: the center group, consisting of 184 gold fingers, provides the interface with the SelBUS; the two outer groups, each consisting of 50 gold fingers, provides the interface with the MFP device interface circuit card. The interface with the MFP device interface circuit card is provided through a passive section of the backplane.

The on line/off line switch provides a means to logically disconnect the MFP from the SelBUS without physically removing it. Refer to Chapter 2, Controls and Indicators, for a complete description of this switch.

The four LED displays provide a visual indication as to the current state of the MFP. Refer to Chapter 2, Controls and Indicators, for a complete description of these displays.

The four connectors (J1, J2, J3, and J4) provide a means of connecting test equipment to the MFP. Normally, there are no connections made to these connectors in the operational environment; however, the connectors are assigned as follows:

- J1 - address and control interface
- J2 - ROM-SIM bits 32 through 47
- J3 - ROM-SIM bits 00 through 31
- J4 - Y bus interface

Refer to Chapter 2, Controls and Indicators, for a complete description of these connectors.

The jumpers X1 through X9 are used to set up the MFP circuit card for operation in a specific computer system. The jumpers are assigned as follows:

- X1 through X6 - SelBUS priority poll
- X7 - channel address and turnkey panel enable
- X8 - interval timer clock rate
- X9 - real-time clock control rate

Refer to Chapter 2, Controls and Indicators, for a complete description of these jumpers and details for setting them up.

### 1.3.2 MFP Device Interface Circuit Card

The MFP device interface is contained on a single 15 inch wide by 9.2 inch deep plug-in type circuit card. See figure 1-3.

The MFP device interface circuit card consists of two connectors on the edge of the circuit card that interfaces with the backplane, eight connectors that interface with input/output (I/O) devices external to the MFP, and three jumper locations X1, X3, and X4.

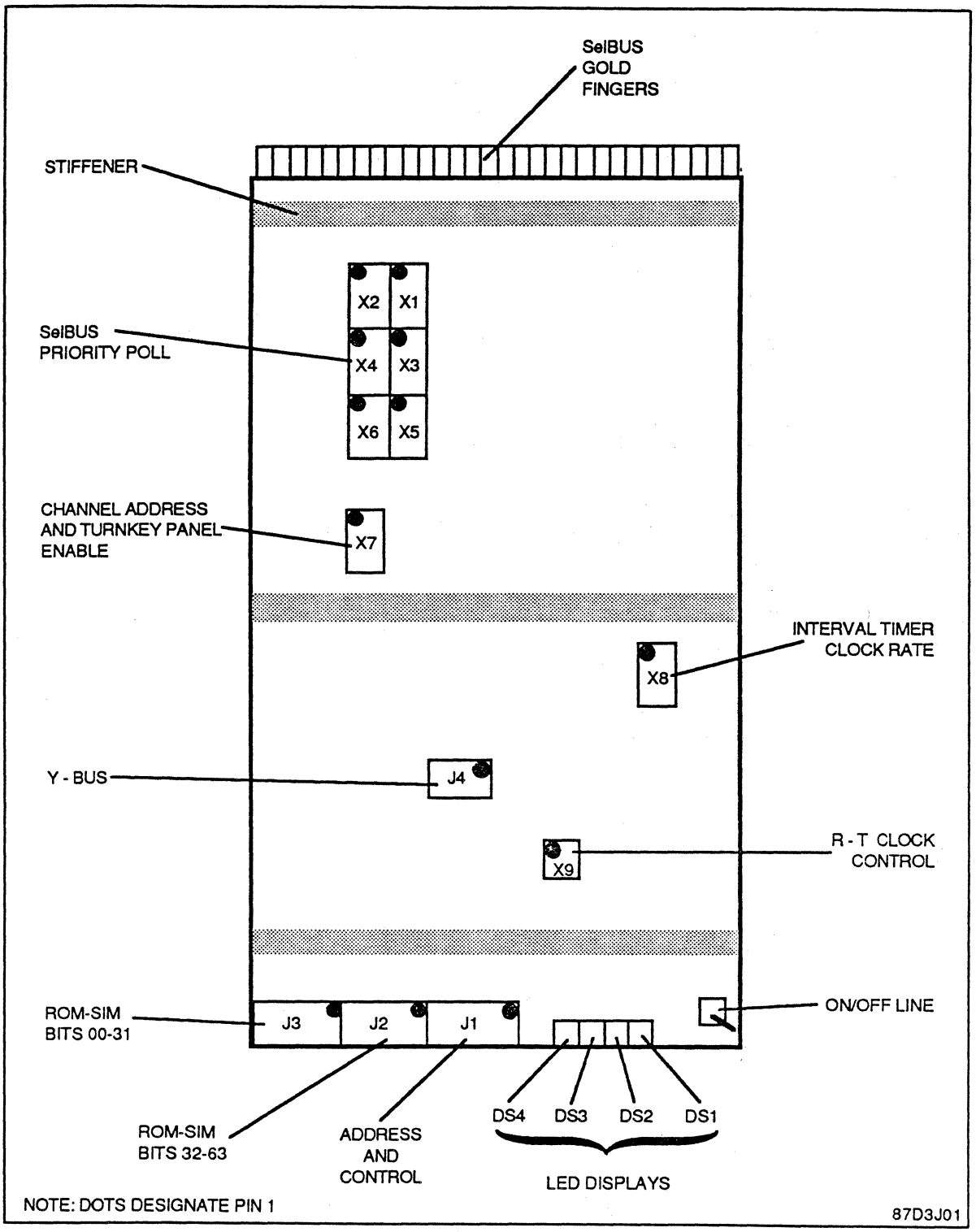



**Figure 1-2. Physical Layout of the MFP Circuit Card**

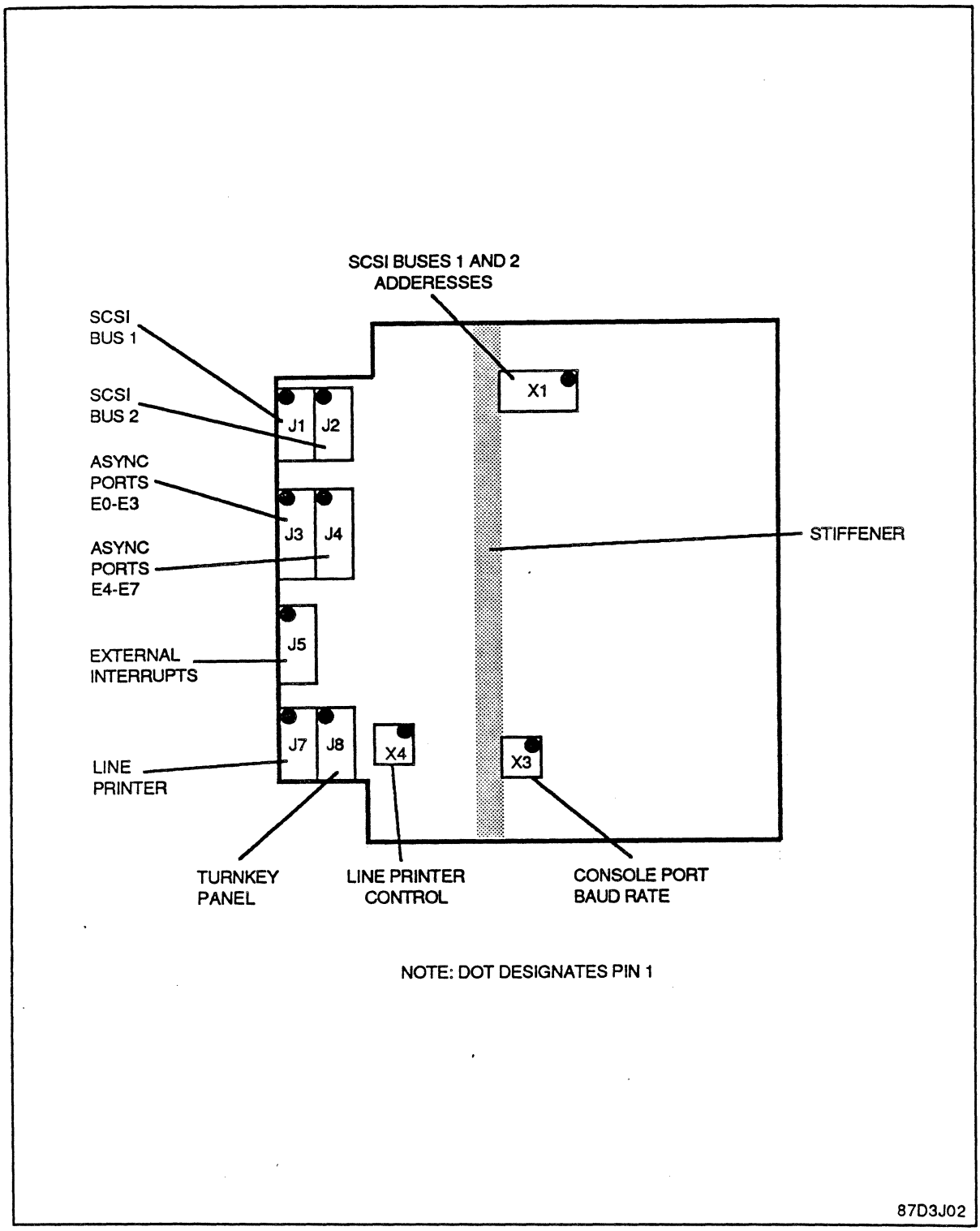



Figure 1-3. Physical Layout of the MFP DI Circuit Card

The two connectors that interface with the backplane are passive to the backplane, they provide the interface for the MFP circuit card; therefore, the MFP circuit card and the MFP device interface circuit card must occupy the same slot position on the SelBUS circuit card chassis. See figure 1-1.

The eight connectors on the outer edge of the circuit card provides the interface with external I/O devices. The connectors are labeled J1 through J8 and interface with the following devices:

- J1 - SCSI bus 1
- J2 - SCSI bus 2
- J3 - asynchronous ports 0 through 3
- J4 - asynchronous ports 4 through 7
- J5 - external interrupts
- J6 - not used
- J7 - line printer
- J8 - turnkey panel

The jumpers X1 through X4 are used to set up the MFP device interface circuit card for operation in a specific computer system configuration. The jumpers are assigned as follows:

- X1 - SCSI bus address
- X2 - not used
- X3 - turnkey panel baud rate
- X4 - line printer control

Refer to Chapter 2, Controls and Indicators, for a complete description of these jumpers and details for setting them up.

### 1.3.3 Secondary General Purpose Device Interface Circuit Card

The secondary general purpose device interface is an optional circuit card. It is contained on a single 15 inch wide by 9.2 inch deep circuit card. It connects to the MFP device interface circuit card via the P1C/P1A jumper cards that are provided when the general purpose device interface circuit card is ordered. Refer to the specific general purpose device interface circuit card technical manual for a complete description.

## 1.4 Functional Description

The MFP is a single slot, Class 'F' protocol, I/O processor, SelBUS interfaced, circuit card. It provides two Small Computer System Interface (SCSI) ports, eight asynchronous communications ports (one port can be utilized by the optional system console CRT), one parallel printer port, the system timer (Real-Time Clock/Interval Timer), and 12 external interrupts. The MFP, when combined with a CPU and memory, produces a basic computer system. See figure 1-4.

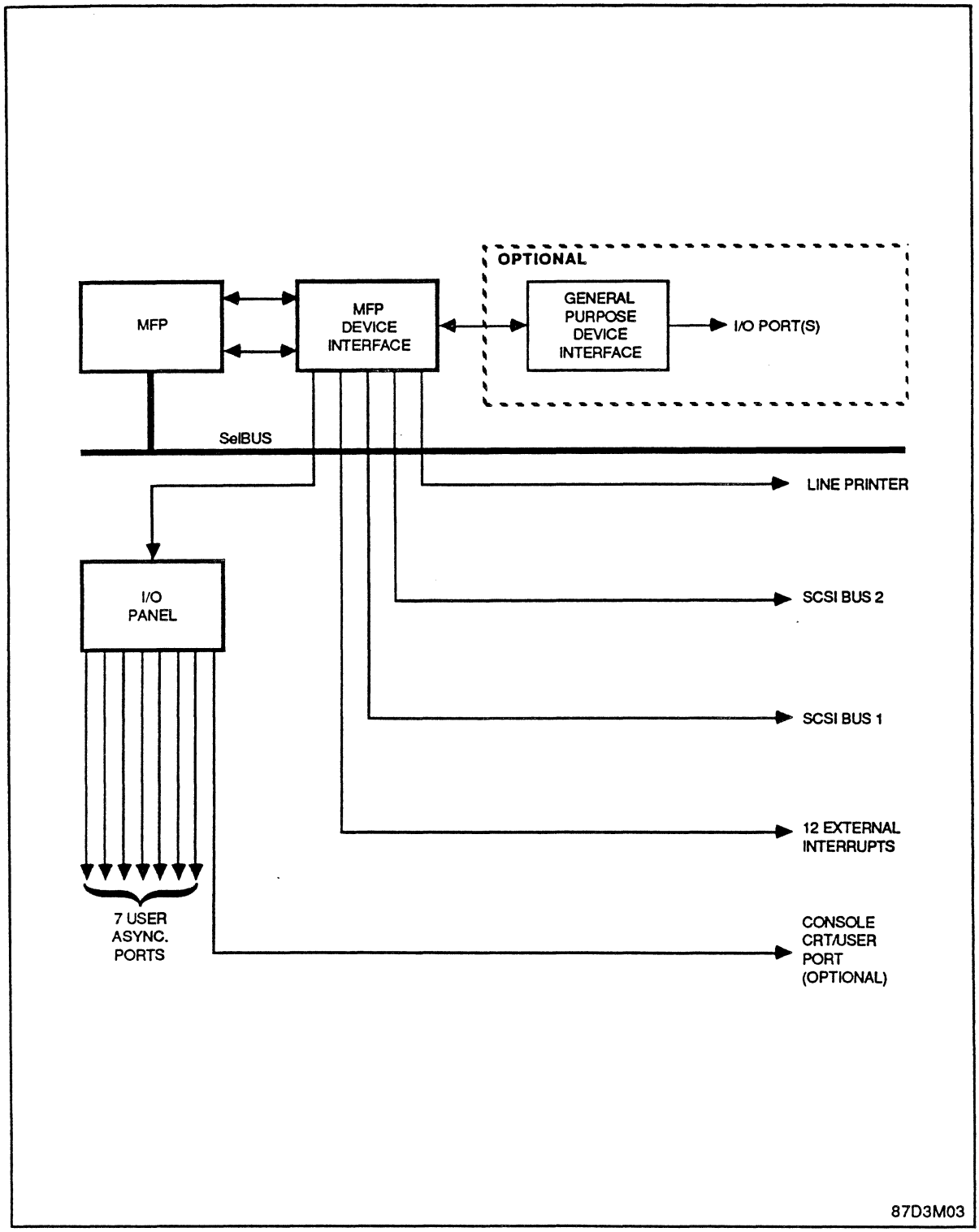



Figure 1-4. Functional Block Diagram with MFP

### 1.4.1 SCSI Ports

The two Small Computer System Interface (SCSI) ports are used to interface with the SCSI Bus 1 and the SCSI Bus 2. Normally, the SCSI Bus 1 is used to interface with I/O devices such as hard disk drives and the SCSI Bus 2 is used to interface with I/O devices such as magnetic tape units. The only difference between the two buses is that SCSI Bus 1 has a higher priority than SCSI Bus 2.

### 1.4.2 Asynchronous Communications Ports

The asynchronous communications port on the MFP device interface circuit card connects with an I/O panel through a cable that is provided with the panel. The I/O panel provides eight asynchronous communications ports, of the eight ports, one can be used for the system Console CRT if the MFP is configured to run the turnkey panel function. The remaining seven I/O ports provide RS-232 communications capabilities for the user.

### 1.4.3 Parallel Printer Port

The parallel printer port is a standard Data Products port, it provides the interface for the line printer.

### 1.4.4 System Timer

The system timer port provides the computer system with the real-time clock and interval timer, as well as the external interrupt capabilities.

## 1.5 Software

The MFP operates with Gould's MPX-32 and UTX/32 operating systems.

## 1.6 Specifications

The specifications for the MFP are listed in Table 1-1.

**Table 1-1. Specifications**

| Characteristics | Specifications |
|---|---|
| Physical | |
| Height | not applicable |
| Width | 15 inches (38.1 centimeters) |
| Depth | 18 inches (45.7 centimeters) |
| Weight | not specified |
| Electrical | |
| Voltage | +5 volts DC<br>+15 volts DC |
| Current | 35 amperes |
| Power Dissipation | not specified |
| Heat Dissipation | not specified |
| Environmental | |
| Temperature | |
| Operating | 50 - 89 degrees F (10 - 32 degrees C) |
| Shipping & Storage | 50 - 122 degrees F (10 - 50 degrees C) |
| Humidity | |
| Operating | 20 - 80%, noncondensing |
| Shipping and Storage | 10 - 90%, noncondensing |
| Altitude | see system and peripheral manuals |
| Performance rates | |
| Data bus word size | 8/16 bits |
| Microinstruction word size | 48 bits |
| Orders | pulse and level |
| External interrupts | 12 RTOM |
| I/O interrupts | 1 class F |
| Protocol | class F; extended I/O.<br>class 3; real-time clock, interval timer, and external interrupts.<br>class 0; turnkey panel function. |
| I/O Ports | |
| SCSI | 2 single ended async. buses |
| Async. Comm. | 8 user ports or<br>7 user and 1 Console CRT port |
| Parallel Printer | 1 |
| System Timer | 1 real-time clock and<br>1 interval timer |
| Expansion | 1 optional; see specific manual |

# CHAPTER 2

# CONTROLS, INDICATORS, and CONNECTORS

## 2.1 Introduction

This chapter contains the information pertaining to the controls, indicators, and connectors of the Multi-Function Processor (MFP). The MFP consists of two circuit cards: the MFP circuit card and the MFP DI circuit card; therefore, the descriptions in this chapter are broken down by circuit cards.

## 2.2 Controls, Indicators, and Connectors, MFP Circuit Card

The MFP circuit card contains many types of components; however, this chapter will be limited to controls (on line/off line switch and jumpers), indicators (light emitting diodes (LED's), and connectors (J connectors). See figure 2-1 for the actual location of the above mentioned components. Each component is described in detail as you progress through this chapter.

### 2.2.1 Controls, MFP Circuit Card

The controls on the MFP circuit card enable the user to configure the MFP for a specific computer system. There are two types of controls on the MFP circuit card: the on line/off line switch, and nine sets of jumpers.

#### 2.2.1.1 On Line/Off Line Switch, MFP Circuit Card

The on line/off line switch provides the means to logically disconnect the MFP from the SelBUS. When the switch is placed in the off line position, all of the SelBUS drivers and receivers, resident to the MFP, are disabled. When the MFP is disabled, its firmware is placed in a "hold" state. With the firmware in a hold state, micro interrupts are serviced; however, initiation of new actions will not take place. When the MFP is off line, the LED indicator, DS4, will not be illuminated, thus indicating that the MFP is in an off line (non-operational) state; however, the other LED's, DS1 through DS3, will remain operational and indicate there status as if on line.

The on line/off line switch of the MFP can be changed while the computer system is powered up; however, it is recommended that the system be halted so that any SelBUS transactions taking place will be completed before the status of the MFP is changed. Note: if the MFP is configured with a turnkey panel, it will not function if the MFP is off line.

#### 2.2.1.2 Jumpers, MFP Circuit Card

The MFP circuit card contains nine sets of jumpers. Each set of jumpers makes up one jumper header. The jumper headers are labeled from X1 through X9 and called out in figure 2-1, as well as, the logic drawings and the actual MFP circuit card.

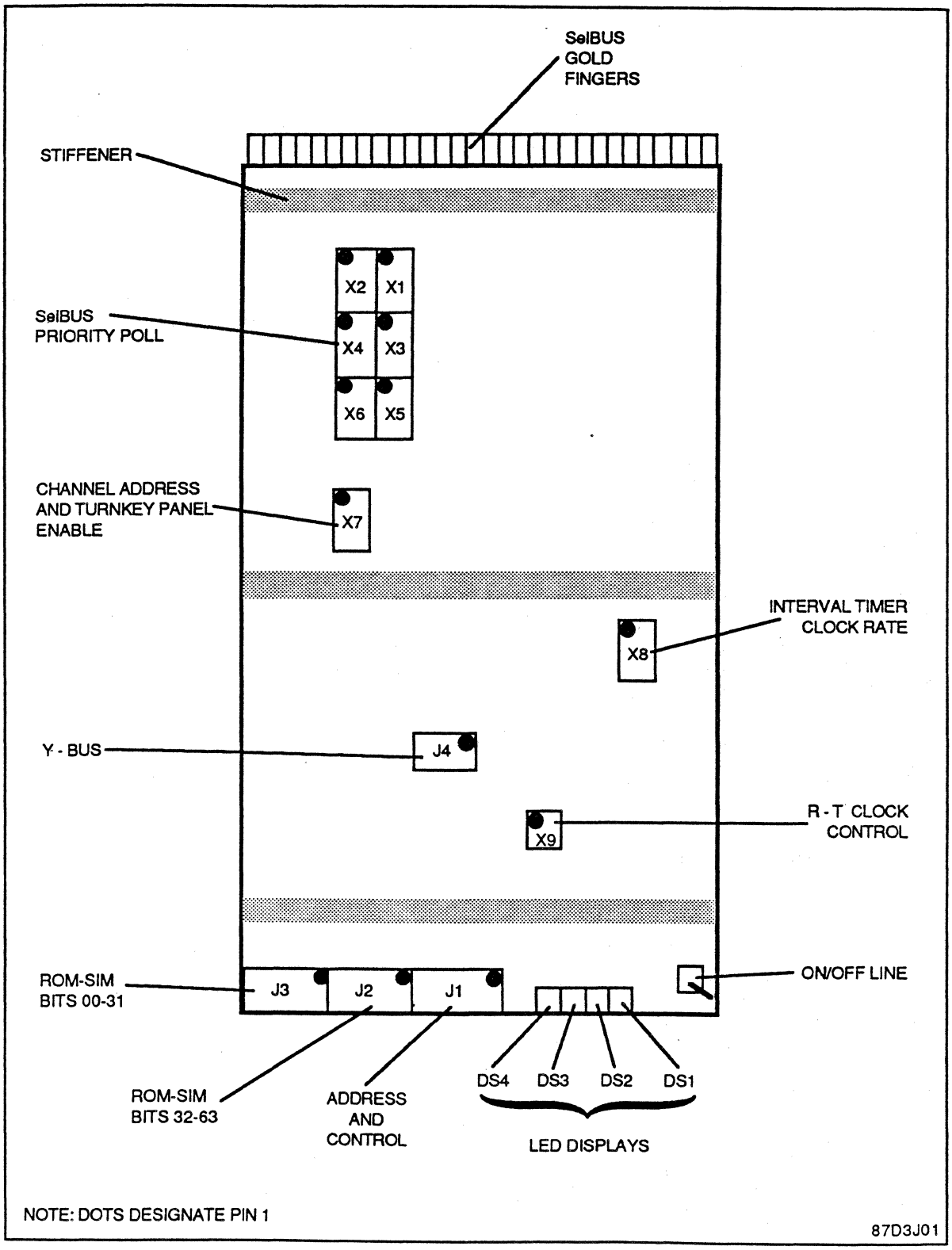



Figure 2-1. MFP Circuit Card Layout

The function of the jumper headers are as follows:

- X1 through X6 - sets up the SelBUS priority.
- X7 - sets up the channel address and turnkey panel enable.
- X8 - sets up the interval timer clock rate.
- X9 - sets up the real-time clock control rate.

To set up the MFP circuit card, jumpers are inserted between opposite pins of the jumper header; for example: on a 16 pin jumper header, a jumper is inserted between pins 1 and 16, pins 2 and 15, and so on, down to pins 8 and 9; similarly, on an 8 pin jumper header, a jumper is inserted between pins 1 and 8, pins 2 and 7, pins 3 and 6, and pins 4 and 5, see figure 2-2.

## SelBUS PRIORITY

The SelBUS priority for the MFP circuit card is controlled by jumper headers X1 through X6. All circuit cards resident on the SelBUS must have a unique priority assigned to them. When installing the MFP circuit card, or any other circuit card, a SelBUS priority must be selected that is not being used by another circuit card. When you are installing the MFP on an existing computer system, the easiest way to select a SelBUS priority is to look at the SelBUS terminator card. Select and remove a priority enable jumper from this card and set up the MFP to correspond with the selected priority. It is recommended that the MFP be assigned to priority 9 when ever possible, this will allow the MFP to operate with an efficient data throughput.

Once you have selected the SelBUS priority and the SelBUS terminator card has been configured appropriately, the MFP circuit card must be configured (jumpered) accordingly.

### SelBUS Priority Recognition

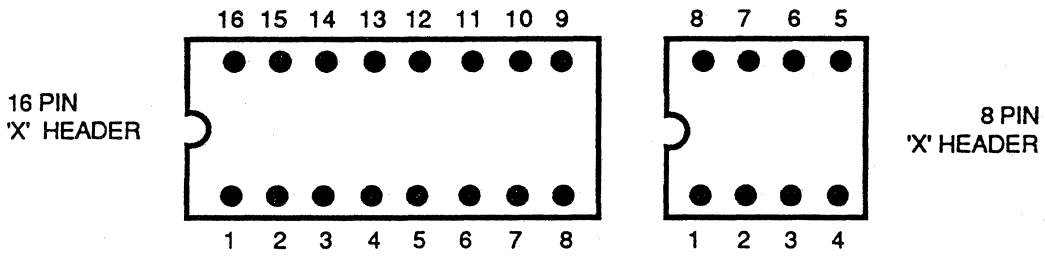
The jumper headers X1, X3, and X5 are used to set up the SelBUS priority recognition. Each jumper position starting from the pin closest to the SelBUS edge of the circuit card and moving away decreases the priority by one, see table 2-1. To set up the SelBUS priority recognition for the MFP, insert a jumper for each priority that is higher than the MFP's priority. For example: if the MFP's SelBUS priority recognition is 9, jumpers will be inserted in X1-1 through X1-8, leaving X3-1 through X5-8 empty. The MFP, like all other circuit cards, does not recognize its own priority.

### SelBUS Priority Generation

The jumper headers X2, X4, and X6 are used to set up the SelBUS priority generation. Each jumper position starting from the pin closest to the SelBUS edge of the circuit card and moving away decreases the priority by one, see table 2-1. To set up the SelBUS priority generation for the MFP, insert a jumper for the priority that has been selected for the MFP's priority. For example: if the MFP's SelBUS priority generation is 9, a jumper will be inserted in X4-1 only, leaving X2-1 through X2-8 and X4-2 through X6-8 empty.

## PHYSICAL CHANNEL ADDRESS and TURNKEY PANEL ENABLE

The jumper header X7 is used to set up the physical channel address of the MFP and whether or not the MFP is turnkey panel enabled.




NOTE: PIN 1 IS LOCATED ON BOTH MFP AND MFP DI AT THE SIDE OF THE HEADER CLOSEST TO THE SelBUS.

87D3J03

Figure 2-2. Typical Jumper Headers

**Table 2-1. SelBUS Priority Jumpers**

| SelBUS Priority Recognition | SelBUS Priority Generation | SelBUS Priority Level |
|---|---|---|
| X1-2 | X2-1 | 1 |
| X1-3 | X2-2 | 2 |
| X1-4 | X2-3 | 3 |
| X1-5 | X2-4 | 4 |
| X1-6 | X2-5 | 5 |
| X1-7 | X2-6 | 6 |
| X1-8 | X2-7 | 7 |
| X3-1 | X2-8 | 8 |
| X3-2 | X4-1 | 9 |
| X3-3 | X4-2 | 10 |
| X3-4 | X4-3 | 11 |
| X3-5 | X4-4 | 12 |
| X3-6 | X4-5 | 13 |
| X3-7 | X4-6 | 14 |
| X3-8 | X4-7 | 15 |
| X5-1 | X4-8 | 16 |
| X5-2 | X6-1 | 17 |
| X5-3 | X6-2 | 18 |
| X5-4 | X6-3 | 19 |
| X5-5 | X6-4 | 20 |
| X5-6 | X6-5 | 21 |
| X5-7 | X6-6 | 22 |
| X5-8 | X6-7 | not used |
| | X6-8 | not used |

### Physical Channel Address

The physical channel address is set up at jumper header X7, pins 2 through 5 (X7-2 through X7-5). Jumpers X7-2 through X7-5 determine the most significant four bits of the physical channel address of the MFP, with jumper X7-2 determining the most significant bit of the physical channel address.

**NOTE**

The CPU scratch pad is used to map the logical channel addresses to the physical channel address that is jumpered on X7.

When setting up the MFP's physical channel address, valid class "F" channel addresses are limited to one of the following: 06, 0E, 16, 1E, 26, 2E, 36, 3E, 46, 4E, 56, 5E, 66, 6E, 76, or 7E. By using those class "F" channel addresses, the RTF address will always be one greater (higher) than the class "F" channel address. For example: an MFP with X7-2 jumpered will respond to class "F" software commands directed to the even physical channel address 46, and RTF software commands directed to the odd physical channel address 47.

The MFP requires eight physical channel addresses on the SelBUS. Four are used as direct memory access (DMA) ports for peripheral devices; one as the MFP channel data port; one as the class "F" protocol port; one as the real time function (RTF) protocol port; and one is reserved for future enhancement. When software performs class "F" I/O to the MFP it uses the address assigned to the class "F" protocol port. When software performs RTF functions it uses the RTF protocol port. See table 2-2 for a description of these jumpers.

### Turnkey Panel Enable

The turnkey panel enable is set up at jumper header X7, pin 1 (X7-1). When a jumper is NOT inserted at X7-1, the MFP is enabled for a turnkey panel. When a jumper is inserted in X7-1, the MFP is disabled for a turnkey panel. Note: when more than one MFP is installed on a computer system, only one of them should be enabled for a turnkey panel, all others should be disabled. See table 2-2 for a description of this jumper.

## INTERVAL TIMER CLOCK RATE

The jumper header X8 is used to set up the interval timer clock rate of the MFP. Only one jumper should be inserted into this header at any one time. If no jumpers are inserted, the interval timer will not count. See table 2-3 for a description of these jumpers.

## REAL-TIME CLOCK CONTROL RATE

The jumper header X9 is used to set up the real-time clock control rate for MFP. Only one jumper should be inserted into this header at any one time. If no jumpers are inserted, the real-time clock will not tic. See table 2-4 for a description of these jumpers. Note: the external clock rate is located at pin 1 of the J5 connector located on the MFP device interface circuit card.

### 2.2.2 Indicators, MFP Circuit Card

The indicators on the MFP circuit card enable the user to determine the status of the MFP at any given time. The indicators are light emitting diodes (LED's). The MFP contains four LED's, they are located next to the on line/off line switch along the front edge of the MFP circuit card, see figure 2-1. The LED's are labeled DS1, DS2, DS3, and DS4; they indicate the following status:

- DS1 - when the MFP is on line and operational, and DS1 is illuminated, it indicates that a channel initialization (INCH) was performed.
- DS2 - when illuminated, indicates that a class "F" interrupt is pending.
- DS3 - when illuminated, indicates that a class "3" interrupt request is active.
- DS4 - when illuminated, indicates that the MFP is on line and operational. If the LED is not illuminated it indicates one of three cases:

    1) The on line/off line switch is in the off line position.

