

JPAD

ASYNCHRONOUS MULTIPLEXER REFERENCE MANUAL

53778-91

JANUARY 1979



Engineering Reference Number
91-53778-00A0

NAKED MINI Division
18651 Von Karman, Irvine, CA 92713
tel. (714) 833-8830 TWX: (910) 595-1767

© 1979 Computer Automation, Inc.
Printed in U.S.A.

REVISION HISTORY

<u>Revision</u>	<u>Issue Date</u>	<u>Comments</u>
A0	January 1979	Original Issue

PREFACE

The Asynchronous Multiplexer Reference Manual describes the Asynchronous Multiplexer used to interface devices using asynchronous communications procedures and operating according to RS-232-C to NAKED MINI 4 Family computers.

The manual contains functional, programming and installation information. The programming section describes the NAKED MINI 4 instructions and commands used to communicate with the Asynchronous Multiplexer. The installation section includes information on the basic system configuration, and the procedures to establish, in hardware, the device address and interrupt address settings. Appendix A describes the function and installation of the RS-232-C Interface Panel, used to interface the Asynchronous Multiplexer to modems and peripherals operating according to EIA RS-232-C.

The Asynchronous Multiplexer Reference Manual assumes familiarity with the applicable Computer Automation Inc., NAKED MINI 4 Family computer reference manual and the MAXI-BUS Interface. The information in this manual corresponds to the following products:

Asynchronous Multiplexer, 4-Channel, Marketing Number 45060-04

Asynchronous Controller Assembly Number 73-53778-04
Interface Panel Assembly Number 73-53768-04
Multiplexer-Panel Cable Assembly Number 72-21156-06

Asynchronous Multiplexer, 8-Channel, Marketing Number 45060-08

Asynchronous Controller Assembly Number 73-53778-08
Interface Panel Assembly Number 73-53768-08
Multiplexer-Panel Cable Assembly Number 72-21156-06

Mandatory Documents (as applicable)

20990-91 LSI 4/10 Computer Reference Manual
20991-91 LSI 4/30 Computer Reference Manual
20945-91 LSI 4/90 Computer Reference Manual
EIA RS-232-C

CONTENTS

SECTION 1	GENERAL DESCRIPTION	1-1
	1.1 SYSTEM DESCRIPTION	1-1
	1.2 SPECIFICATIONS	1-3
SECTION 2	FUNCTIONAL DESCRIPTION	2-1
	2.1 FUNCTIONAL SYSTEM	2-1
	2.1.1 MAXI-BUS Interface Control	2-1
	2.1.2 Interrupt Section	2-2
	2.1.3 Busy and Select Registers	2-3
	2.1.4 Command and Input RAM	2-3
	2.1.5 Status and Output RAM	2-3
	2.1.6 Modem Control Register	2-4
	2.1.7 Channels 0-7	2-4
	2.1.8 Option Tables	2-4
	2.2 RECEIVE MESSAGE SEQUENCE	2-10
	2.3 TRANSMIT MESSAGE SEQUENCE	2-12
	2.4 AUTO ANSWER	2-12
SECTION 3	PROGRAMMING INSTRUCTION SET	3-1
	3.1 DEVICE AND INTERRUPT VECTOR ADDRESSES	3-1
	3.1.1 Device Address	3-1
	3.1.2 Interrupt Vectoring	3-2
	3.1.3 Interrupt Addresses	3-4
	3.2 SELECT AND CHANNEL I/O INSTRUCTIONS	3-4
	3.2.1 Select and Present Instructions (SELP)	3-4
	3.2.2 Channel I/O Instructions	3-5
	3.3 END-OF-BLOCK INTERRUPT	3-7
	3.4 UPDATE INTERRUPT STATUS (UIS) INSTRUCTIONS	3-7
	3.5 COMMAND WORD	3-7
	3.5.1 Reset Receive Half-Channel (:8000)	3-7
	3.5.2 Load Receive Command Field (:4n00)	3-8
	3.5.3 Reset Transmit Half-Channel (:2000)	3-9
	3.5.4 Load Transmit Command Field (:10nn)	3-10
	3.6 STATUS	3-12
	3.6.1 Receiver Status Field	3-12
	3.6.2 Transmit Status Field	3-14

SECTION 4	INSTALLATION	4-1
4.1	SYSTEM CONFIGURATION	4-1
4.2	ADDRESS ASSIGNMENTS	4-1
4.2.1	Device Address Assignments	4-1
4.2.2	Interrupt Vector Address	4-3