**Table 2-2. Physical Channel Address and Turnkey Panel Jumpers**

| Jumper (X7- ): Address Bit*: | 1 | 2 1 | 3 2 | 4 3 | 5 4 | 5 | 6 | 7 | Description |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | | | | | | | | Turnkey panel enabled |
| | 1 | | | | | | | | Turnkey panel disabled |
| | | | | | | | | | MFP Physical Channel Address: |
| | | 0 | 0 | 0 | 0 | | | | 06 |
| | | 0 | 0 | 0 | 1 | | | | 0E |
| | | 0 | 0 | 1 | 0 | | | | 16 |
| | | 0 | 0 | 1 | 1 | | | | 1E |
| | | 0 | 1 | 0 | 0 | | | | 26 |
| | | 0 | 1 | 0 | 1 | | | | 2E |
| | | 0 | 1 | 1 | 0 | | | | 36 |
| | | 0 | 1 | 1 | 1 | | | | 3E |
| | | 1 | 0 | 0 | 0 | | | | 46 |
| | | 1 | 0 | 0 | 1 | | | | 4E |
| | | 1 | 0 | 1 | 0 | | | | 56 |
| | | 1 | 0 | 1 | 1 | | | | 5E |
| | | 1 | 1 | 0 | 0 | | | | 66 |
| | | 1 | 1 | 0 | 1 | | | | 6E |
| | | 1 | 1 | 1 | 0 | | | | 76 |
| | | 1 | 1 | 1 | 1 | | | | 7E |
| | | | | | | 0 | 0 | 0 | Data port 0 - SCSI 1 |
| | | | | | | 0 | 0 | 1 | Data port 1 - SCSI 2 |
| | | | | | | 0 | 1 | 0 | Data port 2 - Async/Line printer |
| | | | | | | 0 | 1 | 1 | Data port 3 - Exspansion port |
| | | | | | | 1 | 0 | 0 | Channel data port |
| | | | | | | 1 | 0 | 1 | reserved for future use |
| | | | | | | 1 | 1 | 0 | Class "F" protocol port |
| | | | | | | 1 | 1 | 1 | Real time functions protocol port |

0 indicates NO jumper inserted.
1 indicates jumper is inserted.

Note: jumpers X7-5, X7-6, and X7-7 are not used, do not insert jumpers at these locations.

* address bit 0 is always zero; address bits 5, 6, and 7 are hard wired.

**Table 2-3. Interval Timer Clock Rate Jumpers**

| Jumper | Clock Rate |
|---|---|
| X8-1 | 76.8 microseconds per tic |
| X8-2 | 38.4 microseconds per tic |
| X8-3 | 19.2 microseconds per tic |
| X8-4 | 9.6 microseconds per tic |
| X8-5 | 4.8 microseconds per tic |
| X8-6 | 2.4 microseconds per tic |
| X8-7 | 1.2 microseconds per tic |
| X8-8 | 600 nanoseconds per tic |

**Table 2-4. Real-Time Clock Rate Jumpers**

| Jumper | Clock Rate |
|---|---|
| X9-1 | 120/100 hertz |
| X9-2 | 60/50 hertz |
| X9-3 | external clock frequency from connector J5 on the MFP DI card. |
| X9-4 | reserved (do NOT insert jumper) |

2) The MFP has received a SelBUS reset or class "F" reset channel, after going through initialization procedures, the LED will illuminate again.

3) The MFP has encountered an unrecoverable hardware or firmware error.

To determine if the DS4 LED is out because of an unrecoverable error, switch the on line/off line switch to the on line position and ensure that the SelBUS reset line is not true. If DS4 remains off, DS1 through DS3 will provide a partially coded message as listed in table 2-5. Resetting the MFP through a SelBUS reset or a channel reset will cause the MFP to attempt to recover from the error condition.

### 2.2.3 Connectors, MFP Circuit Card

The connectors, on the MFP circuit card, are provided for development and test purposes. During normal operation of the MFP there should be no connections made. The connectors are designed to interface the MFP's microprocessor with the Hilevel Corporation Emulyzer. The connectors are labeled J1 through J4, they provide interface as follows:

- J1 - address and control
- J2 - ROM simulation (ROM-SIM) bits 32 through 47
- J3 - ROM simulation (ROM-SIM) bits 00 through 31
- J4 - Y-bus

**Table 2-5. MFP Error Codes Displayed by DS1 - DS3**

| Error Code | Description |
|---|---|
| 0 | Channel Firmware Invalid Activity or<br>Invalid CPU Que Function or<br>Channel Firmware SIQ Underflow or<br>SI NPM |
| 1 | Channel Firmware IOCD Que Overflow or<br>MC1 Que Underflow or<br>Channel Firmware MC2 or<br>MC3 Que Underflow |
| 2 | Channel Firmware MS1 or<br>MS2 Que Underflow or<br>Channel Firmware MS3 Que Underflow |
| 3 | Invalid External Interrupt Function or<br>Async Turnkey Panel Function Request Error |
| 4 | Invalid Class "0" RSTX or<br>Invalid message received from the SCSI Controller or<br>Invalid message received from the Expansion Port |
| 5 | Invalid Turnkey Panel SelBUS transfer or<br>Turnkey Panel Protocol Error |

#### 2.2.3.1 Connector J1, Address and Control

Connector J1, address and control, is a 26 pin connector. It interfaces with the address and control pod of the Emulyzer. The pin numbers and descriptions are provided in table 2-6.

#### 2.2.3.2 Connectors J2 and J3, ROM-SIM

Connectors J2 and J3, ROM simulation (ROM-SIM), are 40 pin connectors. They provide the means to interface the MFP with an external RAM or ROM source. Using the PROM disable input on the address and control connector, the external RAM or ROM is used to provide the microcode for the MFP's microprocessor. Connector J2 provides the interface for the 16 high order bits. Connector J3 provides the interface for the 32 low order bits. There are a total of 48 high and low order bits. The pin numbers and descriptions for connector J2 are listed in table 2-7 and connector J3 are listed in table 2-8.

#### 2.2.3.3 Connector J4, Y-Bus

Connector J4, Y-bus, is a 20 pin connector. It provides the means to monitor the information on the Y-bus. The clock used to synchronize data recovery on the Y-bus is located at pin 21 of connector J1. The pin numbers and descriptions for connector J4 are listed in table 2-9.

**Table 2-6. Connector J1, Address and Control**

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | Program Counter 0 Out | 2 | Program Counter 1 Out |
| 3 | Program Counter 2 Out | 4 | Program Counter 3 Out |
| 5 | Ground | 6 | Program Counter 4 Out |
| 7 | Program Counter 5 Out | 8 | Program Counter 6 Out |
| 9 | Program Counter 7 Out | 10 | Ground |
| 11 | Program Counter 8 Out | 12 | Program Counter 9 Out |
| 13 | Program Counter 10 Out | 14 | Program Counter 11 Out |
| 15 | Ground | 16 | Program Counter 12 Out |
| 17 | Program Counter 13 Out | 18 | Program Counter 14 Out |
| 19 | Program Counter 15 Out | 20 | Ground |
| 21 | Trace Clock Out | 22 | no connection |
| 23 | Low Stop Clock In | 24 | Low PROM Disable |
| 25 | Low Reset SelBUS In | 26 | Ground |

**Table 2-7. Connector J2, ROM-SIM (bits 32 - 47)**

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | Ground | 2 | no connection |
| 3 | no connection | 4 | no connection |
| 5 | no connection | 6 | Ground |
| 7 | no connection | 8 | no connection |
| 9 | no connection | 10 | no connection |
| 11 | Ground | 12 | no connection |
| 13 | no connection | 14 | no connection |
| 15 | no connection | 16 | Ground |
| 17 | no connection | 18 | no connection |
| 19 | no connection | 20 | no connection |
| 21 | Ground | 22 | CROM Bit 47 |
| 23 | CROM Bit 46 | 24 | CROM Bit 45 |
| 25 | CROM Bit 44 | 26 | Ground |
| 27 | CROM Bit 43 | 28 | CROM Bit 42 |
| 29 | CROM Bit 41 | 30 | CROM Bit 40 |
| 31 | Ground | 32 | CROM Bit 39 |
| 33 | CROM Bit 38 | 34 | CROM Bit 37 |
| 35 | CROM Bit 36 | 36 | Ground |
| 37 | CROM Bit 35 | 38 | CROM Bit 34 |
| 39 | CROM Bit 33 | 40 | CROM Bit 32 |

Table 2-8. Connector J3, ROM-SIM (bits 00 - 31)

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | Ground | 2 | CROM Bit 31 |
| 3 | CROM Bit 30 | 4 | CROM Bit 29 |
| 5 | CROM Bit 28 | 6 | Ground |
| 7 | CROM Bit 27 | 8 | CROM Bit 26 |
| 9 | CROM Bit 25 | 10 | CROM Bit 24 |
| 11 | Ground | 12 | CROM Bit 23 |
| 13 | CROM Bit 22 | 14 | CROM Bit 21 |
| 15 | CROM Bit 20 | 16 | Ground |
| 17 | CROM Bit 19 | 18 | CROM Bit 18 |
| 19 | CROM Bit 17 | 20 | CROM Bit 16 |
| 21 | Ground | 22 | CROM Bit 15 |
| 23 | CROM Bit 14 | 24 | CROM Bit 13 |
| 25 | CROM Bit 12 | 26 | Ground |
| 27 | CROM Bit 11 | 28 | CROM Bit 10 |
| 29 | CROM Bit 9 | 30 | CROM Bit 8 |
| 31 | Ground | 32 | CROM Bit 7 |
| 33 | CROM Bit 6 | 34 | CROM Bit 5 |
| 35 | CROM Bit 4 | 36 | Ground |
| 37 | CROM Bit 3 | 38 | CROM Bit 2 |
| 39 | CROM Bit 1 | 40 | CROM Bit 0 |

Table 2-9. Connector J4, Y-Bus

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | Y-Bus Bit 0 | 2 | Y-Bus Bit 1 |
| 3 | Y-Bus Bit 2 | 4 | Y-Bus Bit 3 |
| 5 | Ground | 6 | Y-Bus Bit 4 |
| 7 | Y-Bus Bit 5 | 8 | Y-Bus Bit 6 |
| 9 | Y-Bus Bit 7 | 10 | Ground |
| 11 | Y-Bus Bit 8 | 12 | Y-Bus Bit 9 |
| 13 | Y-Bus Bit 10 | 14 | Y-Bus Bit 11 |
| 15 | Ground | 16 | Y-Bus Bit 12 |
| 17 | Y-Bus Bit 13 | 18 | Y-Bus Bit 14 |
| 19 | Y-Bus Bit 15 | 20 | Ground |

## 2.3 Controls, Indicators, and Connectors, MFP DI Circuit Card

The MFP device interface (DI) circuit card contains controls and connectors; however, it does not contain any indicators. The controls consist of jumpers; the connectors consist of J connectors. See figure 2-3 for the actual location of the above mentioned components. Each component is described in detail as you progress through this chapter.

### 2.3.1 Jumpers, MFP DI Circuit Card

The MFP DI circuit card contains four sets of jumpers. Each set of jumpers make up one jumper header. The jumper headers are labeled from X1 through X4 and called out in figure 2-3, as well as, the logic drawings and the actual MFP DI circuit card. The function of the jumper headers are as follows:

- X1 - SCSI buses, address and reset enable
- X2 - not used
- X3 - turnkey panel baud rate
- X4 - line printer control

#### SCSI BUS 1 ADDRESS and RESET ENABLE

Jumper header X1 (X1-1 through X1-5) is used to set up the MFP address on SCSI bus 1 and enable/disable the reset of SCSI bus 1. Jumpers X1-1 through X1-3 determine the MFP's address on SCSI bus 1. Jumper X1-1 is the least significant of the three jumpers. Jumper X1-4 controls the MFP's ability to reset SCSI bus 1. Normally jumper X1-4 is inserted; however, it may be removed in cases where there are multiple initiators on the SCSI bus 1. Removing jumper X1-4 prevents channel or SelBUS resets from reseting the SCSI bus 1. Jumper X1-5, when inserted, forces the MFP to source +4.3 volts to the SCSI bus line terminator power. The MFP physical channel address on the SCSI Bus 1 can not be used by SCSI device controllers. Any I/O that is directed to this address will be rejected. See table 2-10 for a description of these jumpers.

#### SCSI BUS 2 ADDRESS and RESET ENABLE

The jumper header X1 (X1-6 through X1-10) is used to set up the MFP address on SCSI bus 2 and enable/disable the reset of SCSI bus 2. Jumpers X1-6 through X1-8 determine the MFP's address on SCSI bus 2. Jumper X1-6 is the least significant of the three jumpers. Jumper X1-9 controls the MFP's ability to reset SCSI bus 2. Normally jumper X1-9 is inserted; however, it may be removed in cases where there are multiple initiators on the SCSI bus 2. Removing jumper X1-9 prevents channel or SelBUS resets from reseting the SCSI bus 2. Jumper X1-10, when inserted, forces the MFP to source +4.3 volts to SCSI bus line terminator power. The MFP physical channel address on the SCSI Bus 2 can not be used by SCSI device controllers. Any I/O that is directed to this address will be rejected. See table 2-11 for a description of these jumpers.

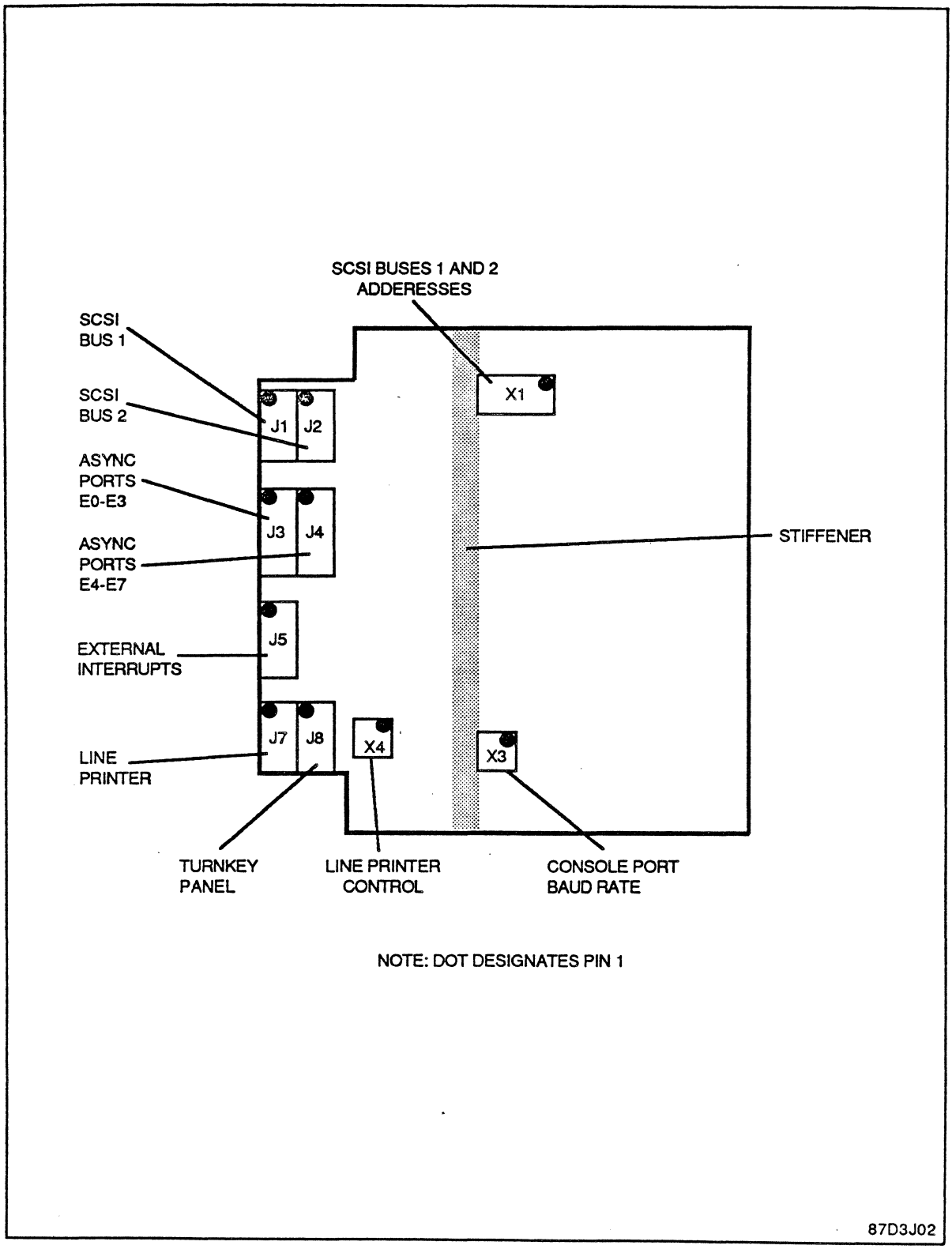



Figure 2-3. MFP DI Circuit Card Layout

**Table 2-10. SCSI Bus 1 Jumpers**

| Jumper: | X1-1 | X1-2 | X1-3 | X1-4 | X1-5 | Description |
|---|---|---|---|---|---|---|
| | | | | | | Allocates Channel Device Subaddresses: |
| | 0 | 0 | 0 | | | 0 - 7 |
| | 1 | 0 | 0 | | | 8 - F |
| | 0 | 1 | 0 | | | 10 - 17 |
| | 1 | 1 | 0 | | | 18 - 1F |
| | 0 | 0 | 1 | | | 20 - 27 |
| | 1 | 0 | 1 | | | 28 - 2F |
| | 0 | 1 | 1 | | | 30 - 37 |
| | 1 | 1 | 1 | | | 38 - 3F |
| | | | | 1 | | enable bus reset |
| | | | | 0 | | disable bus reset |
| | | | | | 1 | source terminator power |
| | | | | | 0 | no connection |

0 indicates NO jumper inserted.
1 indicates jumper is inserted.

## TURNKEY PANEL BAUD RATE

The jumper header X3 is used to set up the MFP's asynchronous port zero as a turnkey panel port. Jumper X3-1 determines whether or not asynchronous port zero is a turnkey panel port. Jumper X3-2 determines whether the turnkey panel port is connected to a Console CRT or a modem. Jumpers X3-3 and X3-4 are not used, no jumper should be inserted in these positions. Jumpers X3-5 through X3-8 are used to determine the baud rate of the turnkey panel port. See table 2-12 for a description of these jumpers.

## LINE PRINTER CONTROL

The jumper header X4 is used to set up the MFP line printer port. Jumpers X4-1 and X4-2 are reserved for future use and should not have any jumpers inserted in them. Jumpers X4-3 and X4-4 set up the paper instruction bit on the line printer interface. One jumper should be inserted in either X4-3 or X4-4, but not both. By inserting a jumper in X4-3, the paper instruction data bit on the line printer interface is grounded. By inserting a jumper in X4-4, the most significant data bit of each data byte drives the paper instruction bit onto the line printer interface.

**Table 2-11. SCSI Bus 2 Jumpers**

| Jumper: | X1-6 | X1-7 | X1-8 | X1-9 | X1-10 | Description |
|---|---|---|---|---|---|---|
| | | | | | | Allocates Channel Device Subaddresses: |
| | 0 | 0 | 0 | | | 0 - 7 |
| | 1 | 0 | 0 | | | 8 - F |
| | 0 | 1 | 0 | | | 10 - 17 |
| | 1 | 1 | 0 | | | 18 - 1F |
| | 0 | 0 | 1 | | | 20 - 27 |
| | 1 | 0 | 1 | | | 28 - 2F |
| | 0 | 1 | 1 | | | 30 - 37 |
| | 1 | 1 | 1 | | | 38 - 3F |
| | | | | 1 | | enable bus reset |
| | | | | 0 | | disable bus reset |
| | | | | | 1 | source terminator power |
| | | | | | 0 | no connection |
| 0 indicates NO jumper inserted.<br>1 indicates jumper is inserted. | | | | | | |

### 2.3.2 Connectors, MFP DI Circuit Card

The connectors, on the MFP DI circuit card, provide the interface between the MFP and the I/O devices. The connectors are labeled J1 through J8 and interface with the following I/O devices:

- J1 - SCSI bus 1
- J2 - SCSI bus 2
- J3 - asynchronous ports 0 through 3
- J4 - asynchronous ports 4 through 7
- J5 - external interrupts and clock
- J6 - not used
- J7 - line printer
- J8 - turnkey panel

#### 2.3.2.1 Connector J1, SCSI Bus 1

Connector J1, SCSI bus 1, is a 50 pin connector. It provides the means to interface the SCSI bus 1 and its peripheral devices with the MFP. The MFP only supports single ended drivers/receivers. The pin numbers and descriptions are provided in table 2-13.

**Table 2-12. Turnkey Panel Port Configuration Jumpers**

| Jumper (X3- ): | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | Description | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | | | | | | | | async port 0 is console | | |
| | 1 | | | | | | | | async port 0 is not console | | |
| | | 0 | | | | | | | console is a terminal | | |
| | | 1 | | | | | | | console is a modem | | |
| | | | X | | | | | | not used | | |
| | | | | X | | | | | not used | | |
| | | | | | | | | | **Baud Rate** | **Parity** | **Character Length** |
| | | | | | 0 | 0 | 0 | 0 | 110 | even | 7 |
| | | | | | 0 | 0 | 0 | 1 | 300 | even | 7 |
| | | | | | 0 | 0 | 1 | 0 | 110 | none | 8 |
| | | | | | 0 | 0 | 1 | 1 | 150 | none | 8 |
| | | | | | 0 | 1 | 0 | 0 | 300 | none | 8 |
| | | | | | 0 | 1 | 0 | 1 | 600 | none | 8 |
| | | | | | 0 | 1 | 1 | 0 | 1200 | none | 8 |
| | | | | | 0 | 1 | 1 | 1 | 2000 | none | 8 |
| | | | | | 1 | 0 | 0 | 0 | 2400 | none | 8 |
| | | | | | 1 | 0 | 0 | 1 | 4800 | none | 8 |
| | | | | | 1 | 0 | 1 | 0 | 1800 | none | 8 |
| | | | | | 1 | 0 | 1 | 1 | 9600 | none | 8 |
| | | | | | 1 | 1 | 0 | 0 | 19.2K | none | 8 |
| | | | | | 1 | 1 | 0 | 1 | 38.4K | none | 8 |
| | | | | | 1 | 1 | 1 | 0 | 16X | none | 8 |
| | | | | | 1 | 1 | 1 | 1 | 1X | none | 8 |

0 indicates NO jumper inserted.
1 indicates jumper is inserted.
X indicates jumper not used.

### 2.3.2.2 Connector J2, SCSI Bus 2

Connector J2, SCSI bus 2, is a 50 pin connector. It provides the means to interface the SCSI bus 2 and its peripheral devices with the MFP. The MFP only supports single ended drivers/receivers. The pin numbers and descriptions are provided in table 2-13.

**Table 2-13. Connectors J1 and J2, SCSI Bus**

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | Ground | 2 | Data Bit 0 |
| 3 | Ground | 4 | Data Bit 1 |
| 5 | Ground | 6 | Data Bit 2 |
| 7 | Ground | 8 | Data Bit 3 |
| 9 | Ground | 10 | Data Bit 4 |
| 11 | Ground | 12 | Data Bit 5 |
| 13 | Ground | 14 | Data Bit 6 |
| 15 | Ground | 16 | Data Bit 7 |
| 17 | Ground | 18 | Data Bit Parity |
| 19 | Ground | 20 | Ground |
| 21 | Ground | 22 | Ground |
| 23 | Ground | 24 | Ground |
| 25 | See Note | 26 | Termination Power |
| 27 | Ground | 28 | Ground |
| 29 | Ground | 30 | Ground |
| 31 | Ground | 32 | Attention |
| 33 | Ground | 34 | Ground |
| 35 | Ground | 36 | Busy |
| 37 | Ground | 38 | Acknowledge |
| 39 | Ground | 40 | Reset |
| 41 | Ground | 42 | Message |
| 43 | Ground | 44 | Select |
| 45 | Ground | 46 | Command/Data |
| 47 | Ground | 48 | Request |
| 49 | Ground | 50 | Input/Output |

Note: All odd pins except pin 25 shall be connected to ground. Pin 25 should be left open.

### 2.3.2.3 Connector J3, Asynchronous Ports 0 - 3

Connector J3, asynchronous ports 0 through 3, is a 60 pin connector. It provides the means to interface asynchronous ports 0 through 3 with the MFP. Note that port 0 may be used for the system turnkey panel/console CRT. Also provided on this connector is the interface for the optional external clock inputs for the asynchronous interface. The pin numbers and descriptions are provided in table 2-14.

### 2.3.2.4 Connector J4, Asynchronous Ports 4 - 7

Connector J4, asynchronous ports 4 through 7, is a 60 pin connector. It provides the means to interface asynchronous ports 4 through 7 with the MFP. Also provided on this connector is the interface for the optional external clock inputs for the asynchronous interface. The pin numbers and descriptions are provided in table 2-15.

**Table 2-14. Connector J3, Asynchronous Ports 0 - 3**

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | Port 0 Transmit Data Out | 2 | Port 0 Request to Send Out |
| 3 | Port 0 Data Terminal Ready Out | 4 | Port 0 Receive Data In |
| 5 | Port 0 Clear to Send In | 6 | Port 0 Data Set Ready In |
| 7 | Port 0 Data Carrier Detect In | 8 | Port 0 Ring In |
| 9 | Ground | 10 | Ground |
| 11 | Port 1 Transmit Data Out | 12 | Port 1 Request to Send Out |
| 13 | Port 1 Data Terminal Ready Out | 14 | Port 1 Receive Data In |
| 15 | Port 1 Clear to Send In | 16 | Port 1 Data Set Ready In |
| 17 | Port 1 Data Carrier Detect In | 18 | Port 1 Ring In |
| 19 | Ground | 20 | Ground |
| 21 | Port 2 Transmit Data Out | 22 | Port 2 Request to Send Out |
| 23 | Port 2 Data Terminal Ready Out | 24 | Port 2 Receive Data In |
| 25 | Port 2 Clear to Send In | 26 | Port 2 Data Set Ready In |
| 27 | Port 2 Data Carrier Detect In | 28 | Port 2 Ring In |
| 29 | Ground | 30 | Ground |
| 31 | Port 3 Transmit Data Out | 32 | Port 3 Request to Send Out |
| 33 | Port 3 Data Terminal Ready Out | 34 | Port 3 Receive Data In |
| 35 | Port 3 Clear to Send In | 36 | Port 3 Data Set Ready In |
| 37 | Port 3 Data Carrier Detect In | 38 | Port 3 Ring In |
| 39 | Ground | 40 | Ground |
| 41 | no connection | 42 | Ground |
| 43 | +5 Volts | 44 | Ground |
| 45 | no connection | 46 | Ground |
| 47 | no connection | 48 | no connection |
| 49 | Ground | 50 | no connection |
| 51 | Port 0 Transmit Clock In | 52 | Port 0 Receive Clock In |
| 53 | no connection | 54 | Port 1 Transmit Clock In |
| 55 | Port 1 Receive Clock In | 56 | Port 2 Transmit Clock In |
| 57 | Port 2 Receive Clock In | 58 | no connection |
| 59 | Port 3 Transmit Clock In | 60 | Port 3 Receive Clock In |

### 2.3.2.5 Connector J5, External Interrupts and Clock

Connector J5, external interrupts and clock, is a 26 pin connector. It provides the means to interface the external interrupts and clock with the MFP. The external clock is used to drive the systems real time clock. All signals, except the external clock, are low true pulses. The pin numbers and descriptions are provided in table 2-16.

**Table 2-15. Connector J4, Asynchronous Ports 4 - 7**

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | Port 4 Transmit Data Out | 2 | Port 4 Request to Send Out |
| 3 | Port 4 Data Terminal Ready Out | 4 | Port 4 Receive Data In |
| 5 | Port 4 Clear to Send In | 6 | Port 4 Data Set Ready In |
| 7 | Port 4 Data Carrier Detect In | 8 | Port 4 Ring In |
| 9 | Ground | 10 | Ground |
| 11 | Port 5 Transmit Data Out | 12 | Port 5 Request to Send Out |
| 13 | Port 5 Data Terminal Ready Out | 14 | Port 5 Receive Data In |
| 15 | Port 5 Clear to Send In | 16 | Port 5 Data Set Ready In |
| 17 | Port 5 Data Carrier Detect In | 18 | Port 5 Ring In |
| 19 | Ground | 20 | Ground |
| 21 | Port 6 Transmit Data Out | 22 | Port 6 Request to Send Out |
| 23 | Port 6 Data Terminal Ready Out | 24 | Port 6 Receive Data In |
| 25 | Port 6 Clear to Send In | 26 | Port 6 Data Set Ready In |
| 27 | Port 6 Data Carrier Detect In | 28 | Port 6 Ring In |
| 29 | Ground | 30 | Ground |
| 31 | Port 7 Transmit Data Out | 32 | Port 7 Request to Send Out |
| 33 | Port 7 Data Terminal Ready Out | 34 | Port 7 Receive Data In |
| 35 | Port 7 Clear to Send In | 36 | Port 7 Data Set Ready In |
| 37 | Port 7 Data Carrier Detect In | 38 | Port 7 Ring In |
| 39 | Ground | 40 | Ground |
| 41 | no connection | 42 | Ground |
| 43 | +5 Volts | 44 | Ground |
| 45 | no connection | 46 | Ground |
| 47 | no connection | 48 | no connection |
| 49 | Ground | 50 | no connection |
| 51 | Port 4 Transmit Clock In | 52 | Port 4 Receive Clock In |
| 53 | no connection | 54 | Port 5 Transmit Clock In |
| 55 | Port 5 Receive Clock In | 56 | Port 6 Transmit Clock In |
| 57 | Port 6 Receive Clock In | 58 | no connection |
| 59 | Port 7 Transmit Clock In | 60 | Port 7 Receive Clock In |

### 2.3.2.6 Connector J7, Line Printer

Connector J7, line printer, is a 40 pin connector. It provides the means to interface the line printer with the MFP. The pin numbers and descriptions are provided in table 2-17.

### 2.3.2.7 Connector J8, Turnkey Panel

Connector J8, turnkey panel, is a 50 pin connector. It provides the means to interface the turnkey panel and CPU with the MFP. The pin numbers and descriptions are provided in table 2-18.

**Table 2-16. Connector J5, External Interrupts and Clock**

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | External Clock Source | 2 | Ground |
| 3 | External Interrupt In 0 | 4 | External Interrupt Out/Acknowledge 0 |
| 5 | External Interrupt In 1 | 6 | External Interrupt Out/Acknowledge 1 |
| 7 | External Interrupt In 2 | 8 | External Interrupt Out/Acknowledge 2 |
| 9 | External Interrupt In 3 | 10 | External Interrupt Out/Acknowledge 3 |
| 11 | External Interrupt In 4 | 12 | External Interrupt Out/Acknowledge 4 |
| 13 | External Interrupt In 5 | 14 | External Interrupt Out/Acknowledge 5 |
| 15 | External Interrupt In 6 | 16 | External Interrupt Out/Acknowledge 6 |
| 17 | External Interrupt In 7 | 18 | External Interrupt Out/Acknowledge 7 |
| 19 | External Interrupt In 8 | 20 | External Interrupt Out/Acknowledge 8 |
| 21 | External Interrupt In 9 | 22 | External Interrupt Out/Acknowledge 9 |
| 23 | External Interrupt In 10 | 24 | External Interrupt Out/Acknowledge 10 |
| 25 | External Interrupt In 11 | 26 | External Interrupt Out/Acknowledge 11 |

**Table 2-17. Connector J7, Line Printer**

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | High Power On In | 2 | Ground |
| 3 | Low Clear Buffer Out | 4 | Ground |
| 5 | no connection | 6 | Ground |
| 7 | High Top of Form In | 8 | Ground |
| 9 | High Bottom of Form In | 10 | Ground |
| 11 | High Line Printer Ready In | 12 | Ground |
| 13 | High On Line In | 14 | Ground |
| 15 | Ground | 16 | Low Line Printer Connected In |
| 17 | High Data Demand In | 18 | Ground |
| 19 | High Data Strobe Out | 20 | Ground |
| 21 | High Paper Instruction Data Bit Out | 22 | Ground |
| 23 | Ground | 24 | Ground |
| 25 | High Data Out Bit 7 | 26 | Ground |
| 27 | High Data Out Bit 6 | 28 | Ground |
| 29 | High Data Out Bit 5 | 30 | Ground |
| 31 | High Data Out Bit 4 | 32 | Ground |
| 33 | High Data Out Bit 3 | 34 | Ground |
| 35 | High Data Out Bit 2 | 36 | Ground |
| 37 | High Data Out Bit 1 | 38 | Ground |
| 39 | Low Centronics In | 40 | Ground |

**Table 2-18. Connector J8, Turnkey Panel**

| Pin # | Description | Pin # | Description |
|---|---|---|---|
| 1 | no connection | 2 | Low Attention Out |
| 3 | no connection | 4 | Low IPL Out |
| 5 | no connection | 6 | Low Reset Out |
| 7 | no connection | 8 | Low Clock Override Out |
| 9 | no connection | 10 | no connection |
| 11 | no connection | 12 | Low Run-Halt Transition |
| 13 | no connection | 14 | no connection |
| 15 | no connection | 16 | Low Parity Error In |
| 17 | no connection | 18 | Low Interrupt Acknowledge In |
| 19 | no connection | 20 | Low Run In |
| 21 | no connection | 22 | Low Halt In |
| 23 | no connection | 24 | Low Wait In |
| 25 | no connection | 26 | Low Unpack In |
| 27 | no connection | 28 | Low Master Clear In |
| 29 | no connection | 30 | no connection |
| 31 | no connection | 32 | no connection |
| 33 | no connection | 34 | no connection |
| 35 | no connection | 36 | no connection |
| 37 | no connection | 38 | no connection |
| 39 | no connection | 40 | no connection |
| 41 | no connection | 42 | no connection |
| 43 | no connection | 44 | no connection |
| 45 | no connection | 46 | no connection |
| 47 | no connection | 48 | no connection |
| 49 | no connection | 50 | Low Disable Cache Out |

# CHAPTER 3

# OPERATION

## 3.1 Introduction

This chapter contains the information pertaining to the operation of the Multi-Function Processor (MFP) and its host computer system.

## 3.2 Overview

Operation of the MFP and its host computer system are implemented through the Console CRT. The Console CRT can function in either system turnkey panel mode or operator console mode.

When the Console CRT is in the system turnkey panel mode, the operator is able to control the computer system with commands entered on the Console CRT's key board. The display of the Console CRT will show the current status of the CPU or system memory and echo back the entries made on the keyboard.

When the Console CRT is in the operator console mode, the operator is able to access the operating system of the computer system and perform programming changes. The display of the Console CRT will show the current status of the operating system and echo back the entries made on the keyboard.

Both the system turnkey panel and the operator console modes are implemented in an interleaved manner. They are available to I/O devices at the console port on the MFP. When the console port is not being used for console functions, they may be used as standard asynchronous communications ports (refer to Chapter 2, Turnkey Panel Enable).

When the system turnkey panel mode is selected, the terminal displays system turnkey panel data.

Automatic switching from system turnkey panel mode to operator console mode occurs when the RUN command is entered on the keyboard during system turnkey panel mode.

When the CPU is halted while in the RUN state, the Console CRT port will switch to the system turnkey panel mode and display the current status of the computer system.

## 3.3 System Turnkey Panel Mode

When the Console CRT is in the system turnkey panel mode, the operator is able to control the computer system with commands entered on the Console CRT's key board. The display of the Console CRT will show the current status of the CPU or system memory and echo back the entries made on the keyboard.

Upon powering up the computer system, the baud rate jumpers of the Console CRT port are read and the appropriate baud rate is selected and set. The Console CRT port is placed in the

system turnkey panel mode. The operator is able to execute the reset (RST) and initial program load (IPL) commands, as well as all other system turnkey panel commands. The prompt displayed on the screen requesting the next command is //. This prompt not only indicates a request for the next command, it also indicates to the operator that the Console CRT is in the system turnkey panel mode.

**NOTE**

The data terminal ready (DTR) line remains high (on) even after the ports are switched to the operator console mode, as long as a disconnect command has not been executed. Therefore, care should be taken when certain operator console modems are used, especially when a ring-in indication is expected to occur, because these modems give a busy signal when they detect an attempted ring-in during a DTR indication. When this happens, the operator console port will not receive the ring-in indication and will fail to report the attention status.

When the RUN command is entered during system turnkey panel mode, the display on the Console CRT is switched to the operator console mode and is then able to display milestone messages.

### 3.4 Operator Console Mode

When the Console CRT is in the operator console mode, the port may be used by the software as a normal asynchronous port with some limitations.

To enter the operator console mode from the system turnkey panel mode, the operator must enter @@C on the keyboard and then press carriage return ([CR]). Upon receipt of the carriage return, the firmware of the MFP moves the cursor to the extreme left margin of the next line.

To return the Console CRT to the system turnkey panel mode, the operator must enter @@P [CR]. Once the system turnkey panel mode is selected, // is used as the prompt for the next command. When the Console CRT is in the operator console mode, the prompt is defined by the software. Operator messages that were on the screen will be left there until they scroll off of the screen.

If the RUN command is entered during system turnkey panel mode, the display on the Console CRT is switched to the operator console mode and the CPU is put into the RUN mode, if the CPU was HALTed.

If the Console CRTs are in the system turnkey panel mode and software has a message that it wants to display in the operator console mode, the firmware of the MFP will queue the message to an inactive subchannel and hold it there until the Console CRT is placed back in the operator console mode.

If a RUN to HALT transition occurs, this will automatically switch the Console CRT to the system turnkey panel mode and display the program status word (PSW) and instruction.

### 3.5 System Turnkey Panel Mode, Controls and Indicators

The system turnkey panel mode supports various forms of controls and indicators. The controls are the commands that the operator is able to enter on the Console CRT's keyboard. The indicators are the various displays, visible to the operator, on the Console CRT's screen.

The panel lock function and power switch capability associated with this mode of operation are not supported by the Console CRT.

#### 3.5.1 General Purpose and Base Registers Display

The general purpose and base registers display provides the operator the ability to write data to, and read data from, the general purpose registers and the base registers resident in the CPU. To do this the CPU must first be halted. When the operator requests a read or write of the general purpose registers or base registers, the appropriate command is entered on the Console CRT's keyboard. For example: to read all eight general purpose registers, enter GPR [CR]; to write to general purpose register one, enter GPR1=XXXXXXXX (where XXXXXXXX is hexidecimal data to be written to the register). Once the command is decoded, a line of data displaying the eight general purpose registers will appear on the Console CRT's screen followed, on the next line, by the prompt //. The commands supported by this mode of operation are:

- GPR [CR] - Read general purpose registers 0 - 7
- GPRA=XXXXXXXX [CR] - Write XXXXXXXX to general purpose register A
- BAS [CR] - Read base registers 0 - 7
- BASA=XXXXXXXX [CR] - Write XXXXXXXX to base register A

The general purpose register and base register display consists of one line of data segmented into eight four-byte fields. Each of the eight fields represents the contents of one register. The eight fields displayed, represent the contents of registers zero through seven. The contents of register zero is represented by the left-most field of the display while the contents of register seven is represented by the right-most field. See figure 3-1 for an illustration of this display.

**NOTE**

The output of this display is not dynamic; that is to say, the display is accurate only at the time of execution.

#### 3.5.2 State Indicator Display

The state indicator display provides a read out of the current status of the CPU. It is invoked by the operator when the appropriate command is entered on the Console CRT's key board. Once the command is decoded, several lines of data will appear on the Console CRT's screen followed, on the next line, by the prompt //. See table 3-1 for a description of the state indicators supported and figure 3-2 for an illustration of this display.

Certain state indicators are paired with other state indicators as follows:

Memory Address (MA) - Memory Data (MD)
Memory Address (MA) - Control Switches (CS)
Program Status Word (PSW) - Instruction (INST)
Program Status Doubleword (PSD) - Blank

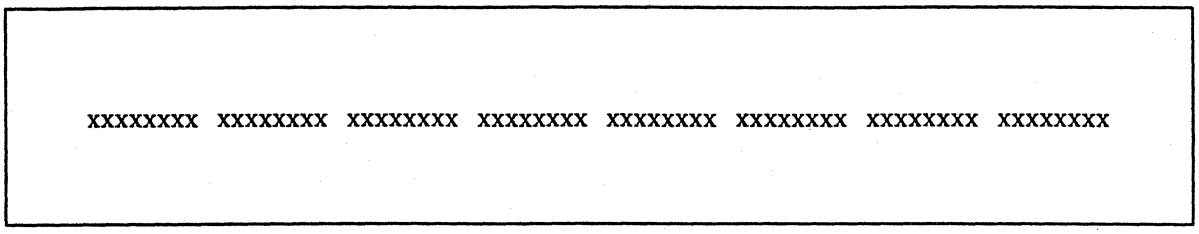



**Figure 3-1. General Purpose and Base Register Display Format**

**Table 3-1. Description of State Indicators**

| Indicator | Description |
|---|---|
| AS | Address stop set (all three stops set) |
| CS | Control switch setting is in adjacent display (XXXXXXXX) |
| EA | Effective address is in adjacent display (XXXXXXXX) |
| HALT | CPU is in the halt mode |
| INST | Instruction is in adjacent display (XXXXXXXX) |
| INT | An interrupt level is active |
| IS | Instruction stop is set |
| MA | Memory address is in adjacent display (XXXXXXXX) |
| MD | Memory data is in adjacent display (XXXXXXXX) |
| OVR | Clock override is activated |
| PE | Parity error from memory |
| PSD | Program status doubleword is in adjacent two displays (XXXXXXXX XXXXXXXX) |
| PSW | Program status word 1 is in adjacent display (XXXXXXXX) |
| RS | Operand read stop is set |
| RUN | CPU is in the run mode |
| ST | Stop, one of the address stop criteria have been met |
| WAIT | Instruction execution is not in progress |
| WS | Operand write stop is set |

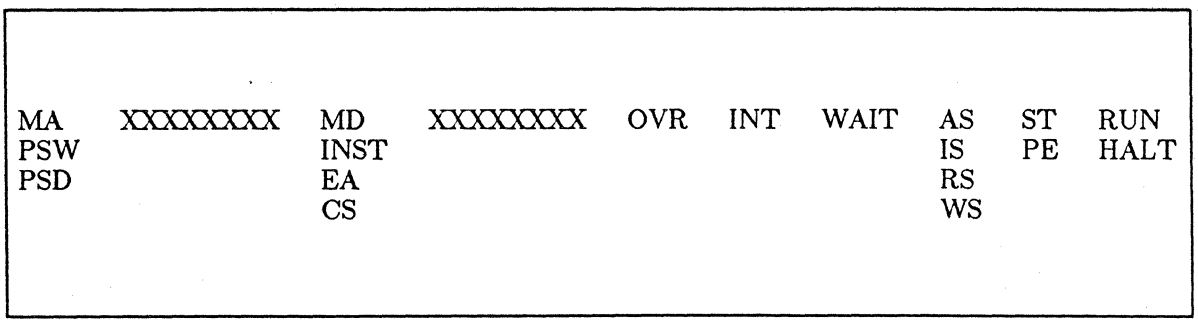



**Figure 3-2. State Indicator Display Format**

When the data is displayed, any state indicators that are not present will be left blank. The data displayed in the left half of the screen, including the two number fields, reflects the information last obtained in response to the operator's command. For example, if the operator enters the clock override command, the corresponding change is shown in the appropriate space designated by the state indicator display; however, the numeric fields at the left of the screen will continue to display the previous information left in that field.

If a read memory command is executed by the operator, and a nonpresent memory occurs, NON PRES MEM will be displayed on the Console CRT's screen followed by the prompt // on the next line.

If a SelBUS error occurs in the course of executing an operator command, SELBUS ERROR will be displayed on the Console CRT screen followed by the prompt // on the next line.

NOTE

The display parameters for the state indicators are not dynamic; that is to say, the display is accurate only at the time of execution, changes in the state of the CPU, while in RUN mode, will continue to happen and not show up on the display.

### 3.5.3 General Display Indicators

For commands not related with general purpose registers, base registers, and state indicator displays, the Console CRT screen displays the general status of the CPU. In other words, when a command is entered, it echos back to the screen and appears next to the prompt //. Once the command is executed, the prompt // is displayed again, indicating to the operator that the previous command entered was executed, and if desired, a new command can be entered.

In the event of an error, such as a misspelled word or wrong command, the display will indicate "OPERATOR ERROR" on the following line, and the prompt // will appear on the next line. Note, the command in error will not be executed. See table 3-2 for a list of Console CRT commands and figure 3-3 for an illustration of this display.

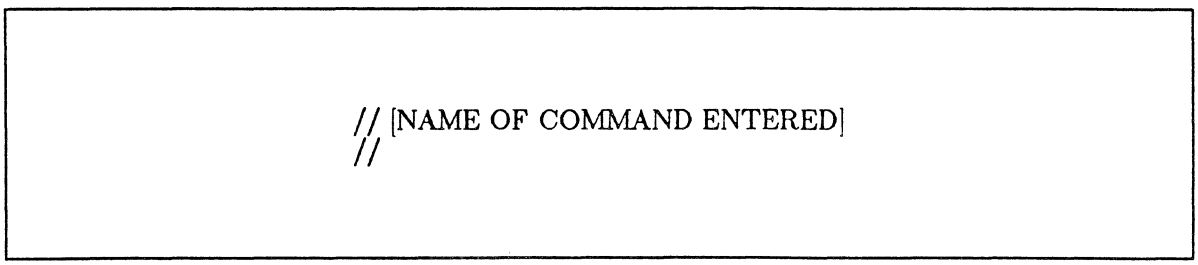



**Figure 3-3. General Display Indicator Format**

**Table 3-2. Console CRT Commands**

| Command | Description |
|---|---|
| @@A | Attention (when in operator console mode) |
| @@C | Enter operator console mode |
| @@P | Enter system turnkey panel mode |
| AS [CR] | Clear address stop |
| AS=XXXXXX [CR] | Set address stop (all 3 stops are set) |
| BAS [CR] | Read base registers 0 - 7 |
| BASA=XXXXXXXX [CR] | Write XXXXXXXX to base register A |
| CLR [CR] | Clear memory |
| [CONTROL] [H] | Deletes last character entered |
| CS [CR] | Read control switch |
| CS=XXXXXX [CR] | Set control switch |
| EA [CR] | Read effective address |
| GPR [CR] | Read general purpose registers 0 - 7 |
| GPRA=XXXXXXXX [CR] | Write XXXXXXXX to general purpose register A |
| HALT [CR] | Halt the operation of the CPU |
| IPL [CR] | Initial program load from default address |
| IPL=XXXX [CR] | Initial program load from XXXX |
| IS [CR] | Clear the instruction stop |
| IS=XXXXXX [CR] | Set the instruction stop to XXXXXX |
| [LF] | Repeat commands (except RTS) |
| MA=XXXXXX [CR] | Read physical memory address location XXXXXX |
| =[CR] | *Increment last memory address and read location |
| MAV=XXXXXXXX [CR] | Read virtual memory address location |
| =[CR] | *Increment last memory address and read location |
| MD=XXXXXXXX [CR] | Write memory data XXXXXXXX into last memory address |
| =XXXXXXXX [CR] | Increment last memory address and write data XXXXXXXX into memory |
| =[CR] | *Increment last memory address and write last data into memory address |
| OVR [CR] | Activates real-time clock override |
| PC=XXXXXX [CR] | Load program counter with XXXXXX |
| PSD [CR] | Read program status doubleword (PSD1 and PSD2) |
| PSD=XXXXXXXX [CR] | Write XXXXXXXX to program status word two (PSD2) |
| PSW [CR] | Read program status word one (PSD1) |
| PSW=XXXXXXXX [CR] | Write XXXXXXXX to program status word one (PSD1) |
| RS [CR] | Clear operand read stop |
| RS=XXXXXX [CR] | Set operand read stop XXXXXX |
| RST [CR] | Reset the computer system |
| RUN [CR] | Requests that the CPU execute instructions |
| STEP [CR] | Requests that the CPU execute only the next instruction |
| =[CR] | *Request that the CPU execute only the next instruction |
| WS [CR] | Clear operand write stop |
| WS=XXXXXX [CR] | Set the operand write stop XXXXXX |

* indicates the continuation of the current command.

### 3.6 Console CRT Operator Commands

The commands supported by the Console CRT are listed in table 3-2. When a command is entered, it must be followed by a carriage return. Some commands require that the CPU be halted before they can be executed, these commands are as follows:

- BAS
- BASA=XXXXXXXX
- CLE
- EA
- EXEC
- GPR
- GPRA=XXXXXXXX
- IPL
- IPL=XXXX
- MAV=XXXXXX
- PC=XXXXXX
- PSD
- PSD=XXXXXXXX
- PSW
- PSW=XXXXXXXX
- STEP
- RST

The lock on and lock off commands are not supported by the MFP.

When a command is entered on the keyboard of the Console CRT and followed by a carriage return [CR], it is decoded by the firmware and executed. As the command is entered on the keyboard, each character is stored in the buffer and echoed (sent) to the screen(s) of the Console CRT(s). After the command has been executed, the appropriate message is displayed, followed by the prompt on the next line. The prompt indicates that the command was executed and a new command can be entered. If the command was not executed, an abbreviated error message will appear on the screen followed by the prompt on the next line.

A Console CRT command followed by a group of X's indicate a hexidecimal number field. The X's indicate that the operator must enter that specific number of characters after the command. For example, if an eight character field is required and the operator only enters five characters, the missing characters are assumed to be zeros; if nine characters are entered, only the eight most significant characters will be read, ignoring the least significant character.

During the execution of some commands, all states of the CPU are checked and the state indicator display is updated.

When an IPL operation is desired, the memory must first be reset to all zeros. To reset or clear the memory the operator must enter CLE, followed by a carriage return, on the keyboard of the Console CRT. This command writes zeros into all of the memory locations. The CLE command must be entered after the computer system is powered up and before an IPL is implemented. After the clear command has been executed, the prompt will be displayed.

### 3.6.1 Attention

Entering @@A [CR] while the system is in the operator console mode causes an attention trap to occur. Entering @@A [CR] while the system in in the system turnkey panel mode causes an operator error to occur with no CPU trap generated.

### 3.6.2 Enter Console Mode

Entering @@C [CR] causes an immediate switch from the system turnkey panel mode to the operator console mode.

### 3.6.3 Enter Panel Mode

Entering @@P [CR] causes an immediate switch from the operator console mode to the system turnkey panel mode. The system turnkey panel mode is evident by the prompt (//).

### 3.6.4 Stop Commands

The stop commands are:

- Clear Address Stop (AS [CR])
- Set Address Stop (AS=XXXXXXXX [CR])
- Clear Instruction Stop (IS [CR])
- Set Instruction Stop (IS=XXXXXXXX [CR])
- Clear Read Operand Stop (RS [CR])
- Set Read Operand Stop (RS=XXXXXXXX [CR])
- Clear Write Operand Stop (WS [CR])
- Set Write Operand Stop (WS=XXXXXXXX [CR])

These stop commands permit the operator to set or reset the address stop conditions. An AS, IS, RS, or WS displayed on the Console CRT's screen indicates that a specific address stop condition has been set up.

When an address stop is active, it causes the CPU to HALT and display ST on the Console CRT's screen if a CPU memory reference occurs at an address previously specified by the stop command.

When a read or write operand stop command is active, it causes the CPU to HALT and display ST on the Console CRT's screen if a CPU read operand or write operand occurs at the address specified by the commands RS=XXXXXXXX or WS=XXXXXXXX.

NOTE

All stop commands cause the CPU to HALT. When ever the CPU encounters a HALT condition, the Console CRT will display ST.

The stop command IS [CR] only relates to an address stop on an instruction fetch operation; RS [CR] only relates to an address stop on a read operand operation; WS [CR] only relates to

an address stop on a write operand operation; and, AS [CR] command relates to all of the above conditions. The stop commands do not function when an I/O channel accesses the specified memory locations.

### SETTING a STOP

To set a stop condition, the operator must decide which type of stop command to use. The appropriate command is then entered on the Console CRT's keyboard followed by the actual stop address (XXXXXXXX). Memory address C-bits, bits 30 and 31, are ignored and truncated by the system turnkey panel.

**NOTE**

When an address match occurs, the CPU is halted; however, the stop is not cleared; therefore, if the address match should occur again, the CPU will be halted again.

Only one stop command can be active at any given time.

### CLEARING a STOP

To clear a stop condition, the operator must enter any stop command (AS, IS, RS, or WS followed by an immediate carriage return) for the stop to be cleared.

### 3.6.5 Clear Memory

The clear memory (CLE [CR]) command writes zeros to all memory addresses, in the system memory. The purpose of this command is to clear parity errors in the memory so that the IPL operation can be performed. The MFP clears all memory even if it has a discontiguous physical address space.

### 3.6.6 Read Control Switches

The read control switches consist of CS [CR] and CS=XXXXXXXX [CR]. The CS [CR] command reads data from the dedicated control switch memory location. The CS=XXXXXXXX [CR] command writes data (XXXXXXXX) to the dedicated control switch memory location. The memory address and contents of the control switch are displayed after each operation. The CPU passed the address of the CS location to the turnkey panel after power up.

### 3.6.7 Rub Out

The rub out command (ASCII 08 [control H]) is used when the computer system is in the system turnkey panel mode to move the cursor back one space and delete the character last entered.

### 3.6.8 Effective Address

The effective address (EA [CR]) command requests the effective address of the instruction currently addressed by the program counter (PC). The effective address is displayed in the second number field of the state indicator. The first number field contains the PSW word 1. The second number field is identified by EA.

### 3.6.9 General Purpose Registers

The general purpose register commands are GPR [CR] and GPRA=XXXXXXXX [CR]. The command GPR [CR] reads the eight general purpose registers as a group, their contents are displayed in the general purpose register display. Refer to the section, General Purpose and Base Register Display, for a complete description of this display and its format. The command GPRA=XXXXXXXX [CR] writes the data specified by XXXXXXXX into the general purpose register specified by A (A being 0 through 7). After this command is executed the contents of all eight general purpose registers are displayed.

### 3.6.10 Base Registers

The base register commands are BAS [CR] and BASA=XXXXXXXX [CR]. The command BAS [CR] reads the eight base registers as a group, their contents are displayed in the base register display. Refer to the section, General Purpose and Base Register Display, for a complete description of this display and its format. The command BASA=XXXXXXXX [CR] writes the data specified by XXXXXXXX into the base register specified by A (A being 0 through 7). After this command is executed the contents of all eight base registers are displayed.

### 3.6.11 Halt

The halt (HALT [CR]) command, when entered while the CPU is running, will stop the CPU from executing instructions from memory. Instead, the CPU is placed in a microroutine that monitors for selected system turnkey panel mode commands. Once the HALT command is executed, the CPU will enact a series of bus transfers to the system turnkey panel; the bus transfers contain the information indicating the current PSW (the instruction at the PC address) and the current state indicator information. Once the bus transfers are complete, the information will be displayed on the screen of the Console CRT. If the CPU is already in the halt mode and the HALT command is entered again, the CPU will remain in the halt mode.

### 3.6.12 Initial Program Load

The initial program load commands consist of IPL [CR] and IPL=XXXX [CR]. When the IPL command is entered, it causes the CPU (CPU in HALT mode) to use a default address for the IPL device. When the IPL=XXXX command is entered, it initiates a microprogram loading sequence so that data can first be read from the address represented by XXXX. Once the IPL from address XXXX is completed and another IPL [CR] is entered, it will cause another IPL from the previous address rather than the default address for the IPL device. The default IPL address is jumpered on the CPU board, refer to the CPU technical manual for a complete description.

### 3.6.13 Line Feed

The line feed (LF) command, while in operator console mode, causes the cursor on the CRT to move down one line without returning it to the left margin. If LF is entered, while in the system turnkey panel mode, after a command has been executed, it causes the reexecution of that previous command. The only command that will not reexecute is the reset (RST) command.

### 3.6.14 Memory Address Read

The memory address read (MA=XXXXXX [CR]) command causes a read of the contents of the memory address XXXXXX. The address and the data are displayed on the line following the command. After a memory address read command has been executed, pressing [CR] again will cause the address to be incremented by one word and the data in the new address will be read. The command MA=XXXXXX [CR] can be used to set up the address so that a subsequent memory write operation can be initiated. Note, memory address C bits, bits 30 and 31, are ignored and truncated by the system turnkey panel.

### 3.6.15 Memory Address Read Virtual

The memory address read virtual (MAV=XXXXXX [CR]) command requests the physical address of a logical address XXXXXX from the CPU. After this address is obtained, a memory read operation of that location is performed. The physical address and its data are displayed on the line following the command. If one-to-one mapping is used, the physical address and the logical address are the same. After a memory address read virtual command has been executed, pressing [CR] again will cause the logical address to be incremented by one and the data in the new corresponding physical address will be read. Note, memory address C bits, bits 30 and 31, are ignored and truncated by the system turnkey panel.

### 3.6.16 Memory Data Write

The memory data write (MD=XXXXXXXX [CR]) command writes data to the memory address previously set up by the execution of the memory read command. When using the MD command, the physical address must be used. The data represented by XXXXXXXX is written to the specified address and then the contents are read back to the display for verification. After the memory write operation is executed, it is only necessary to enter =XXXXXXXX [CR] to write new data to the next memory address. The =XXXXXXXX [CR] command is actually an increment of the original write operation. If the same data is to be written to the next memory address, only the [CR] command is required.

### 3.6.17 Clock Override

The clock override (OVR [CR]) command toggles the clock override state. Activating the clock override state causes a blocking of interrupts from the real-time clock or interval timer. When the OVR command is used to activate the clock override state, OVR is displayed on the Console CRT. If the OVR command is used to clear the clock override state, OVR is removed from the Console CRT's display.

### 3.6.18 Load Program Counter

The load program counter (PC=XXXXXX [CR]) command fetches the PSW and replaces the PC portion with the new PC value. The new PSW is then returned to the CPU. The new PSW1 and instruction are displayed on the Console CRT's screen.

### 3.6.19 Program Status Doubleword

The program status doubleword commands are PSD [CR] and PSD=XXXXXXXX [CR]. These commands read from and write to the second word of the PSD. The PSD [CR] command causes a read of both words, the data is displayed in the two number fields of the state indicator. Refer to the section State Indicator Display for a complete description of the display and its format. Note, the PC value displayed in the first PSD field is a virtual address. The PSD=XXXXXXXX [CR] command writes the new PC value to the second word of the PSD. The new PSD and the system turnkey panel prompt are displayed on the Console CRTs screen, this indicates that the command was executed and the system is waiting for the next command.

### 3.6.20 Program Status Word

The program status word commands are PSW [CR] and PSW=XXXXXXXX [CR]. These commands read from and write to the PSW (the first word of the PSD). When PSW [CR] is entered, it requests the CPU to furnish the physical address of the PC portion of the PSW. This address is then used to fetch the next executable instruction. The PSW command, the instruction, and the system turnkey panel prompt are displayed on the Console CRT's screen. Note, the physical address does not appear in the PC part of the PSW; however, the virtual address does. These conditions are the same for one-to-one mapping. When the PSW=XXXXXXXX [CR] commands is entered, it requests the CPU to change the PSW to XXXXXXXX. The write PC command also exists if the operator only wants to change the contents of the PC. The new PSW1, the next instruction to be executed, and the system turnkey panel prompt, are displayed on the Console CRT's screen.

### 3.6.21 Reset

The reset (RST [CR]) command requests the CPU to perform a system reset. After the reset command is executed, the CPU displays the system turnkey panel PSW information on the Console CRT's screen. The information displayed indicates that the CPU is on line and operational. Entering the reset command when the system is in the halt mode initializes all appropriate logic resident to the SelBUS devices.

### 3.6.22 Run

The run (RUN [CR]) command requests that the CPU be placed in the run mode of operation assuming that it is currently halted. The run command also causes the Console CRT port to enter the console operator mode after a line of data is displayed. The line of data displayed consists of zeros in all of the number fields. If the CPU halts at a later time, the new PSW will be displayed on the Console CRT's screen. If the CPU halts before the Console CRT displays the line of data, the display line will be aborted and the PSW line will be displayed in its place.

Entering the run command to an active Console CRT while the CPU is in the halt mode, causes the CPU to enter the run mode and begin executing instructions from the location specified by the PSW. If the CPU is already in the run mode, it will continue to run. After the

run command is entered via the system turnkey panel, a line of information will be displayed on the Console CRT's screen. This display line contains all zeros in the number fields and the current state of the CPU is displayed in the state indicators.

### 3.6.23 Delete

The delete (ASCII 7F; [SHIFT] [DEL] [CR]) command causes the entire last line of text displayed on the Console CRT's screen to be deleted. When the text has been deleted, a new system turnkey prompt will be displayed.

### 3.6.24 Step

The step (STEP [CR]) command causes the CPU to execute the next sequential instruction only. Once the instruction is executed, the CPU waits for the next command.

# CHAPTER 4

# PROGRAMMING

## 4.1 Introduction

The multi-function processor (MFP) is an integrated channel controller, using Gould CSD class F I/O protocol for transfers on the SelBUS. This chapter describes the MFP's implementation of the class F I/O and features unique to the MFP. Real-time function I/O implementation is described in chapter 5 of this manual; however, the programmer is encouraged to review this chapter for information concerning the commonalities between class 3 and class F I/O systems, CPU scratch pad, and the interrupt system.

### NOTE

In the following paragraphs and illustrations, references to controller and device are generic to multiplexing channels, such as the Input/Output Processor (IOP) or the Disc Processor (DP). With integrated channels, like the MFP, the subchannel is considered to be both the controller and the device.

## 4.2 Programming

The programming information in this manual consists of a description of the instructions and commands executed by the MFP. Some class F instructions are not recognized by the MFP and are not described in this manual. Refer to the reference manual for the computer in which the MFP is installed for complete details of the class F I/O instruction repertoire.

There are two major phases of I/O that concerns software, initiation and termination. This chapter describes both in a general manor. Note, any addresses referenced in figures 4-1 or 4-2, as well as the following text, are standard main memory image address for scratchpad. Subtract x"300" to determine the actual CPU scratchpad address.

### 4.2.1 Initiation of I/O

See figure 4-1. The CPU start I/O (SIO) instruction intiates I/O for MFP "F" class devices. The CPU upon decoding the SIO will calculate the channel and device subaddress by adding the least significant halfword of the instruction to the least significant halfword of a selected general purpose register. If the register field of the instruction is zero, the least significant halfword of the instruction is used alone.