APPENDIXES

A	RS-23C-C INTERFACE PANEL	A-1
---	------------------------------------	-----

FIGURES

<u>Figure</u>		<u>Page</u>
1-1	Asynchronous Multiplexer System Block Diagram	1-2
2-1	Asynchronous Multiplexer Functional Block Diagram	2-2
2-2	Option Table Byte 0 Format	2-5
2-3	Option Table Byte 1 Format	2-6
2-4	Option Table Byte 2 Format	2-7
2-5	Option Table Byte 3 Format	2-8
2-6	Option Table Byte 4 Format	2-9
2-7	Receive Message Flowchart	2-11
2-8	Transmit Message Flowchart	2-13
3-1	Interrupt Vector Addresses	3-3
3-2	Receiver Status Register	3-8
3-3	Load Transmit Command Field Command Format	3-10
3-4	Data Field Format for Write Modem Control Register Command	3-10
3-5	Status Register Format	3-12
3-6	Receiver Status Field Format	3-12
3-7	Transmit Status Field Format	3-14
4-1	Asynchronous Multiplexer Board	4-2
4-2	Device Address Bit Assignments	4-3
4-3	Interrupt Vector Format	4-4

TABLES

3-1	Receive Error Code	3-13
3-2	Transmit Error Code	3-14
4-1	Device Address Switch Settings	4-2

SECTION 1

GENERAL DESCRIPTION

The Asynchronous Multiplexer is a 4- or 8-channel communications controller designed to interface any Computer Automation, Inc., NAKED MINI 4 Family computer to a maximum of eight terminals or peripheral devices. The controller is a half-card device connected by a 6-foot cable to a separate, rack mountable, 4- or 8-channel asynchronous interface panel. The interface panel provides RS-232-C (CCITT V.24)) compatible drivers and receivers. Interface panel channels may be connected directly to a given device or modem, depending on the specific system requirements. Any combination of terminals, modems, or peripherals may be attached to the interface panel; however, the attached devices must be RS-232-C compatible.

1.1 SYSTEM DESCRIPTION

The Asynchronous Multiplexer provides a means of data transfer between the CPU and up to eight terminals or peripherals on a priority basis. Channels are assigned a number from 0-7, with channel priority decreasing as the channel number increases. The eight channels can be individually programmed for full- or half-duplex operation. Configurations of up to 120 channels-using 15 multiplexer boards and 15, 8-channel RS-232-C interface panels are possible. Figure 1-1 shows the system block diagram.