The CPU then uses the channel address as an index into the device portion of CPU scratchpad (starts at x"300") and fetches the entry, which software has initialized at system startup time. The interrupt level portion of this entry is complemented to get the real interrupt priority and added to the interrupt table base address (found at x"3F1") to get this devices interrupt vector. This interrupt vector must have been setup by software to point to an interrupt

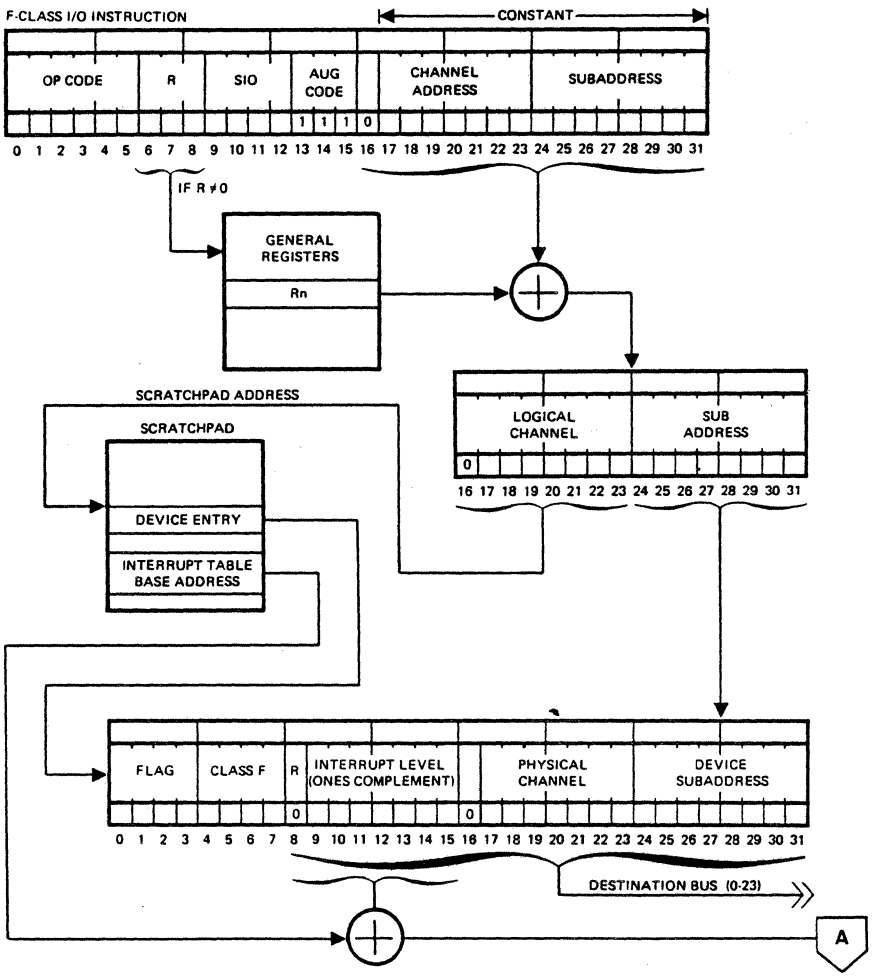



Figure 4-1. Start I/O Instruction Decoding (sheet 1 of 2)

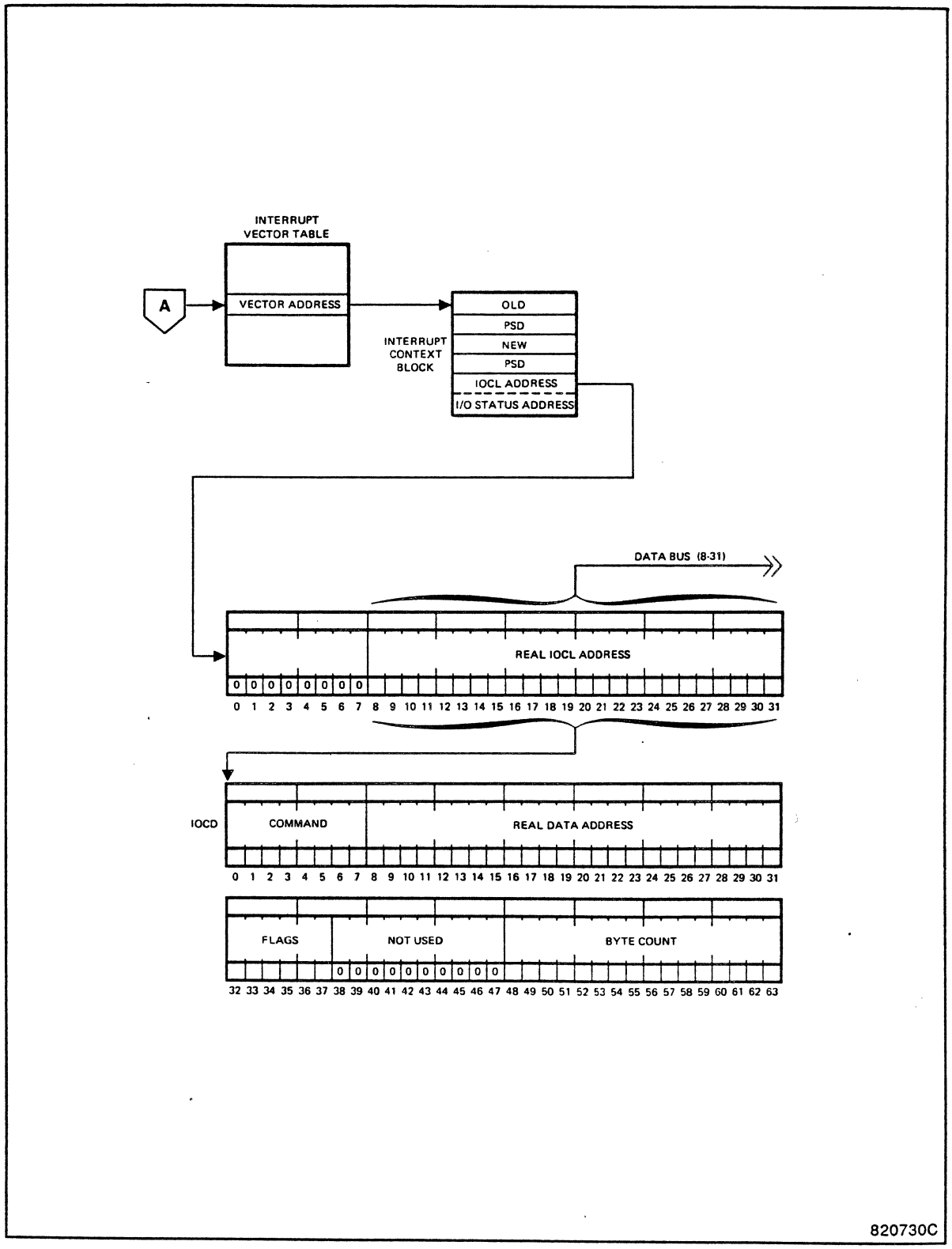



Figure 4-1. Start I/O Instruction Decoding (sheet 2 of 2)

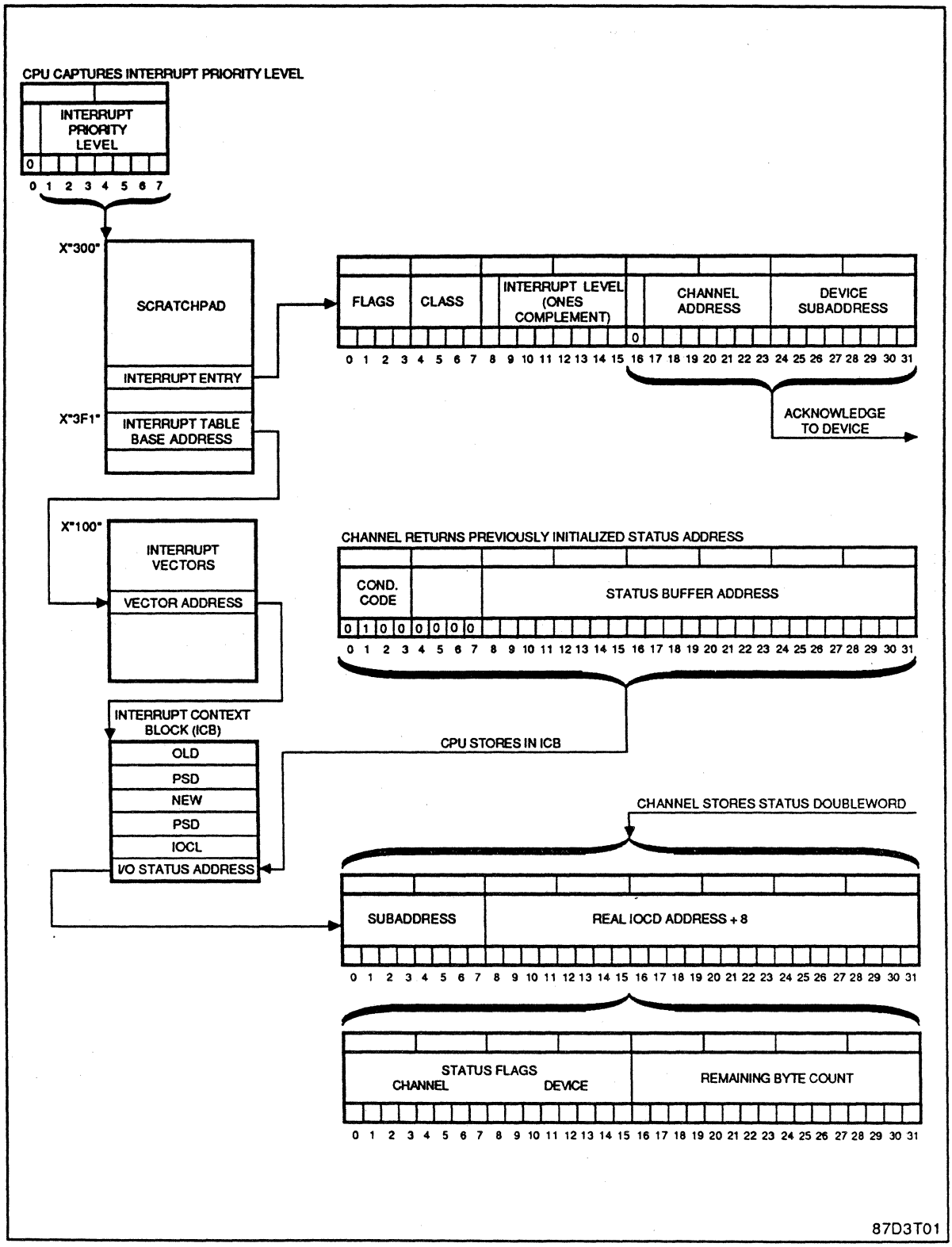



Figure 4-2. Termination of an I/O Operation

context block (ICB). Software has placed the starting address of the I/O command list (IOCL) into the fifth word of the ICB and the CPU fetches it. The CPU now initiates the I/O by passing the channel this IOCL address. The channel returns condition codes to the CPU which indicate acceptance or rejection of the SIO. The CPU then continues execution with the next instruction.

The MFP channel fetches and sends to the controller/device, the information from each IOCD that constitutes an IOCL. Details on the format and execution of this IOCL is described separately. When the channel and controller have finished processing the IOCL they will report status back to software through the interrupt mechanism described in section 4.2.2.

### 4.2.2 Termination of I/O

See figure 4-2, it shows the data structures required to successfully terminate an I/O operation and their relation to each other. The MFP terminates an operation by posting status in the reserved location dedicated by software. This can be accomplished through one of two methods; execution of the test I/O (TIO) instruction or through an interrupt.

#### 4.2.2.1 Termination Via Interrupts

Termination via interrupts is accomplished when the IOCL's execution is completed, the channel will poll on the SelBUS for an interrupt. This process is described later in this chapter. When the channel wins the interrupt polling process and the CPU has interrupts enabled, the CPU will capture the winning interrupt address.

This priority is added to the base interrupt address (x"300") of the CPU scratchpad to get the address of this particular devices interrupt entry. The CPU fetches this entry which contains the channel address and subaddress. The interrupt priority is also added to the base interrupt vector address (x"300") to fetch the address of this channels ICB 8set up by software). The CPU signals acknowledge back to the channel and waits for the channel to respond with a status buffer address.

The channel upon receipt of the acknowledge, stops the interrupt polling process and stores status into the reserved status location, set up previously by an initialize channel operation. Once status is stored, the channel responds to the CPU with the status buffer address. The CPU places this status address into the 6th word of the ICB and context switches software into the interrupt handler. Software may now examine status and take any required action before exiting the interrupt routine.

#### 4.2.2.2 Termination Via Test I/O

Termination via test I/O (TIO) occurs whenever the channel receives a TIO instruction from the CPU, it examines its internal status queue to determine if status is pending.

If there is status pending, the channel stores the next status doubleword in the reserved status buffer. Then it returns status stored condition codes and the status buffer address to the CPU. The CPU will store the status buffer address in the sixth word of the ICB, update condition codes in the PSD and continue execution of the instructions. Software should always check these condition codes (see Test I/O) after executing a test I/O instruction to determine whether status has been posted. Figure 4-2 is a depiction of the status returned at the termination of an operation.

If the channel has no status pending, it will return accepted and queued condition codes to the CPU and no status iiis posted.

## 4.3 The Interrupt Context Block (ICB)

The interrupt context block (ICB) contains information used both during I/O initiation and termination. See figure 4-3 for a depiction of the ICB.

The old PSD contains the current state of the interrupted task in case of an interrupt. The new PSD contains the new state to transfer into for interrupt handling. Note, bits 48 and 49 of the new PSD control the interrupt blocking capability of the interrupt handler. If the interrupt handler is to run with all interrupts blocked, then they are set to zero and one respectively. If the interrupt handler is to run with the interrupting channel going automatically active, then they both set to zero. See interrupt states for details.

The fifth word of the ICB contains the address of the IOCL to execute in case of a SIO instruction. This value must be initialized by software previous to the execution of the SIO instruction.

The sixth word of the ICB contains the address where the channel has posted status in case of an I/O completion. During an interrupt context switch, after storing status itself, the I/O channel will return this value to the CPU, who will store it into the ICB, before starting execution of the interrupt handler.

## 4.4 Supported I/O Instructions

The MFP supports 11 I/O instructions. Four of these instructions are dedicated to interrupt control and are described in section 4.9.2 The other seven deal with initializing, resetting, starting and stopping the I/O channel.

See figure 4-4. The exact function of an I/O instruction is determined by bits 9 through 12, FUNC (function code) field of the instruction. The addressed device is determined by adding the least significant halfword of the instruction to the least significant halfword of the selected register (R field). If the register field is zero, then the least significant halfword of the instruction is used to address the device alone. Note, for some instructions, the device field is not significant. For example, the reset channel instruction resets all devices on a given channel.

Condition codes will always be updated upon execution of "F" class I/O instructions. The condition codes are as follows:

| Code | Meaning | Notes |
|---|---|---|
| 0001 | Channel Busy | Channel initializing or internal process queue is full. |
| 0010 | Channel Inoperative or Undefined | Channel or not present or not functional or not defined in scratchpad. |
| 0011 | Subchannel Busy | Request denied because of previous action allocated MFP internal resources. |
| 0100 | Status Stored | Status was stored in response to last action (Interrupt, TIO). |
| 0101 | Unimplemented | |
| 1000 | Request Accepted and Queued | Normal - status will be queued for posting at end of operation. |

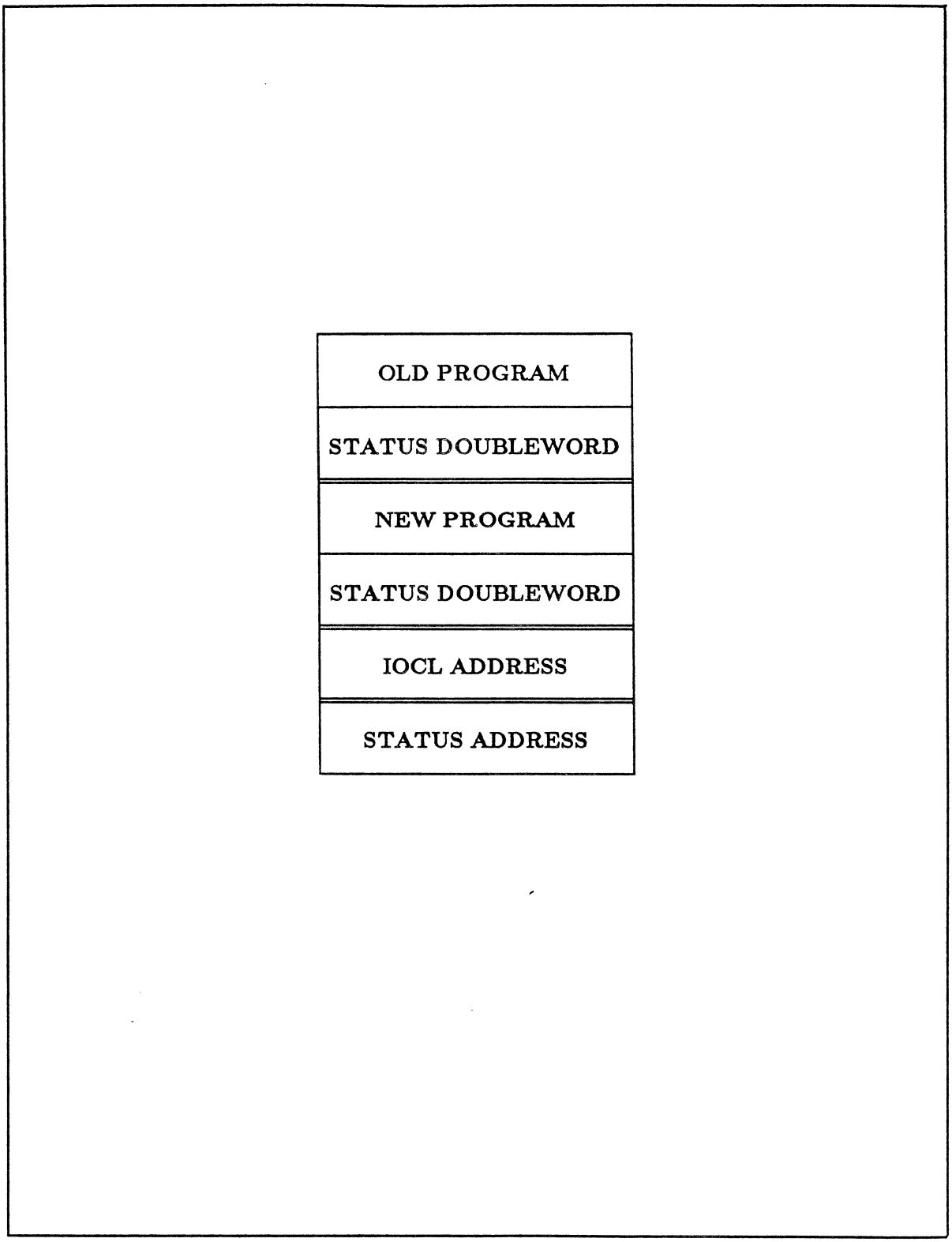
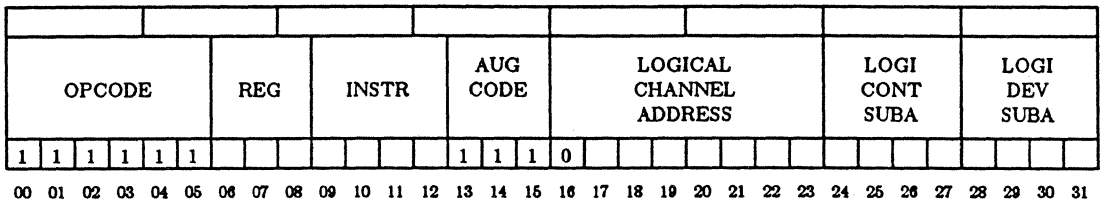



Figure 4-3. Interrupt Context Block (ICB) Format




| Bits | Description |
|---|---|
| 0-5 | OPCODE - specifies the operation code, hexadecimal FC. |
| 6-8 | REGISTER - when non-zero, specifies the general purpose register whose contents are added to constant to form the logical channel, controller, and device address. |
| 9-12 | INSTRUCTION - specifies the hexadecimal representation of the instruction to be performed. Table 4-1 provides a list of the class F I/O instructions recognized by the MFP. |
| 13-15 | AUGMENT CODE - specifies the augment code, hexadecimal 7. |
| 16-23 | LOGICAL CHANNEL ADDRESS - see constant. |
| 24-27 | LOGICAL CONTROLLER SUBADDRESS - see constant. |
| 28-31 | LOGICAL DEVICE SUBADDRESS - see constant. |
| 16-31 | CONSTANT - specifies a constant to add to the contents of the general purpose register specified by register to form the logical channel address, logical controller subaddress, and logical device subaddress. If register is zero, only the constant is used to form the logical address. |

Figure 4-4. Class F I/O Machine Language Format

Table 4-1. MFP Class F I/O Instructions

| Instruction (Bits 9-12) | Hex. Code | Mnemonic | Instruction Name |
|---|---|---|---|
| 0010 | 2 | SIO | Start I/O |
| 0011 | 3 | TIO | Test I/O |
| 0100 | 4 | STPIO | Stop I/O |
| 0101 | 5 | RSCHNL | Reset Channel |
| 0110 | 6 | HIO | Halt I/O |
| 0111 | 7 | GRIO | Clear Que/Grab I/O |
| 1000 | 8 | RSCTL | Reset Controller |
| 1100 | C | ECI | Enable Channel Interrupt |
| 1101 | D | DCI | Disable Channel Interrupt |
| 1110 | E | ACI | Activate Channel Interrupt |
| 1111 | F | DACI | Deactivate Channel Interrupt |

Listed with each I/O instruction is a similar list of possible condition codes returned in response to that particular instruction. Execution of "F" class instructions with non-supported function codes will result in no action taken and condition codes set to reflect unsupported transactions.

### 4.4.1 Start I/O

The start I/O (SIO) instruction causes actual I/O operations to take place depending upon the state of the I/O channel. Software must have previously initialized all of the data structures for the operation to execute successfully. The I/O command list describes the actual commands the I/O port is to execute. Condition codes returned after execution of this instruction are indicate whether the I/O operation was accepted by the MFP. If the operation was accepted by the MFP status will be returned to software at the completion of the command. The condition codes that may be returned in response to this instruction are as follows:

| Code | Meaning | Notes |
|---|---|---|
| 0001 | Channel Busy | Request denied, channel initializing or internal process queue is full. |
| 0010 | Channel Inoperative or Undefined | Channel or not present or not functional or not defined in scratchpad. |
| 0011 | Subchannel Busy | Request denied because SIO queue is full. |
| 0100 | Status Stored | Status was stored in response to last action (Interrupt, TIO). |
| 1000 | Request Accepted and Queued | Normal - status will be queued for posting at end of operation. |

### 4.4.2 Test I/O

The test I/O (TIO) instruction is used by software to determine the current state of the MFP channel. Condition codes returned after execution of this instruction indicate whether the MFP is operational and whether pending status has been posted in the reserved status buffer.

Execution of the TIO instruction is the only method other than through interruption that the MFP can deque pending status (although reset channel and SelBUS reset will clear any pending status). Status is not returned in response to a test I/O instruction itself, but the test I/O instruction will deque pending status from other operations (including asynchronous attention status from devices). The condition codes that may be returned in response to this instruction are as follows:

| Code | Meaning | Notes |
|---|---|---|
| 0010 | Channel Inoperative or Undefined | Channel or not present or not functional or not defined in scratchpad. |
| 0100 | Status Stored | Normal - channel operational and has posted status in response to TIO. |
| 1000 | Request Accepted and Queued | Normal - channel operational and no status pending. |

### 4.4.3 Stop I/O

The stop I/O (STPIO) instruction is used by software to terminate an I/O operation on a specific subaddress. If no operation is in progress for that subaddress, channel end/device end status is queued for posting without further action. If an operation is in progress, then it is terminated when the IOCD currently in progress is completed. After the operation is terminated, channel end/device end status is queued for posting.

Execution of stop I/O with multiple I/O operations queued for that subaddress will result in termination of all operations, if the clear que bit is set during subaddress initialization. The current operation will be terminated with channel end/device end status. Queued operations will be terminated with channel end/device end/unit exception status to indicate that they were not initiated. If the clear que bit is not set, or the subaddress has not been initialized, then only the current operation will be terminated. All following operations will be executed normally.

Note, only one termination operation, halt I/O, stop I/O, or clear que may be active for a given subaddress on the MFP at a given time. The condition codes that may be returned in response to this instruction are as follows:

| Code | Meaning | Notes |
|---|---|---|
| 0001 | Channel Busy | Request denied because channel internal process queue is full. |
| 0010 | Channel Inoperative or Undefined | Channel or not present or not functional or not defined in scratchpad. |
| 0011 | Subchannel Busy | Request denied because of outstanding HIO, STPIO, or clear que. |
| 1000 | Request Accepted | Normal - STPIO operation accepted by channel for execution. |

### 4.4.4 Reset Channel

Execution of the reset channel (RSCHNL) instruction causes all class "F" devices on the MFP channel to be reset into an uninitialized power up state. Real time functions are not affected and continue operation during the reset process. The class "F" channel will go into a busy state

during the initialization process. The device controllers may also remain busy during their initialization process even after the channel has become ready. Note, the channel will accept I/O instructions while the devices are still busy and que them for execution when the devices become free.

The first I/O instruction to the MFP channel following a reset channel instruction should be an interrupt control instruction. The CPU masks off the device address from the next I/O instruction to that channel after a reset channel and executing the interrupt instruction will clear this state, as well as automatically load the channel RAM with its interrupt priority.

The first SIO instruction following a channel or SelBUS reset, should include an initialize channel IOCD which will set up the channels internal status buffer pointer. If the MFP posts status before the channel initialize is executed, the status will be posted in locations 0 and 4 of main memory.

Note, reset channel is the only I/O instruction described in this section that does not result in status posted at completion of the operation. The condition codes that may be returned in response to this instruction are as follows:

| Code | Meaning | Notes |
|---|---|---|
| 0010 | Channel Inoperative or Undefined | Channel or not present or not functional or not defined in scratchpad. |
| 1000 | Request Accepted and Queued | Normal - reset channel operation accepted for immediate execution. |

### 4.4.5 Halt I/O

The halt I/O (HIO) instruction is used by software to immediately terminate an I/O operation on a specific subaddress. HIO causes a clearing action to take place on the device, regardless of whether there is an operation in progress or not. Furthermore, this clearing action takes place as soon as practicable and status is returned when the clearing action is complete.

Execution of HIO with multiple I/O operations queued for that subaddress will result in termination of all operations, if the clear que bit is set during subaddress initialization. The current operation will be terminated with channel end/device end/unit exception status to indicate that they were not initiated. All following operations will be executed normally.

Note, only one termination operation, HIO, STPIO, or clear que, may be active for a given subaddress on the MFP at a given time. The condition codes that may be returned in response to this instruction are as follows:

| Code | Meaning | Notes |
|---|---|---|
| 0001 | Channel Busy | Request denied because channel internal process queue is full. |
| 0010 | Channel Inoperative or Undefined | Channel or not present or not functional or not defined in scratchpad. |
| 0011 | Subchannel Busy | Request denied because of outstanding HIO, STPIO, or clear que. |
| 1000 | Request Accepted and Queued | Normal - HIO operation accepted by channel for execution. |

### 4.4.6 Clear Que

The clear que instruction is used by software to immediately terminate all outstanding I/O operations on a specific subaddress. This function is equivalent to HIO with the clear bit set during subaddress initialization. Clear que clears all outstanding I/O requardless of the state of the subaddress initialization mode bit.

Clear que instructions cause a clearing action to take place on the device, requardless of whether there is an operation in progress or not. Furthermore, this clearing action takes place as soon as practical and status returned when the clearing action is complete.

Execution of clear que with multiple I/O operations queued for that subaddress will result in termination of all operations. The current operation will be terminated with channel end/device end/unit exception status to indicate that they were not initiated.

Note, only one termination operation, HIO, STPIO, or clear que, may be active for a given subaddress on the MFP at a given time. The condition codes that may be returned in response to this instruction are as follows:

| Code | Meaning | Notes |
|---|---|---|
| 0001 | Channel Busy | Request denied because channel internal process queue is full. |
| 0010 | Channel Inoperative or Undefined | Channel or not present or not functional or not defined in scratchpad. |
| 0011 | Subchannel Busy | Request denied because of outstanding HIO, STPIO, or clear que. |
| 1000 | Request Accepted and Queued | Normal - clear que operation accepted by channel for execution. |

### 4.4.7 Reset Controller

The reset controller (RSCTL) instruction is used by software to immediately terminate all outstanding I/O operations on a group of subaddresses. Reset controller causes a clearing action to take place on all associated devices, regardless of whether there is an operation in progress or not. Furthermore, this clearing action takes place as soon as practicable and status returned when the clearing action is complete.

Depending upon the subaddress field of the instruction, the MFP will clear either a group of 8 or 16 subaddresses that constitute a controller. If the subaddress is less than x"80" then the device address is a SCSI controller consisting of 8 devices. The least significant 3 bits of the subaddress in the instruction are not significant and the subaddress in posted status will contain zeros in these bits regardless of the settings of these bits in the instruction. If the subaddress is x"80" or larger then the least significant 4 bits of the subaddress, they are masked out by the MFP channel and the associated 16 subaddresses are cleared.

Execution of reset controller with multiple I/O operations queued for that subaddress will result in termination of all operations. Channel end/device end status is posted only once, when all operations on all associated subaddresses are cleared.

Note, reset controller terminates any of the other outstanding termination operations; HIO, STPIO, or clear que. Only one reset controller operation will be accepted at a given time for a given subaddress group. The condition codes that may be returned in response to this instruction are as follows:

| Code | Meaning | Notes |
|---|---|---|
| 0001 | Channel Busy | Request denied because channel internal process queue is full. |
| 0010 | Channel Inoperative or Undefined | Channel or not present or not functional or not defined in scratchpad. |
| 0011 | Subchannel Busy | Request denied because of outstanding reset controller operation. |
| 1000 | Request Accepted and Queued | Normal - reset controller operation accepted by channel for execution. |

### 4.5 Execution of IOCLs

Start I/O instructions cause the MFP channel to fetch IOCDs from memory for execution. The fifth word of the ICB points to the first IOCD of an I/O command list (IOCL). The channel fetchs that IOCD and decodes the command byte. If the command is a channel control or a transfer-in-channel command, the channel executes the IOCD. Otherwise, the channel will pass the information on to the specified device for execution.

If the command is a transfer-in-channel, the channel will use the address in the data address field of the IOCD to fetch the next IOCD and will do so immediately. If the command is a channel control command, the channel will decode and execute the command appropriately. If the device is to execute the command, it decodes it, and when complete, it indicates completion to the channel.

When a command has been completed, the data and command chaining flags are checked to determine whether that was a final IOCD associated with the IOCL. If chaining is not specified, final status is queued for posting. If chaining is specified, then the channel will fetch the next IOCD from the next sequential location in memory.

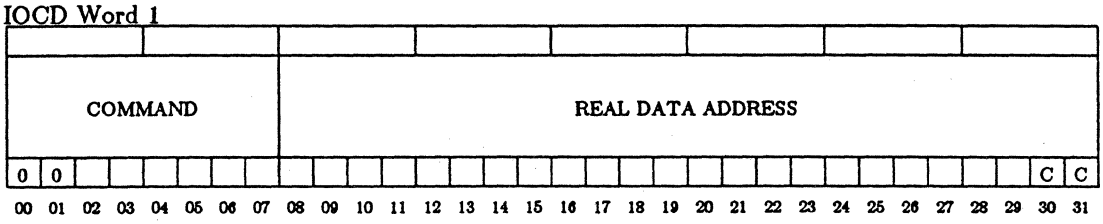
When errors in execution occur, chaining flags are ignored and the operation is stopped by queuing status for posting.

The MFP channel may prefetch the next IOCD (if any) in the IOCL and send the information on to the controller. This is done to enable the controller to execute the next operation without the delay of waiting for the completion of the first command. Software should normally build a complete IOCL before execution of the SIO instruction to prevent the channel from fetching IOCDs that are not yet built. This function is inhibited for the first IOCD of an IPL sequence.

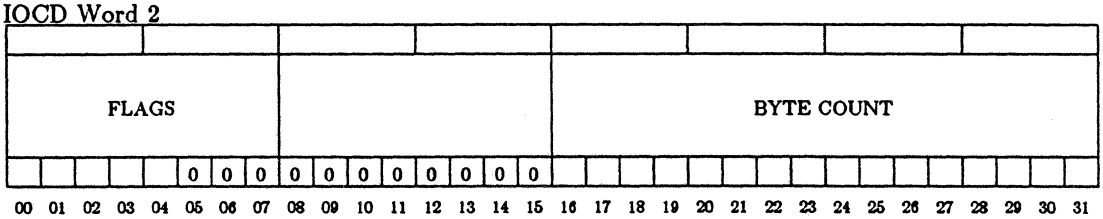
### 4.6 I/O Command Doubleword

The function of the I/O command doubleword (IOCD) is to convey the information required for execution of data input or data output between the CPU and the peripheral devices. The first byte is the command specifying what the operation is. The remaining three bytes of the first word is the address that specifies where the data is to be moved to/from. The first half of the second word contains control flags that indicate to the channel what to do after completion of the IOCD. The remaining halfword of the second word contains the number of bytes to be transferred. Note, the C-bits are ignored in the IOCD addresses; however, the IOCLA reported in the status reflects the C-bits of the initial IOCLA. See figure 4-5 for an illustration of the IOCD.

IOCD Word 1




IOCD Word 2




**Command Bit Assignments:**

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Hex Code | Command |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 00 | Channel Control |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 01 | Write |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 02 | Read |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 03 | Device Control |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 04 | Sense |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 08 | Transfer-In-Channel (TIC) |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0C | Reserved |

**Real Data Address:** see text

**C-Bit Assignments:**

| Bit 30 | Bit 31 | |
|---|---|---|
| 0 | 0 | Byte 0 |
| 0 | 1 | Byte 1 |
| 1 | 0 | Byte 2 |
| 1 | 1 | Byte 3 |

**Flag Bit Assignments:**

Bit 0 - Data Chain (DC)
Bit 1 - Command Chain (CC)
Bit 2 - Suppress Incorrect Length (SIL)
Bit 3 - Skip (SKIP)
Bit 4 - Post Program Controlled Interrupt (PPCI)
Bits 5-7 - Reserved (must be zero)

**Byte Count:** see text

Figure 4-5. Input/Output Command Doubleword (IOCD) Format

The IOCD's are processed through the utilization of a list. The list, an input/output command list (IOCL) is a list of IOCD's. Refer to the section, Execution of IOCLs, for a complete description. An IOCD must be placed on word boundaries; therefore, the channel automatically word bounds all IOCL addresses.

### 4.6.1 IOCD Commands Supported by the MFP

The IOCD commands supported by the MFP are described in the following paragraphs and listed as follows:

- Channel Control Command
- Write Command
- Read Command
- Device Control Command
- Sense Command
- Transfer-In-Channel Command

#### 4.6.1.1 Channel Control Command

The various channel control commands are executed directly by the MFP, not the devices. The channel control commands supported by the MFP are as follows:

- Initialize channel (INCH, command byte 00)
- Initialize Subaddress (command byte F0)
- Channel ID command (command byte 80)
- Command bytes 70 through 90 are reserved for future use.

**Initialize Channel Command**

The initialize channel (INCH) command sets the status buffer address for the MFP channel. The MFP takes the address provided in the first word of the IOCD and saves it internally as the status buffer address. The status buffer address specified should be on a word boundary. If it is not word bounded, the channel will automatically word bound it. If the status buffer address specified is zero, then program check/channel end/device end (PC/CE/DE) status is queued for posting. This command should be issued after a channel or SelBUS reset, or status will be stored in location zero by default. The byte count in an INCH command is ignored.

**Initialize Subaddress Command**

The initialize subaddress command initializes several channel operation parameters for a given subaddress. One data byte is transferred containing the following indicators:

| | Bit | Meaning |
|---|---|---|
| (MSB) | 0 | Clear SIO que on error when 1. When this bit is 1, queued SIOs are cleared from the SIO que when a unit check is detected on a given port. An interrupt and status with the error indication is returned for each SIO in the que. If this bit is 0, queued SIOs are issued as if there was no error. |

| | | |
|---|---|---|
| | 1 | Reserved for future use. |
| | 2 | When this bit is set to 0, it changes a read logical block size of 256 to simulate a logical block size of 768. When this bit is set to 1, the logical block size is maintained. |
| (LSB) | 3-7 | Reserved for future use. |

**Channel ID Command**

The channel identification (Channel ID) command returns three words of data that identifies the: board model number (MMXXXXXX), firmware model number (MMXXXXXX), and firmware revision number (MMXXXXXX). Where X equals the model number and M equals the state flags which are normally zero.

#### 4.6.1.2 Write Command

The write command is executed by device controllers and is used to transmit data from a software specified buffer to a device or media. For example, write a sector of data to a disk. The channel does not decode the modifier bits in this command, it simply passes them to the device controller. The channel does check for a non-zero byte count in a write IOCD. If the byte count is zero, the IOCL is aborted and PC/CE/DE status is queued for posting. See the specific device manual for information on the use of this command.

#### 4.6.1.3 Read Command

The read command is executed by the device controller, it is used to transfer data from a device to a software specified buffer resident to the computer system. For example, read a block of data from a magnetic tape or disk. The channel does not decode the modifier bits in this command, it simply passes them through to the device controller. The channel does check for a non-zero byte count in a read IOCD. If the byte count is zero, the IOCL is aborted and PC/CE/BE status is queued for posting. See the specific device manual for information on the use of this command.

#### 4.6.1.4 Device Control Command

The device control command is executed by the device controller, it is used to configure a device or cause it to perform a specific action. For example, execute a seek on a disk drive, or, reset the baud rate in a communications controller. The channel does not decode the modifier bits in this command, it simply passes them through to the device controller. A zero byte count in a device control IOCD is valid, the control information can be passed on via the modifier bits. Note, NOP commands must specify a byte count of zero. See the specific device manual for information on the use of this command.

#### 4.6.1.5 Sense Command

The sense command is executed by the device controllers, it is used to return specific status to software about the device. For example, the condition of modem lines on an asynchronous port. The channel does not decode the modifier bits in this command, it simply passes them through to the device controller. The channel does check for a non-zero byte count in a sense IOCD; if a zero byte count is detected, the IOCL is aborted and PC/CE/DE status is queued for posting.

See the specific device manual for information on the use of this command.

#### 4.6.1.6 Transfer-In-Channel Command

When the MFP receives a Transfer-in-Channel (TIC) command, it accepts it as an unconditional branch to an IOCD within the same IOCL or to an IOCD within another IOCL. The TIC command may not be the first IOCD in an IOCL, nor may its real data address field contain a pointer to another IOCD that proves to be either another TIC command or a non-existent IOCD. If an IOCL ends with a TIC command, at least one other IOCD in the TIC loop must contain the PPCI flag, since the TIC command does not generate a service interrupt request.

### 4.6.2 IOCD Real Data Address Fields

Real data address field - (IOCD word 1, bits 8 through 31) specifies the address from which data is to be moved from or moved to. This address is a physical (unmapped) address.

The least significant two bits of the real data address are called the C-bits. The C-bits specify from which byte of the word the transfer will start from. If these bits are both zero, the transfer starts at the most significant byte of the word. If both C-bits are one, the transfer starts at the least significant byte of the word. If the least significant C-bit is one and the most significant C-bit is zero, the least significant byte of the most significant halfword is transferred first. If the least significant C-bit is zero and the most significant C-bit is one, the most significant byte of the least significant halfword is transferred first. Note, transfers always start at the most significant byte and progress to the least significant byte.

### 4.6.3 IOCD Flag Fields

Flags field - (IOCD word 2, bits 0 through 7) specifies control flags used in the processing of IOCD's. The definition of the flags are described in the following paragraphs.

#### Data chaining (DC) - bit 0

When this bit is set, it indicates to the MFP channel that another IOCD is in the following two words in memory.

The following IOCD assumes command over the current IOCD (the one with data chaining set). Data chained IOCDs are executed as if they were one, except for the changing of the transfer addresses and counts between them. Data chained reads and writes will only read or write one record.

Multiple IOCDs that are data chained together, all assume the command of the first IOCD with the data chain bit set.

#### Command chaining (CC) - bit 1

When this bit is set, it indicates to the MFP channel that another IOCD is in the following two words in memory.

The following IOCD is executed upon completion of the first IOCD. Command chained IOCDs each use their own command byte as the command to be executed. Command chained IOCDs will each read or write separate records. For example: a read IOCD partially reads a record and is command chained to another read. The following read will not read the rest of the record, but will start to read the next record on the media.

**Suppress incorrect length (SIL) - bit 2**

When this bit is set, it prevents the MFP channel from terminating IOCLs when an incorrect length indication occurs. The channel normally terminates the IOCL (breaks chaining) when the transfer count does not match the amount of data available on the media. Setting this flag causes the error condition (incorrect length) to be suppressed. Note, the remaining transfer count in status is not affected. This flag does not override the breaking of a data chained IOCL when a incorrect length indication is encountered.

**Skip (SKIP) - bit 3**

When this bit is set, it inhibits the actual transfer of data to memory. The operation is actually executed, except for the transfer of data to memory. The common use of the skip is in conjunction with data chain. The first IOCD of a pair has skip and data chin set. Execution of this IOCD will position the media for the following IOCD which will actually transfer data from within a record.

**Post program controlled interrupt (PPCI) - bit 4**

The post program controlled interrupt bit when set, causes the MFP channel to que status for posting upon completion of the associated IOCD.

The status posted contains a zero byte count and only the program interrupt flag in word two. The first word contains the device subaddress and the associated IOCDs address plus eight. Note, following IOCDs may be completed before software receives this PPCI status because of possible delays in interrupts being handled.

The exact time of the PPCI is indeterminate, being dependent on the priority level assigned to the MFP, the number and priority level of currently active interrupts, and whether the CPU has blocked the sensing of interrupts.

### 4.6.4 IOCD Byte Count Field

Byte count field - (IOCD word 2, bits 16 through 31) specifies the number of bytes that software has requested for the transfer of this IOCD. The count is in bytes. Read, write, and sense IOCDs must have a transfer count of at least one or the program check status will be returned and the IOCL terminated. The maximum transfer count is 65535 (x"FFFF"), but larger transfers may be executed through the use of data chaining two or more IOCDs together.

## 4.7 I/O Status Doubleword

The status doubleword is maintained by the MFP. When the device and channel complete an IOCL, or it is terminated because of an unusual condition, status is queued for posting to software. This status is returned to software through an interrupt or a test I/O (TIO)

instruction. After the interrupt occurs, or the TIO instruction is executed, the status pointer of the channels ICB points to where status was written in memory.

The status doubleword is illustrated in Figure 4-6 and its fields are explained as follows:

1. Subaddress - (word 1, bits 0 through 7) specifies the address of the associated I/O status.

2. Real IOCD address - (word 1, bits 8 through 31) specifies the real address of the next IOCD in the IOCL (address of the last IOCD executed, plus eight bytes).

3. Status flags - (word 2, bits 0 through 15) specifies the channel, controller, and device status resulting from the last IOCD executed. Individual status flags are described in Status Flag Bits, below.

4. Remaining byte count - (word 2, bits 16 through 31) specifies the residual byte count of the last IOCD executed.

### 4.7.1 Subaddress

The subaddress field of status contains the device subaddress for which the I/O status is associated. In some cases, the status actually is associated with more than one device. Examples of these operations are reset controller and initialize channel. The software interrupt handler will normally use the subaddress field of status to locate a unique data structure for this subaddress. Status can then be relocated into this data structure and the interrupted receiver exited.

### 4.7.2 IOCD Address

The real IOCD address field of status contains the address in system memory of the last IOCD executed plus eight bytes. If no IOCDs are executed due to an error condition, the address points to the first IOCD in the list. This field can be used by software, if an error occurs, to help determine the IOCD that caused the error.

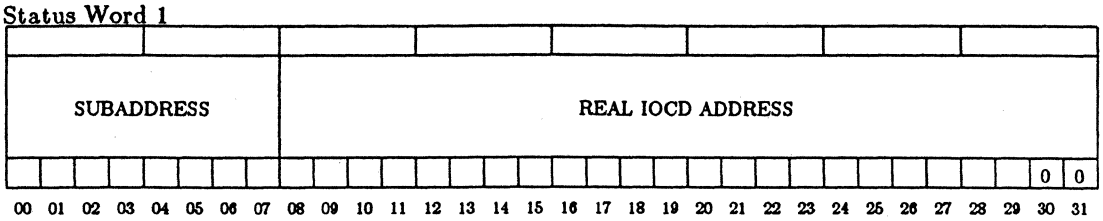
### 4.7.3 Status Flag Bits

The status flag bits of the status doubleword specify the channel, controller, and device status of the subchannel addressed by the last IOCD executed. Note that the status doubleword may not reflect the current status of the subchannel addressed by an SIO instruction that was rejected due to the reasons previously described. See Interrupt Polling Restrictions and Considerations. The meaning of each flag bit that is used by the MFP is described in the following paragraphs.

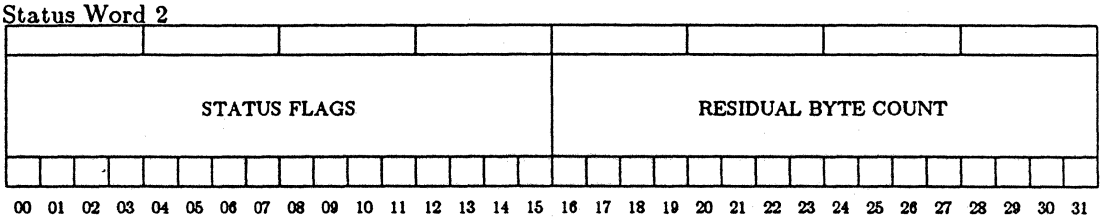
#### 4.7.3.1 Post Program Controlled Interrupt

The post program controlled interrupt (PPCI) is generated when the PPCI flag is set in the IOCD and, either the byte count is reduced to zero (if data chaining is specified) or device end is signaled (if command chaining is specified). To avoid confusion as to the reason for an interrupt, this bit should not be used in the last IOCD of a chained IOCL.

Status Word 1




Status Word 2




**The fields have the following meaning:**

**Subaddress (word 1, bits 0-7):** Specifies the subchannel address.

**Real IOCD Address (word 1, bits 8-31):** Specifies the address of the next IOCD.

**Status Flags (word 2, bits 0-15):** Identifies the conditions in the subchannel that caused the status doubleword to be stored. Each of the 16 bits represent one type of condition as follows:

| Bit | Condition | Remarks |
|---|---|---|
| 0 | Reserved | |
| 1 | PPCI | PPCI bit set in last IOCD |
| 2 | Incorrect Length | Difference in record size & IOCD byte count |
| 3 | Channel Program Check | Logical error in IOCL |
| 4 | Channel Data Check | Parity error in host memory |
| 5 | Reserved | |
| 6 | Channel Interface Check | Error on interface to controller/device |
| 7 | Reserved | |
| 8 | Busy | SIO to an active subchannel/device |
| 9 | Status Modifier | Modifies other status bits |
| 10 | Controller End | Controller is free |
| 11 | Attention | Device interrupt (asynchronous to commanded I/O) |
| 12 | Channel End | End of current IOCL |
| 13 | Device End | End of current IOCL |
| 14 | Unit Check | Error detected at device |
| 15 | Unit Exception | Unusual condition at device |

**Residual Byte Count (word 2, bits 16-31):** Specifies the residual byte count of the last IOCD executed.

**Figure 4-6. Status Doubleword Format**

#### 4.7.3.2 Incorrect Length

Incorrect length (IL) occurs when the byte count specified in the IOCD is not equal to the number of bytes transferred to or from the device. Incorrect length is indicated for any of the following conditions:

1. Long Block on Input - During an input (read) operation, the device requested one or more bytes after the channel had reduced a non-data chained IOCD byte count to zero. The extra bytes are not written to host memory. The residual count of the status will be set to zero.

2. Long Block on Output - During an output (write) operation, the device requested one or more bytes after the channel has reduced a non-data chained IOCD byte count to zero. The extra bytes are not written to the device. The residual count of the status will be zero. Note, on fixed block size devices (disk) any remaining portion of an unwritten block is filled with zeros.

3. Short Block on Input - The number of bytes transferred during an input (read) operation were insufficient to reduce the IOCD byte count to zero. The count in the status will remain nonzero and indicate the number of bytes not transferred.

4. Short Block on Output - The device terminated an output (write) operation before all information was transferred. The count in status will indicate the number of bytes not transferred.

#### 4.7.3.3 Channel Program Check

The channel program check is set by the MFP when it has detected a programming error in the IOCL. The error can be due to any of the following conditions:

1. Invalid IOCD address specification - This condition indicates that the real data address contained in the IOCD is not on a word boundary. In the case of the MFP, when a channel program check error occurs, the IOCL is aborted; if the IOCD was fetched successfully, the byte count will be valid, otherwise it will be zero.

2. Invalid IOCD address - The MFP attempted to read an IOCD from non-present memory.

3. Invalid command - The command specified in the IOCD is not recognized by the MFP.

4. Invalid count - The IOCD contains a byte count which is outside the limits for the command.

5. Invalid data address - The MFP attempted to read or write a byte from non-present memory.

6. Invalid sequence - A TIC command either specifies another TIC command or is the first IOCD in the IOCL.

#### 4.7.3.4 Channel Data Check

The channel data check indicates that the MFP detected a parity error in the data transferred from host memory.

This condition causes the transfer to be aborted and command chaining to be suppressed. An interrupt is generated when the MFP presents device end/channel end.

#### 4.7.3.5 Interface Check

Interface check is an indication of a communications malfunction between the channel and one of its controllers. Causes may include parity errors in the data transmitted to the channel from the controller or invalid event sequences during channel/controller communications. This condition indicates that the MFP could not communicate with the addressed device, or the controller is broken, or a bad cable has been used, or a jumper has been inserted in the wrong position. If received, chaining is suppressed, the remaining byte count is zero, and the operation is terminated.

#### 4.7.3.6 Busy

Busy indicates that the IOCL was terminated because of a busy condition in either the device or its controller. Busy with only channel end or device end indicates that the busy condition was in the device, not the controller.

When the busy condition is indicated in conjunction with channel end, device end, and status modifier, it indicates that the busy condition was in the controller, not the device.

When the busy condition is indicated in conjunction with channel end, device end, and unit exception, it indicates that the busy condition was do to a SCSI device that was previously reserved by another initiator.

#### 4.7.3.7 Status Modifier

The status modifier indicates that the controller cannot provide current status because the controller is busy, or that the normal sequence of IOCDs require modification.

When the status modifier condition is indicated in response to an I/O instruction and no other status bits are indicated, it indicates that the controller cannot execute the instruction and has not provided current status. Any pending interrupt conditions present in the controller or its device are not cleared. The status stored will contain all zeros except for the status modifier bit.

When the status modifier bit appears with the busy bit, it indicates that the condition pertains to the controller end and the addressed device. The controller appears busy when executing operations which preclude the acceptance and execution of any command or I/O instruction, or contain a pending interrupt and status for a device other than the device addressed.

#### 4.7.3.8 Controller End

Controller end indicates that the controller has become available for use in another operation, after previously indicating busy.

#### 4.7.3.9 Attention

When the attention status flag bit is set, it indicates that the device has detected an asynchronous condition that is device dependent. The device can generate attention only when no operation is in progress at the subchannel, controller, and device. Attention can be generated by itself, or, it can be accompanied by device end upon the completion of an operation.

#### 4.7.3.10 Channel End/Device End

The channel end/device end status flags indicate the completion of an I/O operation by the device and channel. These status bits always occur together for the MFP channel. Note, although the MFP has indicated channel end/device end, in some cases, the device may still be working on the operation. For example, with the immediate bit set in the command, when a magnetic tape unit rewinds a tape, channel end/device end will be indicated although the tape is still in motion.

#### 4.7.3.11 Unit Check

Unit check indicates that the controller or device has detected an unusual condition that is detailed by the information available by executing a sense command. When a unit check occurs, the IOCL is terminated, as well as, command chaining. Also, the IOCD address is added to eight, this action reflects the next IOCD that was to be executed and the byte count is updated.

#### 4.7.3.12 Unit Exception

Unit exception indicates the occurrence of an unusual condition at the device which prevents completion of the command. The IOCL is terminated when a unit exception status condition occurs. The IOCD address plus eight reflects the next IOCD that was to be executed and the remaining byte count is updated.

When unit exception occurs and clear que is set and multiple IOCLs are queued for execution, the entire queue is purged. The first status returned indicates the original cause for termination of the operation. Following IOCLs are aborted with unit exception status set, indicating that they were not executed because of the previous error.

### 4.7.4 Remaining Byte Count

The remaining byte count contains the number of bytes remaining to be transferred from the last IOCD under execution before termination. Normal termination results in the remaining byte count to be set to zero. In some status blocks, such as attention status, the remaining byte count field is not significant. Normally, unless suppress incorrect length is set, the incorrect length flag is set whenever an operation terminates because of differences between the transfer count and the record size.

## 4.8 CPU Scratch Pad and Interrupt Vectors

The CPU relies on software for the initialization of the CPU scratch pad and interrupt vector table so that it can execute I/O instructions (the sole exception is IPL).

### 4.8.1 CPU Scratch Pad

The CPU scratch pad is memory; however, it is memory in a special location within the CPU that contains the system configuration information. The operating system must initialize the CPU scratch pad at the time of IPL with the TRSC instruction. The operating system keeps a copy of the CPU scratch pad in main memory, starting at an address location that is 300 greater than the actual CPU scratch pad address. For example, if the actual CPU scratch pad address is 0, add 300 to zero to obtain the address of the copy in main memory; the copy in main memory is at address 300. For I/O to function, two sets of entries must be initialized, Device Entries and Interrupt Entries.

#### 4.8.1.1 Device Entry

The device entries are used to map software device addresses into SelBUS physical addresses. The device entry associates the I/O protocol class with the channel interrupt priority level, the channel physical address, and the device subaddress. Device entries are stored in the CPU scratchpad locations 00 through 7F, according to the software channel address field of the entry. See figure 4-7 for the Device Entry, CPU scratch pad word format.

#### 4.8.1.2 Interrupt Entry

The interrupt entries are use to map software interrupt levels into SelBUS physical addresses. The interrupt entry associates the I/O protocol class with the RTOM interrupt or I/O interrupt identity, the interrupt priority level, the channel address, and the device subaddress. Interrupt entries are stored in the CPU scratchpad at addresses 80 through EF. The interrupt level plus 80, provides an index to the interrupt entries. See figure 4-8 for the Interrupt Entry, CPU scratch pad word format.

### 4.8.2 Interrupt Vector Table

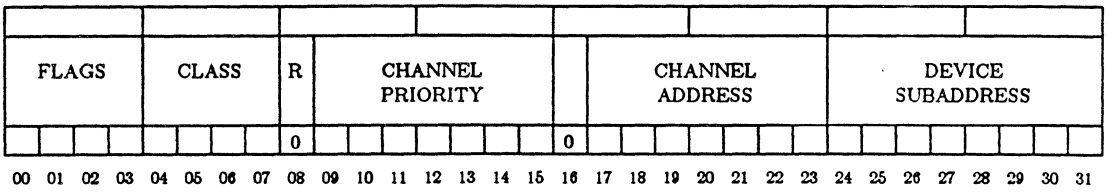
The interrupt vector table contains one entry for each interrupt priority level available. The interrupt vector table starts at location 100, assuming a concept 67 CPU. The interrupt vector tables address is located in the CPU scratch pad. See table 4-2 for a list of the interrupt vectors.

## 4.9 Interrupts

Two events must take place successfully, before the MFP can request an interrupt from the CPU. First, the channel must be placed in an enabled and requesting state (but not an active state). Second, the interrupt flags of the PSD must be set to enable interrupts. Once these conditions are met, the MFP can actively pursue interrupts. The following sections describe the various states of interrupt logic for the MFP and the CPU instructions necessary to control them.

### 4.9.1 Interrupt States

There are eight interrupt states supported by the MFP. In addition to the supported interrupt states, software is able to inhibit or allow interrupts through the use of the control flags available through the second word of the PSD. The interrupt states are described in the following section and are as follows:




**FLAGS (bits 0-3): (controlled by CPU)**

Channel RAM Load with Interrupt Priority (bit 0)
Bits 1 through 3 are not used.

**CLASS (bits 4-7):**

I/O Protocol Class (bit 4)
Class F - MFP Channel (bit 5)
Class 3 - Interval Timer (bit 6)

**R (bit 8): always zero**

**CHANNEL PRIORITY (bits 9-15):**

Ones complement of the service interrupt priority level. Every channel in the system capable of interrupting at a unique priority must have its own unique interrupt level. See scratch pad interrupt entries section for more details.

**CHANNEL ADDRESS (bits 17-23):**

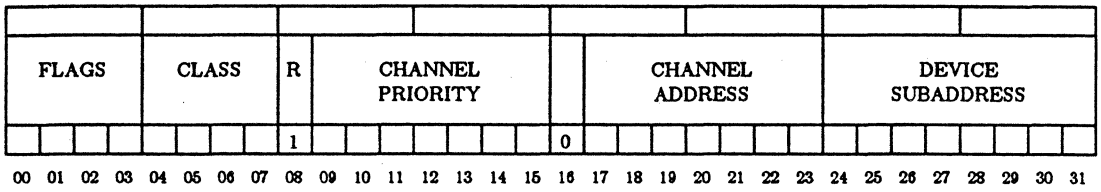
The channel address contains the physical channel address as jumpered on the circuit card.

**DEVICE SUBADDRESS (bits 24-31):**

The MFP channel requires two entries in the device portion of the CPU scratch pad. Entry for class F = 00. Entry for the interval timer = 04. For the real time function subaddresses, only the interval timer has an entry in the device portion of the CPU scratch pad.

**NOTE: bit 16 always equals zero.**

Figure 4-7. Device Entry, CPU Scratch Pad Word Format




**FLAGS (bits 0-3): (controlled by CPU)**

Channel RAM Load with Interrupt Priority (bit 0)
Bit 1 not used
Interrupt Active (bit 2)
Interupt Enabled (bit 3)

**CLASS (bits 4-7):**

I/O Protocol Class (bit 4)
Class F - MFP Channel (bit 5)
Class 3 - Real Time Functions (bit 6)

**R (bit 8): Set to 1 for RTF interrupt levels.**

**CHANNEL PRIORITY (bits 9-15):**

Ones complement of the service interrupt priority level. Every channel in the system capable of interrupting at a unique priority must have its own unique interrupt level. See scratch pad interrupt entries section for more details.

**CHANNEL ADDRESS (bits 17-23):**

The channel address contains the physical channel address as jumpered on the circuit card.

**DEVICE SUBADDRESS (bits 24-31):**

The MFP channel requires 17 entries in the interrupt portion of the CPU scratch pad. Entry for class F = zero and the remaining 16 entries are for the subaddresses involved in real time functions.

**NOTE: bit 16 always equals zero.**

Figure 4-8. Interrupt Entry, CPU Scratch Pad Word Format

**Table 4-2. Interrupt Vectors**

| Priority | Default Interrupt Vector Location | Description |
|---|---|---|
| 00 | 100 | External/software interrupt 0 |
| 01 | 104 | External/software interrupt 1 |
| 02 | 108 | External/software interrupt 2 |
| 03 | 10C | External/software interrupt 3 |
| 04 | 110 | I/O channel 0 interrupt |
| 05 | 114 | I/O channel 1 interrupt |
| 06 | 118 | I/O channel 2 interrupt |
| 07 | 11C | I/O channel 3 interrupt |
| 08 | 120 | I/O channel 4 interrupt |
| 09 | 124 | I/O channel 5 interrupt |
| 0A | 128 | I/O channel 6 interrupt |
| 0B | 12C | I/O channel 7 interrupt |
| 0C | 130 | I/O channel 8 interrupt |
| 0D | 134 | I/O channel 9 interrupt |
| 0E | 138 | I/O channel A interrupt |
| 0F | 13C | I/O channel B interrupt |
| 10 | 140 | I/O channel C interrupt |
| 11 | 144 | I/O channel D interrupt |
| 12 | 148 | I/O channel E interrupt |
| 13 | 14C | I/O channel F interrupt |
| 14 | 150 | External/software interrupt |
| 15 | 154 | External/software interrupt |
| 16 | 158 | External/software interrupt |
| 17 | 15C | External/software interrupt |
| 18 | 160 | Real-time clock interrupt |
| 19 | 164 | External/software interrupt |
| " | " | " |
| " | " | " |
| " | " | " |
| 6E | 2B8 | External/software interrupt |
| 6F | 2BC | Interval timer interrupt |

NOTES:
1) The interval should always have the highest priority number assigned.
2) The highest interrupt level of a Concept 67 is 6F, which is the lowest priority.
3) Class F I/O may occupy any interrupt level.
4) Class E or D I/O is not supported on the MFP.

- state 0 - not active/no request/disabled
- state 1 - not active/request queued/disabled
- state 2 - active/no request/disabled
- state 3 - active/request queued/disabled
- state 4 - not active/no request/enabled
- state 5 - not active/requesting/enabled
- state 6 - active/no request/enabled
- state 7 - active/request queued/enabled

In conjunction with the interrupt states, the MFP also supports the active flag, enabled flag, and requesting flag. Each of the flags are under software control.

Table 4-3 shows the interrupt states for the MFP, as well as the possible events that can change the current state of the channel. The first four events (enable, disable, activate, and deactivate) occur as transfers from the CPU as a result of the execution of an instruction or interrupt context switch. The request event occurs upon the completion of an I/O operation, or because of an asynchronous status from a controller. The acknowledge event occurs as a result of an interrupt context switch taking place.

#### 4.9.1.1 Interrupt States, Described

**State 0 - Not Active/No Request/Disabled**

This is the idle state of the channels interrupt logic. The channel is not taking part in the SelBUS interrupt priority poll and has no status to post. Furthermore, the channel is disabled from posting status through the interrupt mechanism should an I/O operation be completed (see State 1).

**State 1 - Not Active/Request Queued/Disabled**

This is another idle state of the channels interrupt logic. The channel is not taking part in the SelBUS interrupt priority poll; however, the channel does have status to post. The MFP cannot post status through the interrupt mechanism because this channels interrupt level is disabled.

**State 2 - Active/No Request/Disabled**

In this state, the channel is continuously polling on the SelBUS interrupt bus. This prevents lower priority channels from interrupting because they are blocked by the poll of the MFP; however, this channel will not interrupt software, and it has no status to post. Channels with a higher priority than the MFP will win the interrupt poll from the MFP, and they may interrupt software. Since the MFP is active and disabled, it prevents it from interrupting to post status should an I/O operation be completed (see State 3).

Table 4-3. Interrupt States

| | Current State | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Disabled | | | | Enabled | | | |
| | Not Active, No Request<br>State 0 | Not Active, Request Queued<br>State 1 | Active, No Request<br>State 2 | Active, Request Queued<br>State 3 | Not Active, No Request<br>State 4 | Not Active, Requesting<br>State 5 | Active, No Request<br>State 6 | Active, Request Queued<br>State 7 |
| Event: | | | | | | | | |
| Enable | 4 | 5 | 6 | 7 | - | - | - | - |
| Disable | - | 0 | - | 2 | 0 | 0 | 2 | 2 |
| Activate | 2 | 3 | - | - | 6 | 7 | - | - |
| Deactivate | - | - | 0 | 1 | - | - | 4 | 5 |
| Request | 1 | - | 3 | - | 5 | - | 7 | - |
| Acknowledge | X | X | X | X | X | 6 | X | X |

**Key to matrix:**

**A hyphen (-) equals no change in the current state.**
**A number in the matrix equals new state after execution of the event.**
**X equals invalid operation.**

### State 3 - Active/Request Queued/Disabled

In this state, the channel is continuously polling on the SelBUS interrupt bus. It prevents lower priority channels from interrupting because they are blocked by the poll of the MFP; however, while it is active, the MFP will not interrupt software even though it has status to post. Higher priority channels will win the interrupt poll from the MFP and they may interrupt software. Since the MFP is active and disabled, it is prevented from interrupting to post status; however, status may be posted through the test I/O mechanism at any time.

### State 4 - Not Active/No Request/Enabled

This is another idle state for the MFPs interrupt logic. It is not taking part in the SelBUS interrupt priority poll and it has no status to post. It is, however, enabled to post status through the interrupt mechanism should an I/O operation be completed (see State 5).

### State 5 - Not Active/Requesting/Enabled

This is the ONLY state in which the MFP is able to interrupt software. The MFP will actively poll the SelBUS interrupt bus; when it wins, it will interrupt the software, assuming that interrupts are not blocked. If software disables or activates the MFP interrupt level, this state is left and the interrupt will not occur.

### State 6 - Activate/No Request/Enabled

In this state, the MFP continuously polls the SelBUS interrupt bus. It prevents lower priority channels from interrupting because they lose the poll. However, the MFP will not interrupt software and has no status to post. Channels with a higher priority may win the interrupt poll from the MFP and they may interrupt software. Since the MFP is active, it prevents it from interrupting software to post status should an I/O operation be completed (see State 7).

### State 7 - Active/Request Queued/Enabled

In this state, the MFP continuously polls the SelBUS interrupt bus. It prevents lower priority channels from interrupting because they lose the poll. However, the MFP will not interrupt software even though it has status to post. Channels with a higher priority will win the interrupt poll from the MFP if they are polling and they may interrupt software. Since the MFP is active, it prevents it from interrupting software to post status should an I/O operation be completed. Status may be posted through the test I/O mechanism at any time.

### 4.9.1.2 Interrupt State Control Flags

The control flags for the MFP interrupt states are the active flag, enabled flag, and the requesting flag. These flags are under software control. The description of the control flags are as follows:

#### Active Flag

The active flag is set when the MFP is in the active state. The active state is controlled by the software instructions, activate channel interrupt and deactivate channel interrupt. The activate state will also be set when an interrupt context switch takes place with the new PSD specifying unblocked operation. After powering up the system and after a channel reset or SelBUS reset is performed, the MFP is set in a NOT active state.

#### Enabled Flag

The enabled flag is set when the MFP is in the enabled state. The enabled state is controlled by the software instructions, enable channel interrupt and disable channel interrupt. After powering up the system and after a channel reset or SelBUS reset is performed, the MFP is set in a disabled state.

#### Requesting Flag

The requesting flag is set when the MFP is in the request pending state. The request pending state is set by the completion of I/O operations. After powering up the system and after a channel reset or SelBUS reset is performed, the MFP is placed in a no interrupt request pending state. Whenever an I/O operation (except reset channel) is completed, the MFP goes in a request pending state. Note, multiple I/O completions will be queued, awaiting status posting for the previous completion. Thus, when the requesting flag is set, it indicates that at least one I/O completion status is pending, NOT how many. I/O completions may occur because of operations initiated by software, or, they may occur because controllers are returning asynchronous status messages.

#### 4.9.1.3 Interrupt State Events

There are six events associated with interrupt states, they are as follows:

**Enable Event**

The enable event only occurs during execution of an enable channel interrupt instruction. The enable channel interrupt instruction is automatically generated by the CPU during an IPL sequence.

**Disable Event**

The disable event only occurs during execution of a disable channel interrupt instruction. When this instruction is executed, it removes (clears) one pending interrupt from the interrupt que.

**Activate Event**

The activate event occurs either during an activate channel interrupt or an interrupt context switch with the blocking control bits turned off.

**Deactivate Event**

The deactivate event occurs either during execution of the deactivate channel interrupt.

**Request Event**

The request event occurs as a result of I/O completion, or, because of a device reporting an asynchronous status.

**Acknowledge Event**

The acknowledge event always occurs during an interrupt context switch of software.

### 4.9.2 Interrupt Instructions

The interrupt instructions control the interrupt mechanism through variations of the generic class F I/O instructions as listed below:

- Enable Channel Interrupt
- Disable Channel Interrupt
- Activate Channel Interrupt
- Deactivate Channel Interrupt

The format of the instructions was illustrated in figure 4-4, located at the start of this chapter. In addition to these interrupt instructions, the block external interrupt (BEI) and unblock external interrupt (UEI) instructions control whether or not any interrupts are allowed.

#### 4.9.2.1 Enable Channel Interrupt

The enable channel interrupt allows a channel to request an interrupt. When a channel is permitted to interrupt, it does so by interrupting software. Execution of enable channel interrupt returns conditions codes of all zeros to be returned to software if the channel is present and functioning.

#### 4.9.2.2 Disable Channel Interrupt

The disable channel interrupt disables the MFP from interrupting software. When this instruction is executed, it clears the que of one pending request from the request que, and conditions codes of all zeros are returned to software. If desired, the channel can be placed in an active state while disabled, this would prevent channels with lower priorities from interrupting.

#### 4.9.2.3 Activate Channel Interrupt

The activate channel interrupt sets the MFP channel in a state of actively contending for priority on the SelBUS interrupt bus. It does not cause software interrupts even when it wins the right to poll or it has a pending request; however, an activate channel will prevent channels with lower priorities from interrupting software. When this instruction is executed, it returns condition codes of all zeros to software.

#### 4.9.2.4 Deactivate Channel Interrupt

The deactivate channel interrupt instruction resets the MFPs activated state to a deactivated state. By placing the MFP in a deactivated state, it allows it to interrupt the software if it is enabled with a request pending. Channels with lower priorities are also able to interrupt software at this time if they win the interrupt poll. When this instruction is executed, condition codes of all zeros are returned to software.

### 4.10 Special MFP I/O Functionality

The MFP has four unique I/O functionalities, they are:

- Start I/O Queuing
- IOCD Prefetching
- Sense Information
- IOCD Skip Not Implemented

#### 4.10.1 Start I/O Queuing

All I/O devices support Start I/O (SIO) command queuing. However, for the MFP, the que for each subaddress is allocated from a globally linked list of 500 free entries. If all entries become allocated, the MFP will return channel busy condition codes if further I/O is attempted.

When an error occurs on a subaddress that contained queued start I/O's, one of two procedures will take place, depending upon how the subaddress was initialized. In the first procedure, mode 1, the SIO queue is purged and error status is returned for each SIO that was queued, plus, status is returned for all failed SIO's. In the second procedure, mode 2 (default mode), queued SIO's are processed as if there were no errors.

The Halt I/O command will only clear the current SIO in progress unless the clear que bit is set during the subaddress initialization. To clear all SIO's queued for the MFP, the Clear Que I/O instruction must be executed.

For the MFP, the following four considerations must be observed:

1. Halt I/O clears the current SIO and returns status.

2. Clear Que I/O clears the entire SIO que and returns status for each queued SIO instruction. Clear que is not supported on console subaddressed (FC and FD).

3. Only a single HIO or clear que can be handled by the MFP at any one time. If another is issued while a previous HIO is outstanding, subchannel busy CC's will be returned.

4. Only a single reset controller can be handled by the MFP at any one time. If another is issued before the previous is processed, subchannel busy CC's will be returned.

### 4.10.2 IOCD Prefetching

In some cases it is necessary for the MFP to prefetch IOCDs so that it can convert IOCLs into SCSI CDBs. In other cases, performance can be enhanced by firmware prefetching the IOCDs; however, during an IPL, the loading of the bootstrap by the IOCL may not be complete when a new IPL is issued. For this reason, when an IPL is received, the first IOCD is executed without fetching the next one. For the remaining portion of the IPL sequence, IOCDs may be prefetched until the chain of IOCDs is terminated. This is necessary to create a CDB with the appropriate block transfer count.

**NOTE**

Programmers should not create IOCLs that are partially complete when an SIO is issued because there are no guarantees that the MFP will not attempt to prefetch an IOCD before it is added by software to the IOCL.

### 4.10.3 Sense Information

Sense information is always cleared when a sense operation is completed. Halt I/O or clear que will leave sense in the MFP's controller unaltered. For SCSI devices, refer to specific controller documentation.

### 4.10.4 Skip IOCD's

The input/output processor (IOP) supports a function where the controller may skip IOCDs in a chain, under certain circumstances; however, the MFP does not support this function.

# CHAPTER 5

# REAL TIME FUNCTIONS

## 5.1 Introduction

This chapter contains the information pertaining to the real time functions of the Multi-Function Processor (MFP).

## 5.2 Overview

The real time function (RTF) provides the computer system with an interval timer, real time clock, external interrupts (both input and output), as well as, software generated interrupts. Each of these four distinct areas are described in this chapter.

The MFP implements the real time functions on a separate logical I/O channel. The physical address of the real time function channel is always an odd address that is one hexadecimal value greater than the MFP's class F channel. Only the first 16 subaddresses of the real time function channel are supported. 12 of the subaddresses support external interrupt capability. One of the subaddresses support the real time clock and another supports the interval timer. The remaining two subaddresses support software interrupts only; however, all 16 subaddresses can support software controlled interrupts. Although most of the subaddresses support more than one function, it is NOT recommended that one subaddress be used for more than one function at a time. Table 5-1 lists the subaddresses and there supported functions.

## 5.3 Real Time Function Interrupts

The interrupt protocol is similar to that described in chapter 4 of this manual; the differences between the interrupt protocols are described in the following sections. It is recommended that the programmer understand class F protocols before reading this section. The major differences are the supported instructions and the fact that each subaddress of the RTF has its own interrupt priority verses class F with one global priority.

### 5.3.1 Scratch Pad Entries

Scratch pad entries are formatted as described in chapter 4; however, several special considerations must be taken when building a scratch pad entry for the RTF. Since the RTF supports an interrupt priority for each subaddress, there must be one interrupt entry for each subaddress that is used. The only subaddress that supports command device instructions is the interval timer (subaddress 4). For this reason, only subaddress 4 requires a device entry in the scratch pad. Therefore, to fully support real time functions, there must be 16 interrupt entries and 1 device entry made in the scratchpad.

**Table 5-1. RTF Subaddress Functions**

| Subaddress | Software Interrupt | External Interrupt Number | Real Time Clock | Interval Timer |
|---|---|---|---|---|
| 0 | Yes | 0 | No | No |
| 1 | Yes | 1 | No | No |
| 2 | Yes | 2 | No | No |
| 3 | Yes | 3 | No | No |
| 4 | Yes | No | No | Yes |
| 5 | Yes | 4 | No | No |
| 6 | Yes | No | Yes | No |
| 7 | Yes | No | No | No |
| 8 | Yes | No | No | No |
| 9 | Yes | 5 | No | No |
| A | Yes | 6 | No | No |
| B | Yes | 7 | No | No |
| C | Yes | 8 | No | No |
| D | Yes | 9 | No | No |
| E | Yes | 10 | No | No |
| F | Yes | 11 | No | No |

Note: Yes = supported function; No = unsupported function

### 5.3.2 Internal Interrupt Polling

The RTF portion of the MFP supports internal interrupt polling that is based on subaddresses, not assigned interrupt priority. The subaddresses are arranged with the F representing the highest priority and 0 representing the lowest. In the MFP, the higher priority subaddress is serviced before a lower priority subaddress. In the event that a subaddress with a higher priority requests an interrupt, the MFP will deactivate a lower priority subaddress and allow the higher priority subaddress to poll for the interrupt. Once a subaddress wins the internal poll of the MFP, the MFP polls the SelBUS interrupt bus for an interrupt. The MFP always polls the SelBUS interrupt bus at its assigned priority level.

### 5.3.3 Interrupt Control Instructions

There are five interrupt control instructions available to the MFP to contend for an interrupt on the SelBUS interrupt bus. Of the five instructions, four are similar to those used in class F interrupts; however, class 3 has a fifth instruction called Request Interrupt.

The request interrupt causes two actions to take place. First, an external interrupt pulse is generated; second, a SelBUS interrupt is queued for the subaddress.

If the SelBUS interrupt is not desired, software may take one of two actions:

1. If software does not enable the subaddress, it prevents the interrupt (as well as any external interrupts) from occurring.

2. If interrupts are required from the subaddress, but not desired during execution of the request interrupt instruction, software may execute the following sequence: activate interrupt, request interrupt, disable interrupt, enable interrupt, deactivate interrupt. The activate interrupt prevents any pending interrupts from the subaddress from occurring. The request interrupt causes the external interrupt signal to be pulsed and a SelBUS interrupt is queued. The disable interrupt removes the pending interrupt for the subaddress from the queue. The enable interrupt cancels the disable interrupt with the exception of the dequed interrupt. Finally, the deactivate interrupt allows requests from the subaddress to poll again.

The MFP's real time functions do not queue interrupt requests in a stack. Only one interrupt can be pending at any one time. For example, two interrupt requests are executed while the RTF is in a disabled state, the RTF is enabled, and only one interrupt occurs as a result of the two request interrupt instructions.

### 5.4 Software Interrupts

Software interrupts are executed through the use of the request interrupt instruction. When software executes the instruction, it causes a SelBUS interrupt to occur; however, the interrupt priority level address must be in an inactive state and enabled before the interrupt can occur. All real time function subaddresses on the MFP support this function.

### 5.5 External Interrupts

External interrupts allow the capability of external devices to interrupt the host computer through a mechanism. The mechanism consists of a latched interrupt input and a request interrupt output on the external interrupt device interface.

The external interrupt device interface can be used to signal events to remote computers or peripherals. This is done through the execution of the request interrupt instruction which causes an interrupt pulse to be sent out to the remote device.

The external interrupt device interface can also be used to receive and acknowledge interrupts from remote computers or peripherals. This is done by enabling the appropriate external interrupt subaddress to perform an interrupt. When the remote device generates an interrupt pulse, the MFP's real time function begins polling for an interrupt to software. When the software interrupt context switch occurs, the MFP's real time function fires the external interrupt output signal which can be used by the remote computer or peripheral as an interrupt occurred indication.

Implementing both input and output external interrupts on one subchannel can cause difficulties, depending on the application. The difficulty is due to the request interrupt instruction which causes both the external interrupt output, as well as, the external interrupt input (software interrupt). The software interrupt is indistinguishable from an external MFP interrupt. If either of the above procedures for preventing the interrupt from happening,

incoming external interrupts may be lost. The problem can be avoided by one of the two methods described:

1. Define the interface to the remote device to avoid any situations where an incoming external interrupt occurs at the same time that a request interrupt instruction is executed.

2. Use a pair of external interrupt subaddresses, one for input and one for output.

Note, similar problems can occur and must be prevented at the device side of the interface, because the request interrupt output line can occur as a result of a software context switch, or, as a result of a request interrupt instruction.

### 5.5.1 External Interrupt Input

The external interrupt input is a pulse and circuit, see figure 5-1. The external interrupt input pulse has a minimum duration parameter of 300 nanoseconds and a maximum duration parameter of 1000 nanoseconds. The pulse is latched by the real time function hardware until firmware is able to process it and start polling for a SelBUS interrupt. To prevent saturation of the channel and software with interrupts, do not repeat this pulse at a frequency greater than 1 thousand hertz.

### 5.5.2 External Interrupt Output

The external interrupt output is a pulse, see figure 5-2. The pulse is generated as a result of two possible actions, the execution of a request interrupt instruction, or, during a software context switch that is caused by an interrupt from the associated real time function subaddress. The external interrupt output pulse has a duration parameter of 450 nanoseconds.

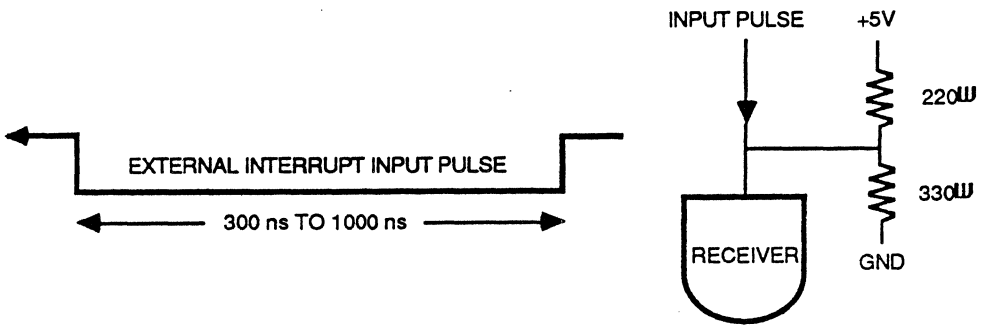
## 5.6 Interval Timer

The interval timer is a processor on the MFP that can be programmed by the command device instruction as follows:

- Select 1 of 3 counting rates.
- Select either single or multiple interrupts for a single count value.
- Enabled or disabled by software interrupt instructions.
- Write or read the interval timer count.

When the interval timer is read or written, the data is taken directly from the CPU's register 0 (zero).

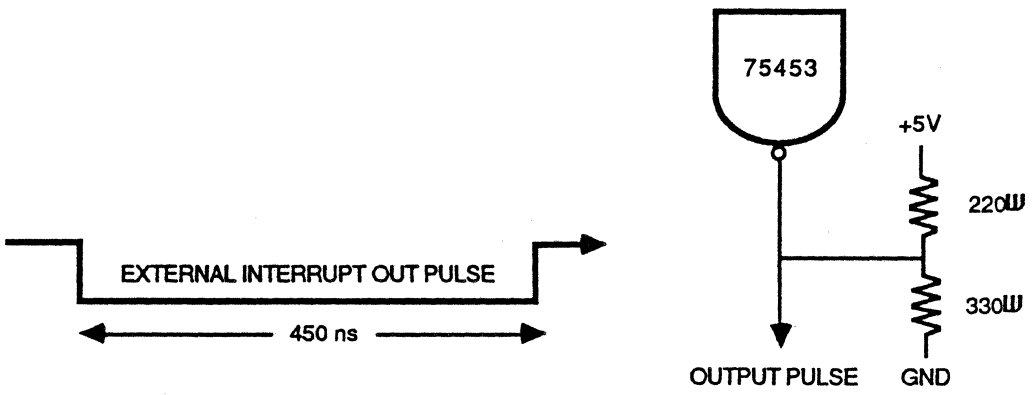
The interval timer is controlled by software command device (CD) instructions using a CPU register to/from input/output concept. The interval timer or CPU does not need to access memory during a CD instruction (other than normal instruction fetches); therefore, the execution time of a CD instruction is reduced. During a CD instruction execution, general purpose register zero of the CPU is used to send data (initial count) to or receive status (current count) from the interval timer. See figure 5-3 for an illustration of the CD instruction.




| Characteristics | Specifications |
|---|---|
| Signal Driver | 75453 recommended |
| Driving End Terminator | 220 ohms to +5vdc<br>330 ohms to ground (GND) |
| Pulse Width | nominal = symetrical<br>- minimum 300 nanoseconds<br>- maximum 1000 nanoseconds |
| Pulse Repetition Rate | 1000 hertz |

87D3J04

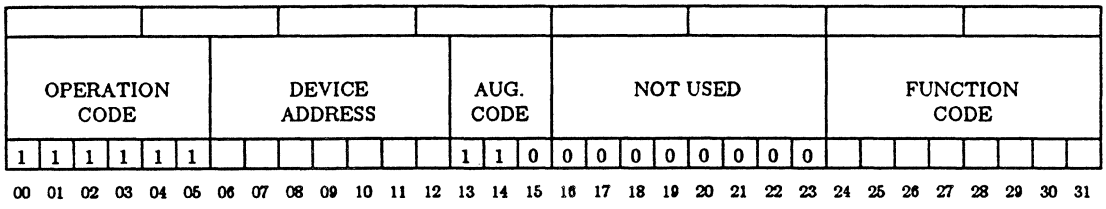
Figure 5-1. External Interrupt, Input Pulse Parameters




| Characteristics | Specifications |
|---|---|
| Logical Level Outputs | one = +0.8 vdc (external interrupt request)<br>zero = +2.4 vdc (no request) |
| Signal Driver | 75453 |
| Receiving End Terminator | 220 ohms to +5vdc<br>330 ohms to ground (GND) |
| Pulse Width | 450 nanoseconds |

87D3L01

**Figure 5-2. External Interrupt, Output Pulse Parameters**




| Bit | Value | Description |
|---|---|---|
| 25 | 1 | Specifies read time, causing the 32 bit contents of the timer to be loaded into GPR0. This may be done at any time (even while counting). |
| 26 | 1 | Specifies program interval timer and bits 27-31 are valid. |
| 27 | 1 | Start interval timer. |
| | 0 | Stop interval timer. |
| 28 | 1 | Load bits 0-31 from GPR0 into the interval timer bits 0-31. |
| | 0 | Does not alter the interval timer starting count value. |
| 29 | 1 | Generate multiple interrupts. When count of zero is reached, the interval timer generates an interrupt, reloads starting count, and continues counting. This is repeated until this bit is reset or the timer is stopped. |
| | 0 | Generate single interrupt. When count of zero is reached, the interval timer generates an interrupt, and continues counting negative. |
| 30-31 | | This is a code that is used to select the count frequency. |
| | 00 | Select jumpered frequency. |
| | 01 | Select jumpered frequency. |
| | 10 | Select real time clock frequency. |
| | 11 | Select external clock frequency. |

Figure 5-3. Command Device (CD) Instruction Format

The interval timer does not support test device (TD) instructions. If an attempt is made to execute a TD 4000 or TD 8000 instruction, a condition code of all zeros will be returned. If an attempt is made to execute a TD 2000 instruction, a condition code of 2 (two) will be returned, indicating that the transfer was not performed.

## 5.7 Real Time Clock

The real time clock subaddress of the real time function portion of the MFP generates interrupts to the host CPU at a frequency dependent on the setting of the jumpers located on the MFP circuit card. If no jumpers are inserted, the clock is disabled. If the external clock frequency is selected, an externally generated clock can be used. The parameters for this signal are listed in table 5-2.

**Table 5-2. Real Time Clock Parameters**

| Characteristics | Specifications |
|---|---|
| Signal Driver | 75453 recommended |
| Driving End Terminator | 220 ohms to +5vdc |
| Pulse Width | nominal = symetrical minimum 300 nanoseconds |
| Pulse Repetition Rate | 1000 hertz (maximum) |

# CHAPTER 6

# I/O SUPPORT

## 6.1 Introduction

This chapter contains the information pertaining to I/O support for the Multi-Function Processor (MFP).

## 6.2 Overview

The MFP provides I/O support for small computer system interface (SCSI) peripheral devices, a line printer, and eight asynchronous communication ports. It also provides an expansion port for a Gould designed general purpose device interface circuit card (refer to the appropriate technical manual for a description of this port).

## 6.3 SCSI Support

SCSI is a local I/O bus that can burst at data rates up to 1.25 megabytes per second, depending upon circuit implementation and peripheral devices. The primary objective of the SCSI is to provide the computer system with device independence within a class of peripheral devices. Thus, different disk drives, magnetic tape units, printers and communications devices can be added to the computer system without requiring modification to the hardware or software. Provisions are made for additional nongeneric features and functions through vendor unique fields and codes.

It is assumed that the programmer is familiar with the SCSI interface and protocol, if not, the American National Standard, ANSI X3.131-1986 or later, gives a complete description of the SCSI interface.

The SCSI host adapter, resident to the MFP, decodes the command byte of an IOCD that it is to execute. The IOCD provides the information required to transfer the appropriate command descriptor block (CDB) and the associated data, to or from the SCSI peripheral device. The MFP also provides an option which allows user built CDBs to be transmitted across the SCSI bus.

There are two variations within the class F protocol; they are, class F1 and class F2. In class F1 protocol, software provides a buffer and address in which the CDB is prebuilt. Once built, the channel fetches the CDB from memory and sends it to the controller. In class F2 protocol, the MFP builds and sends the appropriate CDB based on the command in the IOCD.

### 6.3.1 SCSI Initialization

The MFP is designed to optimize the most common disk and tape operations that an operating system might perform. To do this, the first I/O operation to any SCSI controller after a channel reset, SelBUS reset, or IPL sequence will cause the MFP channel to take special actions. These special actions occur in a series of events, as shown in figure 6-1. If errors occur

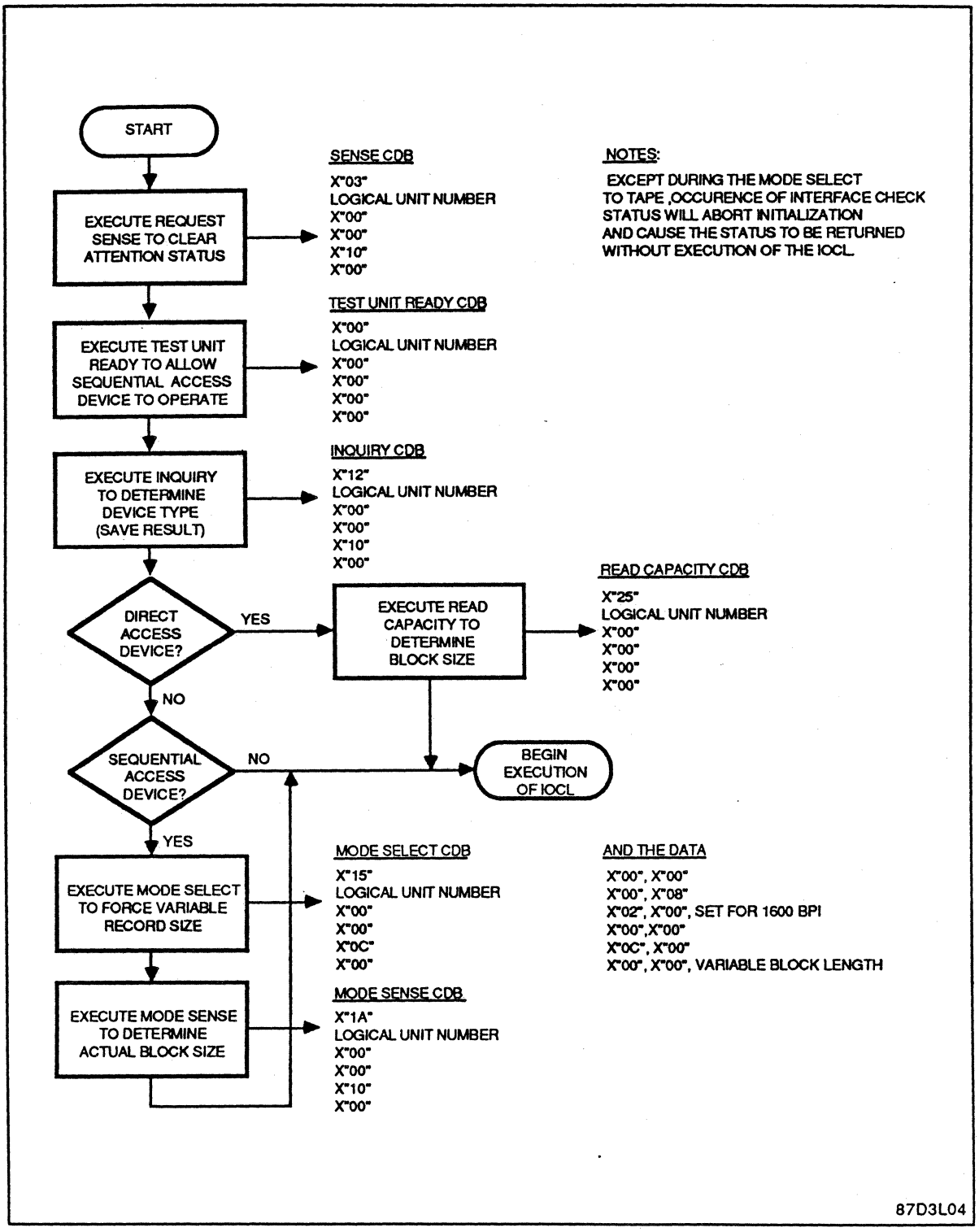



Figure 6-1. SCSI Initialization Protocol

during this initialization process (except during execution of the mode select to a sequential access peripheral device), it will be aborted with status returned to the software and the IOCL will not be executed.

For direct access peripheral devices, this initialization process is used to determine the logical blocksize of the peripheral device. The logical block size is used by the MFP to generate CDBs, with the correct block count, during execution of class F2 reads and writes.

For sequential access peripheral devices, this initialization process is used to set the peripheral device into a known mode of operation. For example, a magnetic tape unit would be set to 1600 BPI with a variable block length. This is done primarily to allow a system IPL to be performed from a sequential access peripheral device that would normally default to a different mode of operation. If an error status is returned after the execution of a mode select, the MFP will initiate execution of the IOCL rather than aborting the operation. If the tape rejects the mode selected, the MFP will issue a mode select to determine the actual operational parameters.

A side effect of initialization is the reset of the unit attention status; this happens upon powering up the system or reset on the SCSI bus.

**NOTE**

If the logical block size of a direct access peripheral device is changed while online through a mode select, the MFP will automatically reissue the initialization sequence prior to the next operation.

### 6.3.2 SCSI Commands

The SCSI commands supported by the MFP SCSI host adapter are listed in table 6-1. The table lists the CMD code, class of operation, as well as, the command name and description. All commands except read, write, and transfer command packet (TCP) result in class F2 operation. The read and write command operate in both class F1 and F2 protocols. If a TCP IOCD is command chained to a read or write command, the operation would follow the class F1 protocol. If a read or write command is data chained to another read or write which has been command chained to a TCP IOCD, the operation would follow the class F1 protocol. All other cases result in a class F2 operation.

#### 6.3.2.1 Write (direct and sequential access)

The write command requests that the target write the data transferred by the initiator to the medium. The write command operates in both class F1 and class F2 protocols.

The write command via sequential access writes data to the magnetic tape. It is executed in fixed or variable block size depending upon the configuration of the device and the total of the transfer count is the number of bytes (not blocks) to be transferred.

The write command via direct access writes data to a disk. The logical block address is taken from the MFP channel and specifies the logical block at which the operation shall begin. The register is updated whenever a seek IOCD command is executed, or, a read or write takes place. In the case of the seek command, it takes the logical address from memory and places it in the register. A read or write also updates the address that points to the next logical block to

**Table 6-1. SCSI Commands**

| CMD Code | Class | Name and Description |
|---|---|---|
| X"01" | F1,F2 | Write |
| X"02" | F1,F2 | Read |
| X"04" | F2 | Sense |
| X"07" | F2 | Seek (direct access only) |
| X"13" | F2 | Reassign Block (direct access only) |
| " | F2 | Write Many Filemarks (sequential access only) |
| X"23" | F2 | Rewind (sequential access only) |
| X"37" | F2 | Rezero (direct access only) |
| X"43" | F2 | Advance Record (sequential access only) |
| X"53" | F2 | Read Capacity (direct access only) |
| " | F2 | Backspace Record (sequential access only) |
| X"63" | F2 | Space One Filemark (sequential access only) |
| X"73" | F2 | Backspace One Filemark (sequential access only) |
| X"83" | F2 | Set Mode (NOP) (sequential access only) |
| X"93" | F2 | Write One Filemark (sequential access only) |
| X"A3" | F2 | Reserve Unit (direct access only) |
| X"B3" | F2 | Inquiry |
| X"C3" | F2 | Release Unit |
| X"D3" | F1 | Transfer Command Packet (specifies CDB to send) |
| X"E3" | F2 | Space Many Filemarks (sequential access only) |
| X"F3" | -- | Reserved |

be read. Normally, software builds a seek command that is command chained to a read or write; this ensures that the logical block address is correct.

The block count is calculated by taking the sum of the transfer counts for all of the data chained IOCDs. The sum is divided by the logical block size obtained during initialization and rounded up to the final block count. Note, only four logical block sizes are supported by class F2 protocol: 256, 512, 768, and 1024. A block size of 256 can be used to emulate a block size of 768 as long as the block conversion bit is not set during subaddress initialization (or subaddress initialization is not performed). See the seek command description for more detail.

The CDB that is built will vary, depending on the total block count and the logical block address. A group 0 (zero) CDB is set if the block count is less than 256 and the logical block address is less than 2,097,152. Otherwise, a group 1 (one) CDB (extended write) is sent to the peripheral device.

### 6.3.2.2 Read (direct and sequential access)

The read command requests that the target transfers data to the initiator. The read command operates in both class F1 and class F2 protocols. It transfers data via sequential access or direct access. The read command works exactly like the write command.

The read command via sequential access reads data from the magnetic tape. It is executed in fixed or variable block size depending upon the configuration and the total of the transfer

count is the number of bytes (not blocks) to be transferred. The logical block address specifies the logical block at which the operation shall begin. Fixed block sizes supported by magnetic tape are 256, 512 and 1024. Note, fixed block sizes for both read and write, are supported only for devices that do not have the capability to run in a variable block mode (see figure 6-1).

The read command via direct access reads data from disk. The logical block address is taken from the MFP channel and specifies the logical block at which the operation shall begin. The register is updated whenever a seek IOCD command is executed, or, a read or write takes place. In the case of the seek command, it takes the logical address from memory and places it in the register. A read or write also updates the address that points to the next logical block to be read. Normally, software builds a seek command that is command chained to a read or write; this ensures that the logical block address is correct.

The block count is calculated by taking the sum of the transfer counts for all of the data chained IOCDs. The sum is divided by the logical block size obtained during initialization and rounded up to the final block count. Note, only four logical block sizes are supported by class F2 protocol: 256, 512, 768, and 1024. A block size of 256 can be used to emulate a block size of 768 as long as the block conversion bit is not set during subaddress initialization (or subaddress initialization is not performed). See the seek command description for more detail.

The CDB that is built will vary, depending on the total block count and the logical block address. A group 0 (zero) CDB is set if the block count is less than 256 and the logical block address is less than 2,097,152. Otherwise, a group 1 (one) CDB (extended read) is sent to the peripheral device.

#### 6.3.2.3 Sense (direct and sequential access)

The sense command provides a means for a target to report its medium, logical unit, or peripheral device parameters to the initiator. It results in a request sense CDB to be sent to the SCSI controller if it is a SCSI address. The SCSI address and transfer count of the IOCD are used to place the sense data in main memory. This IOCD can be data chained, in which case, the sum of all of the data chained IOCDs transfer counts are used to generate the allocation length. The allocation length is placed in the request sense CDB. This command is supported by all SCSI peripheral devices.

#### 6.3.2.4 Seek (direct access only)

The seek command requests that the logical unit seek, move to, the specified logical block address. It does not issue a CDB to the addressed peripheral device; instead, an internal register of the MFP is loaded with the specified logical address. This logical address is added to the next read or write CDB that is built. The real data address field of the IOCD points to the buffer containing 1, 2, or 4 bytes of logical disk address. This logical address specifies a byte boundary in memory. If one or two bytes are specified, the remaining bytes of the seek register are filled with zeros. This IOCD may not be data chained.

#### 6.3.2.5 Reassign Block (direct access only)

The reassign command requests the target to reassign the defective logical block(s) to an area on the logical unit reserved for this purpose. The address of the IOCD points to the defect list to be reassigned. The transfer count must be equal to the length of the defect list. This command (IOCD) may not be data chained.

#### 6.3.2.6 Write Many Filemarks (sequential access only)

The write many filemarks command causes a specified number of filemarks to be written to the medium beginning at the current medium position on the logical unit. The byte count field of the IOCD is used as the count field of the CDB. The real data address of the IOCD is ignored. This command may not be data chained.

#### 6.3.2.7 Rewind (sequential access only)

The rewind command requests that the target rewind the logical unit to the beginning-of-medium, or, the load-point. The byte count and real data address field of this command (IOCD) are not used. This rewind command does not invoke an immediate rewind; therefore, status is not returned until the rewind has been completed.

#### 6.3.2.8 Rezero (direct access only)

The rezero command requests that the target set a logical unit to the logical block address of zero. It does not issue a CDB to the addressed peripheral device; instead, an internal register of the MFP is loaded with a logical address of zero. This logical address is added to the next read or write CDB that is built. This logical address specifies a block of zeros on the unit, that is the next block to be read or written. The real data address and the byte count field of the IOCD are ignored. This IOCD may not be data chained.

#### 6.3.2.9 Advance Record (sequential access only)

The advance record command provides a means for the initiator to request that the target space to a specific record. This command is only valid for sequential access peripheral devices. The byte count field of the IOCD is used as the count field of the CDB. The real data address of the IOCD is ignored. This command may not be data chained.

#### 6.3.2.10 Read Capacity (direct access only)

The read capacity command provides a means for the initiator to request information regarding the capacity of the logical unit. The byte count field of this IOCD determines the maximum number of data bytes that are transferred to main memory (Note, eight data bytes are always taken from the device). The real data address of the IOCD specifies the buffer address for the read capacity data. This IOCD may not be data chained.

#### 6.3.2.11 Backspace Record (sequential access only)

The backspace record command provides a means for the initiator to request that the target backspace to a specific record. This command is only valid for sequential access peripheral devices. The byte count field of the IOCD is used as the count field of the CDB. The real data address of the IOCD is ignored. This command may not be data chained.

#### 6.3.2.12 Space One Filemark (sequential access only)

The space one filemark command provides a means for the initiator to request that the target space to the next filemark. This command is only valid for sequential access peripheral devices.

The count is specified by one. Both the real data address and the byte count field of the IOCD are ignored. This IOCD may not be data chained.

#### 6.3.2.13 Backspace One Filemark (sequential access only)

The backspace one filemark command provides a means for the initiator to request that the target backspace to the previous filemark. This command is only valid for sequential access peripheral devices. The count is specified by one. Both the real data address and the byte count field of the IOCD are ignored. This IOCD may not be data chained.

#### 6.3.2.14 Set Mode (sequential access only)

The set mode command is treated as a no operation (NOP) by the MFP. It is only implemented to provide the compatibility with other tape controllers.

#### 6.3.2.15 Write One Filemark (sequential access only)

The write one filemark command causes one filemark to be written to the medium beginning at the current medium position on the logical unit. The count is specified as one. The real data address and the byte count field of the IOCD are ignored. This command may not be data chained.

#### 6.3.2.16 Reserve Unit (direct and sequential access)

The reserve unit command is used to reserve logical units for the use of the initiator. The reserve and release commands provide the basic mechanism for contention resolution in multiple-initiator systems.

The extent is specified as the entire unit. The reservation identification is specified as zero and third party reservation is not supported through this command. The real data address and the byte count field of the IOCD are ignored. This IOCD may not be data chained.

#### 6.3.2.17 Inquiry (direct and sequential access)

The inquiry command requests that information regarding the parameters of the target and its attached peripheral device(s) be sent to the initiator. The byte count field for this command (IOCD) and any other IOCDs that are data chained to it, are added together to obtain the total allocation length. The real data address of each IOCD specifies the buffer address(es) for the inquiry data.

#### 6.3.2.18 Release Unit (direct and sequential access)

The release unit command is used to release a previously reserved logical unit. It is not an error for an initiator to attempt to release a reservation that is not currently active. In this

case, the target returns GOOD status without altering any other reservations.

Logical Unit Release - This command will cause the target to terminate all logical unit reservations that are active from the initiator to the specified logical unit.

For the MFP, the extent is specified as the entire unit. The reservation identification is specified as zero and the third party reservation is not supported through this command. The real data address and byte count field of the IOCD are ignored. This IOCD may not be data chained.

#### 6.3.2.19 Transfer Command Packet (direct and sequential access)

The transfer command packet (TCP) command sends a SCSI CDB to a peripheral device. If a data transfer is requested, this IOCD must be command chained to a read or write IOCD. Data chaining is not permitted. The CDB can reside in memory on any byte bounded address. A count of less than 6 in the TCP IOCD results in the return of program check status.

If an error occurs during execution of a TCP IOCD that has been command chained to a read or write, the IOCLA in the returned status will reflect the read or write IOCD plus eight, rather than the TCP IOCD plus eight.

To command chain two class "F2" IOCDs, the TCP IOCD must be followed by a read or write IOCD. This read or write IOCD is then command chained to the actual class "F2" IOCD.

#### 6.3.2.20 Space Many Filemarks (sequential access only)

The space many filemarks command provides a means for the initiator to request that the target space a specified number of spaces to the next filemark. This command is only valid for sequential access peripheral devices. The byte count field of the IOCD is used as the count field of the CDB. The real data address of the IOCD is ignored. This IOCD may not be data chained.

### 6.3.3 Non-Optimized SCSI Support, Class F1

The SCSI support of class F1 protocol commands is considered non-optimized, because it involves more software/firmware overhead than the standard class F protocol. The increase in software overhead is attributed to the fact that most I/O operations require at least 2 IOCDs, unless no data is transferred. Any SCSI CDB can be executed by building it in memory and transmitting it to the SCSI controller through the use of a D3 command. To do so, the D3 command of an IOCD is chained to a read or write IOCD command. The IOCD command contains the data buffer address and count. See figure 6-2 for an illustration of the SCSI class F1 protocol.

The SCSI class F1 protocol looks the same to the software as the class F protocol with the exception of a special transfer command packet IOCD containing the command D3. The address in this IOCD points to the actual SCSI command descriptor block that is to be sent across the SCSI bus to a targeted controller. If any data is transferred, this transfer command packet command must be command chained to a read or write IOCD which points to the associated data buffer. The length of the CDB to be sent to the SCSI peripheral device must be contained in the byte field of the transfer command packet IOCD.

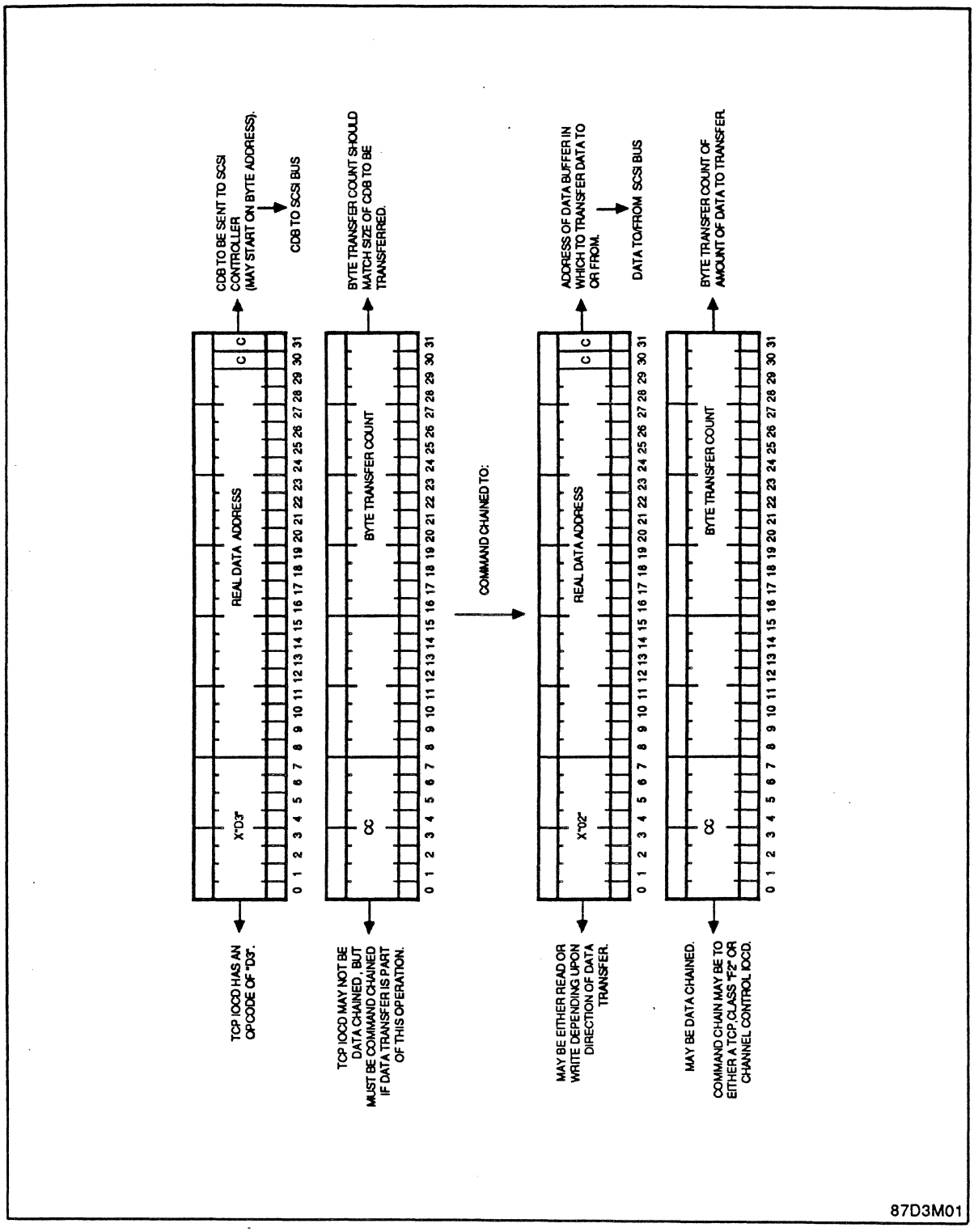



Figure 6-2. SCSI Class F1 Protocol

### 6.3.4 Optimized SCSI Support, Class F2

The SCSI support of class F2 protocol is optimized because it involves the translating of IOCDs into the appropriate SCSI CDB command by the MFP. Optimization is realized because it improves the efficiency of the software by reducing the need to format both IOCDs and SCSI CDBs. See figure 6-3 for an illustration of the SCSI class F2 protocol.

Because of the diversity of the SCSI CDBs, only a limited number of conversions from IOCD to CDBs are implemented under class F2 protocol. This includes special support for direct (disk) and sequential (tape) access peripheral devices. Note, inquiry and request sense should work on any SCSI peripheral device. Also, the CDB built during reads and writes may be grouped in either Group 0 or Group 1, depending upon the byte count and block address requested.

**NOTE**

The class F2 subset of commands is adequate to operate either disk or magnetic tape in an operating software environment; however, both mode initialization and diagnostics must be performed using the class F1 protocol.

### 6.3.5 Mixing Class F1 and F2 Protocols

Class F1 and F2 protocol operations may be mixed together within the same IOCL. For example, a class F1 read command can be command chained to a class F2 sense command as follows: SCP-READ-READ-SENSE; in this example, the first three IOCDs are from a single SCSI read operation, with the sense being executed as a SCSI request sense operation. Note, the channel creates the SCSI CDB solely from the information contained in the IOCL for each start I/O. Read or write IOCDs may be interleaved with other types of IOCDs in the class F2 IOCL, that is to say, a valid sequence could be SEEK-READ-READ-SENSE-WRITE.

The MFP channel differentiates between the various I/O classes based upon which class of I/O addressed peripheral device is designated to run. However, controllers on either SCSI bus (SCSI bus 1 or SCSI bus 2) can be driven by either class F1 or class F2 protocol, or a combination of them.

Command chaining a TCP (class F1) to a non TCP (class F2) requires at least one read or write IOCD following the TCP or else the non TCP IOCD may not be decoded correctly. TCPs may be command chained directly to another TCP as long as there are no data transfers expected.

## 6.4 Line Printer

The MFP supports one line printer. The line printer port is a parallel line interface. The subaddress for this port is X"F8".

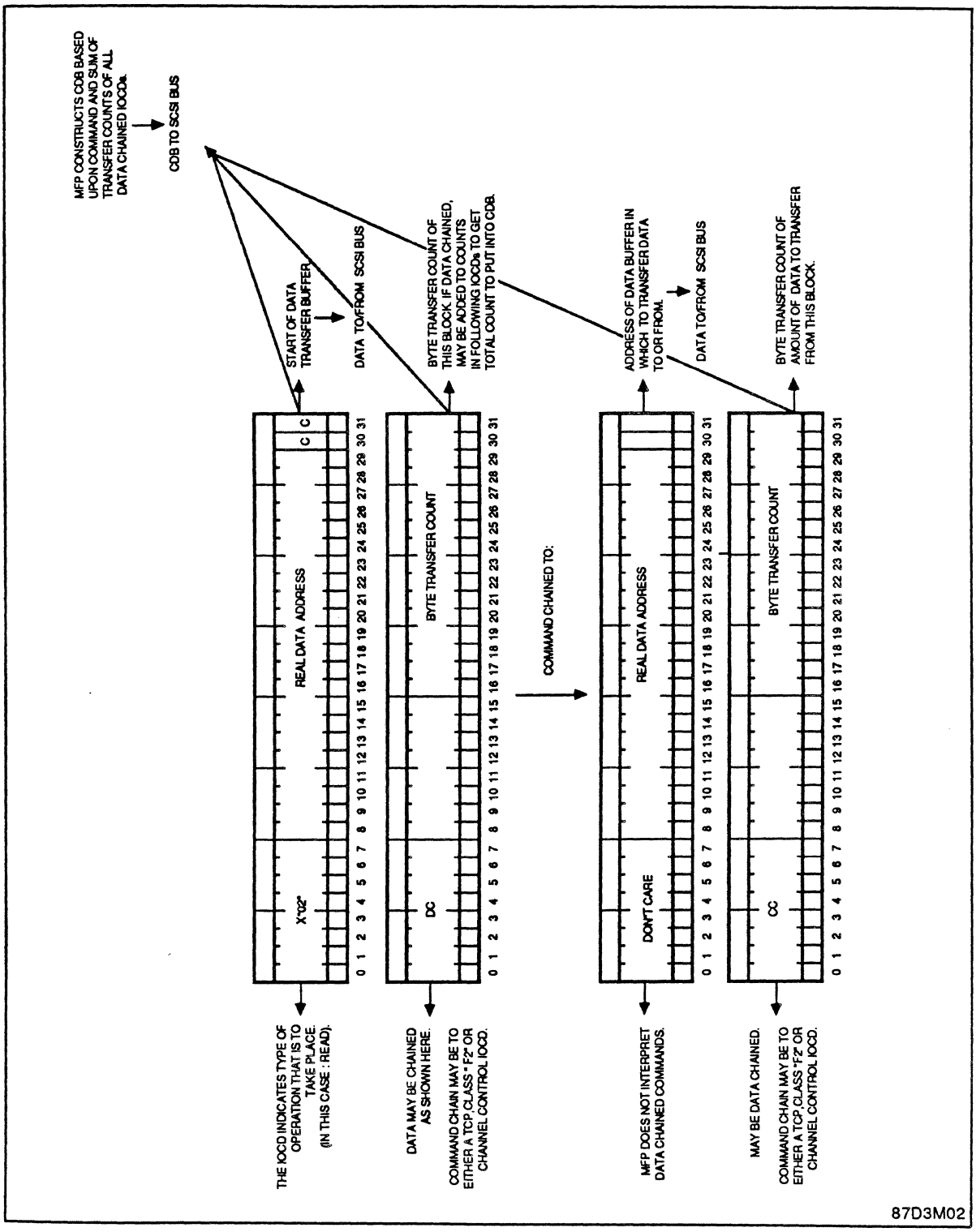



Figure 6-3. SCSI Class F2 Protocol

### 6.4.1 Line Printer Control Commands

The line printer control commands supported by the MFP allow paper moves to occur without a data transfer instruction (when there is no data in the line printer's data buffer). Note: the printers data buffer can be cleared by executing the clear buffer command.

The line printer control commands supported by the MFP are:

- No operation
- Control command
- Do paper control (force print)
- Line advance skip count
- Advance to top of form (TOF)
- Clear line printer buffer

The bit format for the line printer control commands are illustrated in figure 6-4. Bit seven is the most significant bit. Bits zero through five are modifier bits to the basic control command. If no modifier bits are defined in the command, a no operation command is executed. A no operation command performs no actions; however, channel-end/device-end status is returned to software. When a modifier bit is set, that appropriate action takes place.

If more than one function is to be executed at any one time, the sequence of execution for the control command functions are: clear the line printer buffer, advance to the top of form, and line advance skip count. The following sections describe the functions of the line printer control commands.

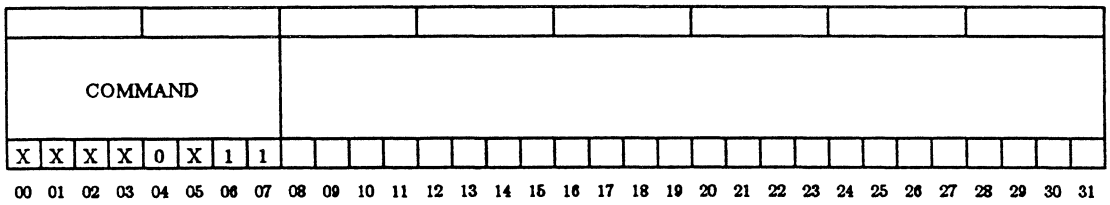
#### 6.4.1.1 Do Paper Control (Force Print)

The do paper control (force print) command is defined by setting bit 5. This command causes the line printer to print the contents of its buffer prior to completion of this control command. If this bit is not set, the line advance skip count command and advance to the top of form command functions will not be executed until one of the following two events occur: one, the execution of a control command with the do paper control bit set, or two, the line printer receives a carriage return character. The carriage return character forces the printer to print the contents of the buffer.

#### 6.4.1.2 Line Advance Skip Count

The line advance skip count command is defined by setting bits 2 and 3. This command defines how many vertical space lines the printer should place between each line of text. The number of lines to be skipped are determined by setting bits 2 and 3 as follows:

| Bits- | 2 | 3 | Definition |
|---|---|---|---|
| | 0 | 0 | Skip 0 lines (carriage return) |
| | 0 | 1 | Skip 1 line |
| | 1 | 0 | Skip 2 lines |
| | 1 | 1 | Skip 3 lines |




**Command - Bits 0 - 7:**

| Bits | | | | | | | | Hex | Command |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 03 | No operation |
| M | M | M | M | 0 | M | 1 | 1 | -- | Control command |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 07 | Do paper control (force print) |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 33 | Line advance skip count |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 43 | Advance to top of form (TOF) |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 83 | Clear line printer buffer |

**Bits 8 - 31: not defined**

Notes:

1) X = variable bit

2) M = modifier bit

**Figure 6-4. Line Printer Control Commands Word Format**

The line feed character X"0A" is sent to the line printer whenever the skip one, two, or three lines command is executed (X"0A" for skip one line, X"0A0A" for skip two lines, and X"0A0A0A" for skip three lines). The number of line feed characters sent to the line printer depends upon which skip instruction is used. Executing skip 00 causes the carriage return character X"0D" to be sent to the line printer. Executing the advance to the top of form instruction causes the form feed character X"0C" to be sent to the line printer.

#### 6.4.1.3 Advance to Top Of Form

The advance to top of form command is defined by setting bit 1. When bit one is set, it instructs the line printer to advance the paper to the top of the form. This bit is only used if the line printer contains a vertical format unit (VFU). Executing the advance to top of form command causes the form feed character X"0C" to be sent to the line printer.

#### 6.4.1.4 Clear Line Printer Buffer

The clear line printer buffer command is defined by setting bit 0. When bit zero is set, it instructs the line printer to clear its buffer. Clearing the line printer buffer always takes place prior to other operations selected by this control command.

### 6.4.2 Paper Control Instruction

The paper control instructions may be embedded in the data stream. The code for the different paper control instructions, the number for each instruction, as well as, the definition of each instruction are unique to each individual line printer. Therefore, they are not described in this document. Refer to the individual line printer manuals for a description of paper control instructions.

The paper control instructions must have the most significant bit set to one (1) because the controller uses this bit (the eighth bit) to generate a signal to inform the line printer that the data must be interpreted as a paper control instruction. The paper control instruction format is as follows:

| | |
|---|---|
| 1 | 7-BIT CODE |

The format for commands and data embedded into the data stream is as follows:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| LF | DATA | FF | DATA | CR | DATA | LF | LF | LF |

The sequence of events described in the above illustration are: line feed, data, followed by a form feed, data, followed by a carriage return, data, followed by three line feeds.

The order in which data and commands are embedded is unrestricted. The control characters embedded in the data stream are executed by the line printer without entering its data buffer. It should be noted that different line printers have different maximum line lengths. For example, a line printer could have a maximum line length of 136 bytes; therefore, for this example, the data streams would have to be limited to a maximum length of 136 bytes of data between control characters. These embedded control characters must be the actual ASCII codes recognized by the line printer.

### 6.4.3 Line Printer Write Commands

The line printer write commands transfer data to the line printer for printing. The bit format for the write commands are illustrated in figure 6-5. Bit seven is the most significant bit and bits zero through five are the modifier bits to the basic write command. If no modifier bits are set, the command will do a write without a carriage return or carriage advance. If any of the modifier bits are set, the appropriate action will take place.

If more than one instruction is executed at the same time, the sequence of execution for the control command functions are: clear the line printer buffer, advance to the top of the form, and skip lines. The following sections describe the function of the modifier bits.

#### 6.4.3.1 Allow Paper Control

The allow paper control command is defined when bit 5 of the write command is set to one. This command allows for automatic insertion of control commands in the data. The control commands are carriage return (CR), line feed (LF), and top of form (TOF). The are defined in command bits one, two, and three, respectively. When this bit is set to zero, all control commands must be inserted in either the data stream, or, sent by the user as an individual control command.

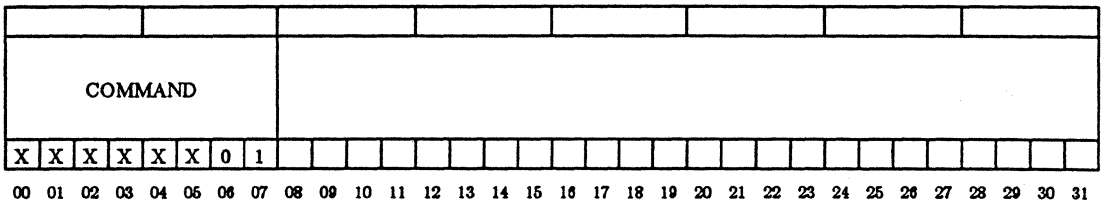
#### 6.4.3.2 Post Print

The post print command is defined when bit 4 of the write command is set to one. This command allows for carriage control operations prior to printing data. When this bit is set to zero, it indicates that the line printer should first print data, then perform carriage control operations (pre-print).

#### 6.4.3.3 Line Advance Skip Count

The line advance skip count command is defined by bits 2 and 3 of the write command. These bits specify the number vertical space lines the printer should place between each line of text. The number of lines to be skipped are determined by setting bits 2 and 3 as follows:

| Bits- 2 | 3 | Definition |
|---|---|---|
| 0 | 0 | Skip 0 lines (carriage return) |
| 0 | 1 | Skip 1 line |
| 1 | 0 | Skip 2 lines |
| 1 | 1 | Skip 3 lines |




**Command - Bits 0 - 7:**

| Bits | | | | | | | | Hex | Command |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 01 | Write with no carriage return |
| M | M | M | M | M | M | 0 | 1 | -- | Write command |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 05 | Allow paper control |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 09 | Post print (do carriage control before printing) |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 31 | Line advance skip count |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 41 | Advance to top of form (TOF) |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 81 | Clear line printer buffer |

**Bits 8 - 31: not defined**

**Notes:**

1) X = variable bit

2) M = modifier bit

Figure 6-5. Line Printer Write Commands Word Format

The line feed character X"0A" is sent to the line printer whenever the skip one, two, or three lines command is executed (X"0A" for skip one line, X"0A0A" for skip two lines, and X"0A0A0A" for skip three lines). The number of line feed characters sent to the line printer depends upon which skip instruction is used. Executing skip 00 causes the carriage return character X"0D" to be sent to the line printer. Executing the advance to the top of form instruction causes the form feed character X"0C" to be sent to the line printer.

#### 6.4.3.4 Advance To Top Of Form

The advance to top of form command is defined by setting bit 1. When bit one is set, it instructs the line printer to advance the paper to the top of the form. This bit is only used if the line printer contains a vertical format unit (VFU). Executing the advance to top of form command causes the form feed character X"0C" to be sent to the line printer.

#### 6.4.3.5 Clear Line Printer Buffer

The clear line printer buffer command is defined by setting bit 0. When bit zero is set, it instructs the line printer to clear its buffer. Clearing the line printer buffer always takes place prior to other operations selected by the control command and write command.

### 6.4.4 Line Printer Sense Command

The line printer sense command (X"04") returns four bytes of sense information to software. The first two bytes contain status flags as follows:

**Sense Data Byte Zero**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | Bus out check |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | Operator intervention required |
| | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Command reject |

**Sense Data Byte One**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Top of form |
| | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | Beginning of form |
| | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | Off line |
| | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | Device check |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | Device power |
| | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Device verify |

The last two bytes of sense data contain the remaining byte count in the line printer controller; however, it is not transferred to the device. The remaining byte count should always be zero.

#### 6.4.4.1 Bus Out Check

Bus out check, when equal to one, indicates that a parity error has been detected during the execution of the previous command.

#### 6.4.4.2 Operator Intervention Required

Operator intervention required, when equal to one, indicates that the line printer is not operational. Refer to byte one of the sense data to determine the exact problem.

#### 6.4.4.3 Command Reject

Command reject, when set to one, indicates that the controller received an invalid command. The command is not executed and a unit check status is returned.

#### 6.4.4.4 Top of Form

Top of form, when set to one, indicates that the line printer is at the top of the form. This bit is only significant if the line printer has a vertical format unit (VFU) specified.

#### 6.4.4.5 Beginning of Form

Beginning of form, when set to one, indicates that the line printer is at the beginning of the form. This bit is only significant if the line printer has a vertical format unit (VFU) specified.

#### 6.4.4.6 Off Line

Off line, when set to one, indicates that the line printer is off line. This bit, when set, is always accompanied by bit one of byte zero (operator intervention required).

#### 6.4.4.7 Device Check

Device check, when set to one, indicates that the line printer is not ready. This bit, when set, is always accompanied by bit one of byte zero (operator intervention required).

#### 6.4.4.8 Device Power

Device power, when set to one, indicates that the line printer does not have power. This bit, when set, is always accompanied by bit one of byte zero (operator intervention required).

#### 6.4.4.9 Device Verify

Device verify, when set to one, indicates that the line printer's interface cable is disconnected. This bit, when set, is always accompanied by bit one of byte zero (operator intervention required).

### 6.4.5 Channel Status Responses

The line printer controller returns one of three variations of channel status upon the completion of each command. The channel status is in the form of a line printer (device) status byte. The line printer status byte is defined as follows:

| Line Printer Status Byte | Definition |
|---|---|
| X"0C" | Normal completion |
| X"0E" | Unit check |
| X"0D" | Unit exception |

#### 6.4.5.1 Normal Completion

Normal completion status is returned when the operation is successfully completed and the carriage is not at the bottom of the form.

#### 6.4.5.2 Unit Check

Unit check status is returned when the operation has been aborted because of a problem that exists in the line printer. To determine what the problem is, issue a sense command.

#### 6.4.5.3 Unit Exception

Unit exception status is returned when the carriage is at the bottom of the form and the line printer has successfully completed an operation.

### 6.5 Asynchronous Port Support

The MFP supports eight asynchronous ports, each port is assigned a channel subaddress for receive and transmit functions. The subaddresses of the ports are E0 through EF, they are configured as follows:

| Port | Receive Subaddress | Transmit Subaddress |
|---|---|---|
| 0 | E0 | E8 |
| 1 | E1 | E9 |
| 2 | E2 | EA |
| 3 | E3 | EB |
| 4 | E4 | EC |
| 5 | E5 | ED |
| 6 | E6 | EE |
| 7 | E7 | EF |

**Note**

The MFP will only respond to E0 and E8 when the jumper X3-1 is installed.

### 6.5.1 Asynchronous Port Write Commands

The write commands supported by the asynchronous ports are: write, write with input subchannel monitoring, and write with hardware flow control only. All write commands must use subaddresses E8 through EF (hexadecimal). Data Set Ready (DSR) must be true when a write command is issued or the command will be rejected with a unit check. Exception, when a write command is issued to a port that is full duplex with modem ring enabled, the DSR check is bypassed. The write commands are configured as follows:

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Hex | Definition |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 01 | Write |
| | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 05 | Write with input subchannel monitoring |
| | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0D | Write with hardware flow control only |

#### 6.5.1.1 Write

The write command causes a block of data (1 to 65,535 bytes) to be transferred from main memory to the addressed subchannel. A channel end/device end (CE/DE) status is returned at the completion of the command. Note, read commands are accepted on the corresponding subchannel.

#### 6.5.1.2 Write with Input Subchannel Monitoring

The write with input subchannel monitoring command causes a block of data to be transferred to a device while simultaneously monitoring the corresponding input subchannel for receipt of a DC3 (X-OFF = 13 hexadecimal), DC1 (X-ON = 11 hexadecimal), or, ETX (End of Text = 03 hexadecimal) character.

If a DC3 is received, the output function is suspended until a DC1 command is received. After the command is completed, the port will continue to monitor for a DC3 until a new write or write with hardware flow control only command is issued, or, until the port is reinitialized.

If an ETX character is received while the write command is active, an orderly termination of the output function occurs and CE/DE device status is returned.

This command also monitors the clear to send (CTS) line; if the CTS line goes false, data transmission is temporarily suspended until CTS goes true again. This is a hardware function only, it should be used for devices with high transmission rates.

Data received that is not DC1, DC3, or ETX, is placed in a type-ahead buffer.

### 6.5.1.3 Write with Hardware Flow Control Only

The write with hardware flow control only command monitors the clear to send (CTS) line for flow control only. When this command is active, the read command is accepted on the corresponding read subchannel.

### 6.5.2 Asynchronous Port Read Commands

The asynchronous port read commands are: read, read echoplex, read with flow control, read with hardware flow control only. All read control commands must use subaddresses E0 through E7. Data Set Ready (DSR) must be true when a read command is issued or the command will be rejected with a unit check. Exception, when a read command is issued to a port that is full duplex with modem ring enabled, the DSR check is bypassed.

The asynchronous port read commands are configured as follows:

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Hex | Definition |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | A | S | P | 0 | 0 | 1 | 0 | X2 | Read |
| | 0 | A | S | P | 0 | 1 | 1 | 0 | X6 | Read echoplex |
| * | 0 | A | S | P | 1 | 0 | 1 | 0 | XA | Read with flow control |
| | 0 | A | S | P | 1 | 1 | 1 | 0 | XE | Read with hardware flow control only |

* If bit 0 is set in the command byte, DTR is used for flow control.
If bit 0 is reset in the command byte, RTS is used for flow control.

**Key:**

A - ASCII control character detect.
S - special character detect.
P - purge input buffer.
X - variable.

### 6.5.2.1 Read

The read command causes a block of data to be transferred to main memory from the addressed subchannel.

Data received before the read command is stored, it is included in the read. If the purge bit is set, the data received before the read command is dumped, only the data received after the read command is moved to system memory. Each receive subchannel has two 256 byte buffers. If both buffers are full (without a read command pending), the contents of the first buffer is dumped and becomes the second buffer. When this occurs, the sense overflow bit is set.

If the special character detect bit is set, the input operation is terminated whenever a previously defined eight bit character is received. At this point, the input operation is terminated and the special character is written into memory as the last character. The command is terminated with CE/DE device status and the special character detect bit is set in the sense data.

If the ASCII control character detect bit is set, the input operation is terminated whenever a control character (hexadecimal 00 through 1F) or delete character (hexadecimal 7F) is detected. The last character written into memory, is the ASCII control character. Data received after the ASCII control character is stored in the type ahead buffer.

#### 6.5.2.2 Read Echoplex

The read echoplex command allows a block of data to be read while all printable ASCII characters are sent to the corresponding subchannel.

If a write command is attempted on the output subchannel, a busy status will be returned. The command bits supported by this command are: purge, ASCII control character detect, and special character detect.

#### 6.5.2.3 Read with Flow Control

The read with flow control command is the complement to the write with input subchannel monitor mode command. Once this command is issued, the subchannel will stay in this mode until a nonflow control read command is issued. When the subchannel has filled a buffer with 456 bytes of data, without a read large enough/read pending command to free the buffer, it will send a X-OFF (hex 13) character out of the write subchannel. At the same time, either DTR (if bit 0 is set in the command), or, RTS (if bit 0 is reset in the command) is reset. When the subchannel frees a buffer, a X-ON (hex 11) is issued.

The command bits supported by this command are: purge, ASCII control character detect, and special character detect.

#### 6.5.2.4 Read with Hardware Flow Control Only

The read with hardware flow control only command is functionally equivalent to the read flow control command; except, only the hardware lines are toggled. Xon/Xoff characters are not sent out on the corresponding subchannel.

The command bits supported by this command are: purge, ASCII control character detect, and special character detect.

### 6.5.3 Asynchronous Port Control Commands

The asynchronous port control commands (bits 0 through 7) supported by the MFP are as follows:

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Hex | Definition |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 03 | No operation (NOP) |
| | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 04 | Sense - not a control command |
| | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0B | Define special character (one byte follows) |
| | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 13 | Reset data terminal ready |
| | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 17 | Set data terminal ready |
| | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1B | Reset request to send |
| | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1F | Set request to send |
| | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 33 | Reset break |
| | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 37 | Set break |
| | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 4B | Enabled port drivers |
| | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 4F | Disable port drivers |
| | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 53 | Transparent flow control |
| | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | FF | Define parameters (three bytes follow) |

#### 6.5.3.1 No Operation

The no operation (NOP) command does not cause any activity at the subchannel; however, it does cause a CE/DE device status byte to be returned to software.

#### 6.5.3.2 Sense

The sense command causes eight bytes of sense information to be transferred to main memory. The sense information pertains to the addressed asynchronous port.

**Sense Byte 0: Device Status**

The first six bits of byte 0 are standard for all controllers; however, the MFP only uses bits 0, 3, and 4.

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | - | - | - | - | - | - | - | Command reject (program violation) |
| | - | 0 | - | - | - | - | - | - | not used (always 0) |
| | - | - | 0 | - | - | - | - | - | not used (always 0) |
| | - | - | - | 1 | - | - | - | - | Equipment check |
| | - | - | - | - | 1 | - | - | - | Data check |
| | - | - | - | - | - | 0 | - | - | not used (always 0) |
| | - | - | - | - | - | - | 0 | - | not used (always 0) |
| | - | - | - | - | - | - | - | 0 | not used (always 0) |

The description of each status bit is as follows:

**Command Reject (bit 0)** - This bit indicates that an improper command has been received and ignored. If a port is defined as half-duplex, a command reject occurs when a write with subchannel monitoring, or, a read echoplex command is executed. When an undefined command occurs, a program violation is also indicated. Unit check status is returned when a program violation is encountered. Note, a busy condition, not a program violation, is generated if a read command is received during a write operation, or, a write command is received during a read operation while the controller is in the half-duplex mode.

**Equipment Check (bit 3)** - This bit indicates that an I/O device malfunction has occurred during the last operation. This bit is set when either the data set ready line, or, the carrier detect line goes low while the data terminal ready line is high (true). In full-duplex mode, an equipment check indication will occur if the clear to send line goes low while the RTS line is high. In either situation, the I/O operation is halted, unit check status is posted, and a unit check interrupt is issued.

**Data Check (bit 4)** - This bit indicates that either the data is in error, or, the data is lost. For this bit to be set, an overrun, parity, or framing error must have occurred during input. The I/O operation is halted, unit check status is posted, and a unit check interrupt is issued.

**Sense Byte 1: Line Status and Error Conditions**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | - | - | - | - | - | - | - | ASCII control character detected |
| | - | 1 | - | - | - | - | - | - | Special character detected |
| | - | - | 1 | - | - | - | - | - | ETX detected |
| | - | - | - | 1 | - | - | - | - | Break interrupt |
| | - | - | - | - | 1 | - | - | - | Framing error |
| | - | - | - | - | - | 1 | - | - | Parity error |
| | - | - | - | - | - | - | 1 | - | Overrun |
| | - | - | - | - | - | - | - | 1 | Wake-up character detected |

The description of each status bit is as follows:

**ASCII Control Character Detected (bit 0)** - This bit indicates that an ASCII control character (hexadecimal 00 through 7F) has been received during the last read operation. The ASCII control character detect modifier bit (bit 1) must be set in the read command byte to cause this status condition. All buffered data, including the ASCII control character, is sent to main memory. The CE/DE status is posted and a CE/DE interrupt is issued.

**Special Character Detected (bit 1)** - This bit indicates that the previously defined byte has been received during the last read operation. The special character detect modifier bit (bit 2) must be set in the read command byte to cause this status condition. All buffered data, including the special character, is sent to main memory. The CE/DE status is posted and a CE/DE interrupt is issued.

**ETX Detected (bit 2)** - This bit indicates that an ETX control character has been received, thus terminating the port's output. The only command which can cause this situation is a write with input subchannel monitoring command to a full-duplex port. Output is terminated immediately, CE/DE status is posted, and a CE/DE interrupt is issued.

**Break Interrupt (bit 3)** - This bit indicates that the data line has been held in the spacing state (logic state 0) for a period that is longer than a full character time period. If a read operation was in progress, the unit exception status is posted, and a unit exception interrupt is issued. Framing error (bit 4) is also set. If there were no I/O operations in progress, the attention status is posted. Only one break interrupt is accepted between each read operation.

**Framing Error (bit 4)** - This bit indicates at least one character in the last read operation did not have a valid stop bit. If this bit is set without the break interrupt bit, a unit check status will be posted, and a unit check interrupt will be issued.

**Parity Error (bit 5)** - This bit indicates that at least one character in the last read operation did not have the correct parity. When this occurs, the data check bit in sense byte 0 is set, unit check status is posted, and unit check interrupt is issued.

**Overrun (bit 6)** - This bit indicates that the internal 256 byte type ahead buffer has been overrun with data, or, that the data has been lost at the DUART controller. This condition occurs because data was not read fast enough from its holding register.

**Wake-up Character Detected (bit 7)** - This bit indicates that a wake-up character has been received when a read operation was not in progress. When this condition occurs, attention status is posted and an attention interrupt is issued.

**Sense Byte 2: Modem Status**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | - | - | - | - | - | - | - | Received line signal detect status |
| | - | 1 | - | - | - | - | - | - | Ring indicator line status |
| | - | - | 1 | - | - | - | - | - | Data set ready line status |
| | - | - | - | 1 | - | - | - | - | Clear to send line status |
| | - | - | - | - | 1 | - | - | - | Delta receive line signal detect |
| | - | - | - | - | - | 1 | - | - | Modem ring interrupt |
| | - | - | - | - | - | - | 1 | - | Delta data set ready |
| | - | - | - | - | - | - | - | 1 | Delta clear to send |

The description of each status bit is as follows:

**Received Line Signal Detect Status (bit 0)** - This bit indicates the logic condition of the RLSD input signal. When this bit is set, the signal is true.

**Ring Indicator Line Status (bit 1)** - This bit indicates the logic condition of the RI input signal. When this bit is set, the signal is true.

**Data Set Ready Line Status (bit 2)** - This bit indicates the logic condition of the DSR input signal. When this bit is set, the signal is true.

**Clear To Send Line Status (bit 3)** - This bit indicates the logic condition of the CTS input signal. When this bit is set, the signal is true.

**Delta Receive Line Signal Detect (bit 4)** - This bit indicates that the receive line signal detect (RLSD) input signal has gone low while a read operation was in progress. The read operation is terminated, unit check is posted, and a unit check interrupt is issued.

**Modem Ring Interrupt (bit 5)** - This bit indicates that the ring indicator (RI) signal has changed states (trailing edge only). If the modem ring enable bit is set during the initialization of byte 1, the controller will generate an attention status. The DTR line must then be set, by software, to answer the phone.

**Delta Data Set Ready (bit 6)** - This bit indicates that a transition occurred on the data set ready (DSR) input signal. If a read operation was in progress, it is terminated with unit check status posted and a unit check interrupt issued. If a read operation was not in progress and a DTR line is high, attention status is posted and an attention interrupt is issued.

**Delta Clear To Send (bit 7)** - This bit indicates that the clear to sent (CTS) input signal shifted from high to low. If a read operation was in progress (full-duplex mode), or, a write operation is in progress (half-duplex mode), the operation is terminated, unit check status is posted, and a unit check interrupt is issued.

**Sense Byte 3: Modem Control/Operation Mode**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | - | - | - | - | - | - | - | Half-duplex operation |
| | - | 1 | - | - | - | - | - | - | Modem ring enabled |
| | - | - | 1 | - | - | - | - | - | Line parameters defined |
| | - | - | - | 1 | - | - | - | - | Diagnostic mode set |
| | - | - | - | - | 1 | - | - | - | Port drivers disabled |
| | - | - | - | - | - | 0 | - | - | not used (always 0) |
| | - | - | - | - | - | - | 1 | - | Request to send (RTS) |
| | - | - | - | - | - | - | - | 1 | Data terminal ready (DTR) |

The description of each status bit is as follows:

**Half-duplex operation (bit 0)** - This bit indicates that the port is designated for half-duplex operation. If this bit is not set, the port is designated for full-duplex operation.

**Modem ring enabled (bit 1)** - This bit indicates that the port is designated for modem ring enabled. If this bit is set, the subchannel responds to a modem line ring interrupt. When this condition occurs, an attention status interrupt is issued. The DTR line is not set by the subchannel, it is the responsibility of software to set or reset the DTR signal.

**Line parameters defined (bit 2)** - This bit indicates that the subchannel DUART has been initialized with the control command form feed (FF).

**Diagnostic mode set (bit 3)** - This bit indicates that the diagnostic loop-back feature of the DUART is enabled and all data transmitted is received on the corresponding receive subchannel. Only data is looped-back, modem signals are not.

**Port drivers disabled (bit 4)** - This bit indicates that the port's drivers, data and modem control line RS-423, are in a disabled or high impedance mode. When this bit is set, the signal is true.

**Request to send (RTS) (bit 6)** - This bit indicates the logic condition of the request to send (RTS) output signal. When this bit is set, the signal is true.

**Data terminal ready (DTR) (bit 7)** - This bit indicates the logic condition of the data terminal ready (DTR) output signal. When this bit is set, the signal is true.

**Sense Byte 4: Line Parameters**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | - | - | - | - | - | - | - | not used (always 0) |
| | - | 0 | - | - | - | - | - | - | Break reset |
| | - | 1 | - | - | - | - | - | - | Break set |
| | - | - | 0 | - | - | - | - | - | Normal parity as defined |
| | - | - | 1 | - | - | - | - | - | Force parity to 1 if odd, 0 if even |
| | - | - | - | 0 | - | - | - | - | Odd parity |
| | - | - | - | 1 | - | - | - | - | Even parity |
| | - | - | - | - | 0 | - | - | - | Parity disabled |
| | - | - | - | - | 1 | - | - | - | Parity enabled |
| | - | - | - | - | - | 0 | - | - | One stop bit |
| | - | - | - | - | - | 1 | - | - | Two stop bits or (1 1/2 stop bits for five-bit characters) |
| | - | - | - | - | - | - | 0 | 0 | Five-bit character length |
| | - | - | - | - | - | - | 0 | 1 | Six-bit character length |
| | - | - | - | - | - | - | 1 | 0 | Seven-bit character length |
| | - | - | - | - | - | - | 1 | 1 | Eight-bit character length |

The description of each status bit is as follows:

**Break Set (bit 1)** - When this bit is set, the data output line is held in the spacing (logic 0) state.

**Forced Parity (bit 2)** - When this bit is set, the parity bit will always be a one for odd selected parity, or, a zero for even selected parity.

**Even/Odd Parity (bit 3)** - When this bit is set, even parity has been selected.

**Parity Enabled (bit 4)** - When this bit is set, the proper parity is generated during an output operation and checked during an input operation.

**Stop Bits (bit 5)** - When this bit is set, two stop bits are transmitted with each character, as well as, expected to receive two stop bits with every character. If the character length has been defined as five bits, then one and one half stop bits are transmitted and received. If this bit is not set, one stop bit is transmitted and received.

**Character Length (bits 6 and 7)** - These two bits indicate the character length for transmitted and received data.

**Sense Byte 4: Line Parameters**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | - | - | - | - | - | - | - | not used (always 0) |
| | - | 0 | - | - | - | - | - | - | not used (always 0) |
| | - | - | 0 | - | - | - | - | - | Allow wake-up character detection |
| | - | - | 1 | - | - | - | - | - | Inhibit wake-up character detection |
| | - | - | - | 0 | - | - | - | - | not used (always 0) |
| | - | - | - | - | 0 | 0 | 0 | 0 | 75 baud |
| | - | - | - | - | 0 | 0 | 0 | 1 | 110 baud |
| | - | - | - | - | 0 | 0 | 1 | 0 | 134.5 baud |
| | - | - | - | - | 0 | 0 | 1 | 1 | 150 baud |
| | - | - | - | - | 0 | 1 | 0 | 0 | 300 baud |
| | - | - | - | - | 0 | 1 | 0 | 1 | 600 baud |
| | - | - | - | - | 0 | 1 | 1 | 0 | 1,200 baud |
| | - | - | - | - | 0 | 1 | 1 | 1 | 2,000 baud |
| | - | - | - | - | 1 | 0 | 0 | 0 | 2,400 baud |
| | - | - | - | - | 1 | 0 | 0 | 1 | 4,800 baud |
| | - | - | - | - | 1 | 0 | 1 | 0 | 1,800 baud |
| | - | - | - | - | 1 | 0 | 1 | 1 | 9,600 baud |
| | - | - | - | - | 1 | 1 | 0 | 0 | 19.2K baud |
| | - | - | - | - | 1 | 1 | 0 | 1 | 38.4K baud |
| | - | - | - | - | 1 | 1 | 1 | 0 | External clock is 16X |
| | - | - | - | - | 1 | 1 | 1 | 1 | External clock is 1X |

The description of each status bit is as follows:

**Inhibit Wake-up Character Detection (bit 2)** - When this bit is set, the detection of the wake-up character is inhibited. If the bit is not set, the detection of the wake-up character is allowed.

**Baud Rate Selection (bits 4, 5, 6 and 7)** - These bits represent the baud rate selected for the asynchronous port. If the port has not been initialized, these bits will be 0.

**Sense Bytes 6 and 7:** These bytes represent the firmware identification and revision level.

### 6.5.3.3 Define Special Character

The define special character command causes a one-byte transfer of data to the controller. This transfer contains an eight-bit character that is used for input termination. More than one character can be defined for each subchannel. To delete a character, the define parameters command (FF) is reissued to delete all special characters.

#### 6.5.3.4 Reset Data Terminal Ready

The reset data terminal ready command resets the DTR signal for the addressed port. The DTR signal is automatically reset during a controller clear or power rest operation.

#### 6.5.3.5 Set Data Terminal Ready

The set data terminal ready command sets the DTR signal for the addressed port.

#### 6.5.3.6 Reset Request To Send

The reset request to send command resets the RTS signal for the addressed port. The RTS signal is automatically reset during a device clear, a controller clear, or a power reset operation. If the port is defined as half-duplex, this signal is automatically reset at the completion of a write command.

#### 6.5.3.7 Set Request To Send

The set request to send command sets the RTS signal for the addressed port. If the addressed port is defined as half-duplex, this signal is automatically set at the start of each write command.

**NOTE**

The subchannel will automatically control the signals RTS and DTR depending on how the ring enable and full/half-duplex are defined in the define parameter command (FF). If ring is enabled, the line is assumed to be remote (i.e., modem) and both signal are reset. If the ring is not enabled, the line is assumed to be local (i.e., local terminal), in this case, the DTR and RTS signals are set if the line is full-duplex. If the line is half-duplex, the RTS signal is reset.

#### 6.5.3.8 Reset Break

The reset break command is used to disable the break feature. When this occurs, it releases the serial output line from the clamped (logic 0) state.

#### 6.5.3.9 Set Break

The set break command is used to set the serial output line to the spacing (logic 0) state. This state is maintained until a reset break or define parameter command is received.

### 6.5.3.10 Enabled Port Drivers

The enable port drivers command is used to put the port's data and modem control line RS-423 drivers into an active mode.

### 6.5.3.11 Disable Port Drivers

The disable port drivers command is used to put the port's data and modem control line RS-423 drivers into an inactive or high impedance mode.

### 6.5.3.12 Transparent Flow Control

The transparent flow control command allows the MFP to monitor incoming data for user definable flow control characters and will automatically stop/start the transmission of data from the port when the stop/start characters are received.

The characters received are not modified in any way and are reported to the read subchannel untouched. This mode of operation is controlled through one command. The command (53 hex) requires three bytes of information each time it is executed. The first byte of information is the mode byte, the second byte is the start character, and the third byte is the stop character.

**Mode Byte (Byte 1):**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | - | - | - | - | - | - | - | Normal write |
| | 1 | - | - | - | - | - | - | - | Enable transparent flow control |
| | 1 | 0 | - | - | - | - | - | - | Restart on start character only |
| | 1 | 1 | - | - | - | - | - | - | Restart on non stop character |

**Start Character (Byte 2):**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | c | c | c | c | c | c | c | c | Start character |

**Stop Character (Byte 3):**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | c | c | c | c | c | c | c | c | Stop character |

When the most significant bit (bit 0) of the mode byte is set to one, the transparent flow control mode is enabled. Once enabled, each character is compared to the stop character. When the characters match, a write control flag is reset in the MFP's memory. This flag is checked by the write character routine before each character is sent out and if the flag is reset no write action is taken for that subchannel. If the incoming character is not the stop

character, then bit 1 of the mode byte is checked. If bit 1 is zero, then the incoming character is compared with the start character. When the characters match, then the write control flag is set in the MFP's memory which allows the output of characters to continue. If bit 1 of the mode byte is set and the incoming character is not the stop character, the flag is set unconditionally, allowing the write to start with the reception of any character other than the stop character.

If the start character and stop character are set equal to each other, then the write will be stopped the first time the character is received and started the second time the character is received. This allows a toggling off and on of the output data with a single character.

This mode of operation should work with any read or write commands that does not use the other subchannel (for example, non-usable commands would be read with echoplex, write with input subchannel monitoring, or read with flow control).

### 6.5.3.13 Define Parameters

The define parameters command proceeds a three-byte transfer which defines the port parameters. The parameters defined are: wake-up and half- or full-duplex operations. The parameters are defined by bytes 1, 2, and 3. These parameters are described in the sense status section of this manual.

**Define Parameters Byte 1:**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | - | - | - | - | - | - | - | Full-duplex operation |
| | 1 | - | - | - | - | - | - | - | Half-duplex operation |
| | - | 0 | - | - | - | - | - | - | Disable modem ring |
| | - | 1 | - | - | - | - | - | - | Enable modem ring |
| | - | - | 0 | - | - | - | - | - | Normal parity as defined |
| | - | - | 1 | - | - | - | - | - | Force parity to 1 if odd, 0 if even |
| | - | - | - | 0 | - | - | - | - | Odd parity |
| | - | - | - | 1 | - | - | - | - | Even parity |
| | - | - | - | - | 0 | - | - | - | Parity disabled |
| | - | - | - | - | 1 | - | - | - | Parity enabled |
| | - | - | - | - | - | 0 | - | - | One stop bit |
| | - | - | - | - | - | 1 | - | - | Two stop bits or (1 1/2 stop bits for five-bit characters) |
| | - | - | - | - | - | - | 0 | 0 | Five-bit character length |
| | - | - | - | - | - | - | 0 | 1 | Six-bit character length |
| | - | - | - | - | - | - | 1 | 0 | Seven-bit character length |
| | - | - | - | - | - | - | 1 | 1 | Eight-bit character length |

**Define Parameters Byte 2:**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | - | - | - | - | - | - | - | not used (always 0) |
| | - | 0 | - | - | - | - | - | - | not used (always 0) |
| | - | - | 0 | - | - | - | - | - | Allow wake-up character detection |
| | - | - | 1 | - | - | - | - | - | Inhibit wake-up character detection |
| | - | - | - | 0 | - | - | - | - | Reset diagnostic loop |
| | - | - | - | 1 | - | - | - | - | Set diagnostic loop |
| | - | - | - | - | 0 | 0 | 0 | 0 | 75 baud |
| | - | - | - | - | 0 | 0 | 0 | 1 | 110 baud |
| | - | - | - | - | 0 | 0 | 1 | 0 | 134.5 baud |
| | - | - | - | - | 0 | 0 | 1 | 1 | 150 baud |
| | - | - | - | - | 0 | 1 | 0 | 0 | 300 baud |
| | - | - | - | - | 0 | 1 | 0 | 1 | 600 baud |
| | - | - | - | - | 0 | 1 | 1 | 0 | 1,200 baud |
| | - | - | - | - | 0 | 1 | 1 | 1 | 2,000 baud |
| | - | - | - | - | 1 | 0 | 0 | 0 | 2,400 baud |
| | - | - | - | - | 1 | 0 | 0 | 1 | 4,800 baud |
| | - | - | - | - | 1 | 0 | 1 | 0 | 1,800 baud |
| | - | - | - | - | 1 | 0 | 1 | 1 | 9,600 baud |
| | - | - | - | - | 1 | 1 | 0 | 0 | 19.2K baud |
| | - | - | - | - | 1 | 1 | 0 | 1 | 38.4K baud |
| | - | - | - | - | 1 | 1 | 1 | 0 | External clock is 16X |
| | - | - | - | - | 1 | 1 | 1 | 1 | External clock is 1X |

**Define Parameters Byte 3:**

| Bits- | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Definition |
|---|---|---|---|---|---|---|---|---|---|
| | c | c | c | c | c | c | c | c | Eight-bit wake-up character |

## 6.6 Expansion Port

The MFP's expansion port is an additional I/O channel that is built into the MFP. This interface supports class F protocol as previously specified. The specific IOCD commands are dependent upon the implementation of the controller.

If the MFP channel receives an I/O instruction for a subaddress in the range of X"80" through X"DF", or, X"F0 through X"F7", it will be directed to the expansion port. Note that interrupt instructions are channel NOT controller related; therefore, they will function even if there are no expansion controllers attached to the expansion port addressed. If a controller circuit card is not attached to the expansion port, the I/O instruction will not be completed, nor will status be returned. For a complete description, refer to the specific MFP expansion port technical manual.

# INDEX

**Gould Inc., Computer Systems Division**
6901 W. Sunrise Blvd.
P. O. Box 409148
Fort Lauderdale, FL 33340-9148
Telephone (305) 587-2900

GOULD
*Electronics*

## Users Group Membership Application

USER ORGANIZATION: _____

REPRESENTATIVE(S): _____

_____

_____

ADDRESS: _____

_____

TELEX NUMBER: _____ PHONE NUMBER: _____

NUMBER AND TYPE OF GOULD CSD COMPUTERS: _____

_____

OPERATING SYSTEM AND REV. LEVEL: _____

_____

APPLICATIONS (Please Indicate)

1. EDP
   - A. Inventory Control
   - B. Engineering & Production Data Control
   - C. Large Machine Off-Load
   - D. Remote Batch Terminal
   - E. Other

2. Communications
   - A. Telephone System Monitoring
   - B. Front End Processors
   - C. Message Switching
   - D. Other

3. Design & Drafting
   - A. Electrical
   - B. Mechanical
   - C. Architectural
   - D. Cartography
   - E. Image Processing
   - F. Other

4. Industrial Automation
   - A. Continuous Process Control Op.
   - B. Production Scheduling & Control
   - C. Process Planning
   - D. Numerical Control
   - E. Other

5. Laboratory and Computational
   - A. Seismic
   - B. Scientific Calculation
   - C. Experiment Monitoring
   - D. Mathematical Modeling
   - E. Signal Processing
   - F. Other

6. Energy Monitoring & Control
   - A. Power Generation
   - B. Power Distribution
   - C. Environmental Control
   - D. Meter Monitoring
   - E. Other

7. Simulation
   - A. Flight Simulators
   - B. Power Plant Simulators
   - C. Electronic Warfare
   - D. Other

8. Other

Please return to:
Users Group Representative

Date: _____

## Gould Inc., Computer Systems Division Users Group. . .

The purpose of the Gould CSD Users Group is to help create better User/User and User/Gould CSD communications.

There is no fee to join the Users Group. Simply complete the Membership Application on the reverse side and mail to the Users Group Representative. You will automatically receive Users Group Newsletters, Referral Guide and other pertinent Users Group activity information.

Fold and Staple for Mailing

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

**BUSINESS REPLY MAIL**

FIRST-CLASS MAIL PERMIT NO. 947 FT. LAUDERDALE, FL

POSTAGE WILL BE PAID BY ADDRESSEE

**GOULD INC., COMPUTER SYSTEMS DIVISION**
ATTENTION: USERS GROUP REPRESENTATIVE
6901 W. SUNRISE BLVD.
P. O. BOX 409148
FT. LAUDERDALE FL 33340-9970

Fold and Staple for Mailing

(Detach Here)

GOULD
*Electronics*