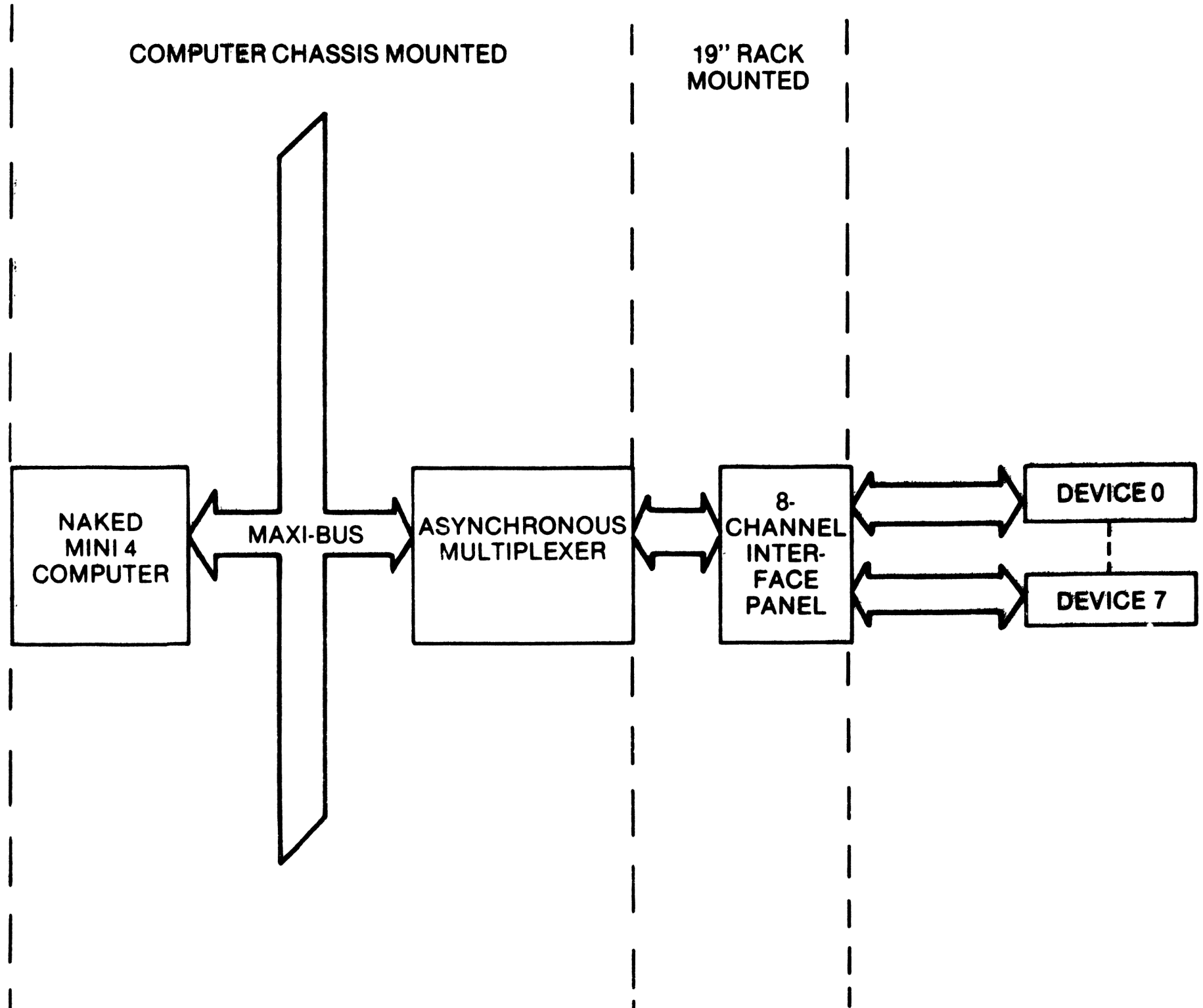


Figure 1-1. Asynchronous Multiplexer System Block Diagram

1.2 SPECIFICATIONS

Electrical Logic Levels

Device Interface RS-232-C/CCITT V.24

Transmission Rates

Maximum Single Channel Baudrate 19.2K baud
Maximum Throughput, Asynchronous Multiplexer . . . 2K character/second

NOTE

2K characters/second is the maximum throughput that the Asynchronous Multiplexer Board can accomplish regardless of the channel baud rates chosen. The character rate is independent of the length of the characters and the number of channels in operation.

Programmable Baud Rates	50	150	1800	4800
	75	300	2000	7200
	110	600	2400	9600
Environment	134.5	1200	3600	19200

Operating Temperature Range (Degrees Celsius) . . . 0 to +50
Non-operating Temperature Range (Degrees Celcius) . -40 to +80
Relative Humidity (Percent, Non-condensing) 5 to 90

Physical

Length 16.88 in. (42.88 cm.)
Width 7.5 in. (19.05 cm.)

Modem Panel Interface Connector 34-Conductor Cable

Cable Length

Asynchronous Multitplexer to Interface Panel 1.83 m (6 ft)

Device Mating Connector

RS-232-C TRW Cinch DB25P
EIA D-type, 25 pin.

Device Mating Connector

Device Address Range :1 to :F
Computer Automation Software Compatible Address . . :7
Interrupt Address Range 0 to :3F80
Computer Automation Software Compatible Address . . :8000 to :BF80
:0180

Electrical

Input Voltage Regulation (All Voltages) ±5 percent

Power Consumption (Maximum)

	8-Channel			4-Channel		
	+12V	-12V	+5V	+12V	-12V	+5V
Asynchronous Multiplexer	-	-	2.0A	-	-	1.25A
53768 RS-232-C Interface Panel	0.2A	0.2A	0.5A	0.1A	0.1A	0.3A

SECTION 2

FUNCTIONAL DESCRIPTION

The Asynchronous Multiplexer is a half-card device that communicates with the CPU via the MAXI-BUS interface. Data transfer is by interrupt initiated Automatic I/O. The major functions of the multiplexer include:

- Prioritized communications over eight full-or half-duplex channels.
- Serial-to-parallel conversion of data received from a peripheral device and transmission of the converted characters to the CPU.
- Parallel-to-serial conversion of control and data characters from the CPU into a serial data stream and transmitting the characters to a peripheral device (where this is a device requirement).

To facilitate accomplishing these functions, the Multiplexer accepts and stores control information from the CPU which determines Multiplexer operating parameters. The functional block diagram for the Asynchronous Multiplexer is shown in Figure 2-1.

2.1 FUNCTIONAL SYSTEM

The operation of each Asynchronous Multiplexer channel is sequenced under microprocessor control, initiated by computer command. Once begun, these operational sequences take control of channel peripheral I/O, freeing the CPU for other operations. Asynchronous Multiplexer channels function independently, only relative priority affecting the functioning of one channel with respect to the others. The only board conditions affecting all channels are the enabling and disabling of board interrupts, board reset and reset interrupts.

2.1.1 MAXI-BUS Interface Control

Before any Multiplexer channel can operate, the CPU must be able to address appropriate commands to the Multiplexer board and the individual channels. This operation is achieved through the MAXI-BUS interface control logic. The MAXI-BUS control circuits compare addresses received on the MAXI-BUS interface address bus with the Multiplexer base device address. CPU access to the board is established when the addresses match. Once access is established, the control circuits decode the instruction and function code. If the instruction is for an I/O operation, the control circuits perform interrupt and information transfer control and timing for the designated channel. Selection of the Multiplexer base device address is explained in the installation section.

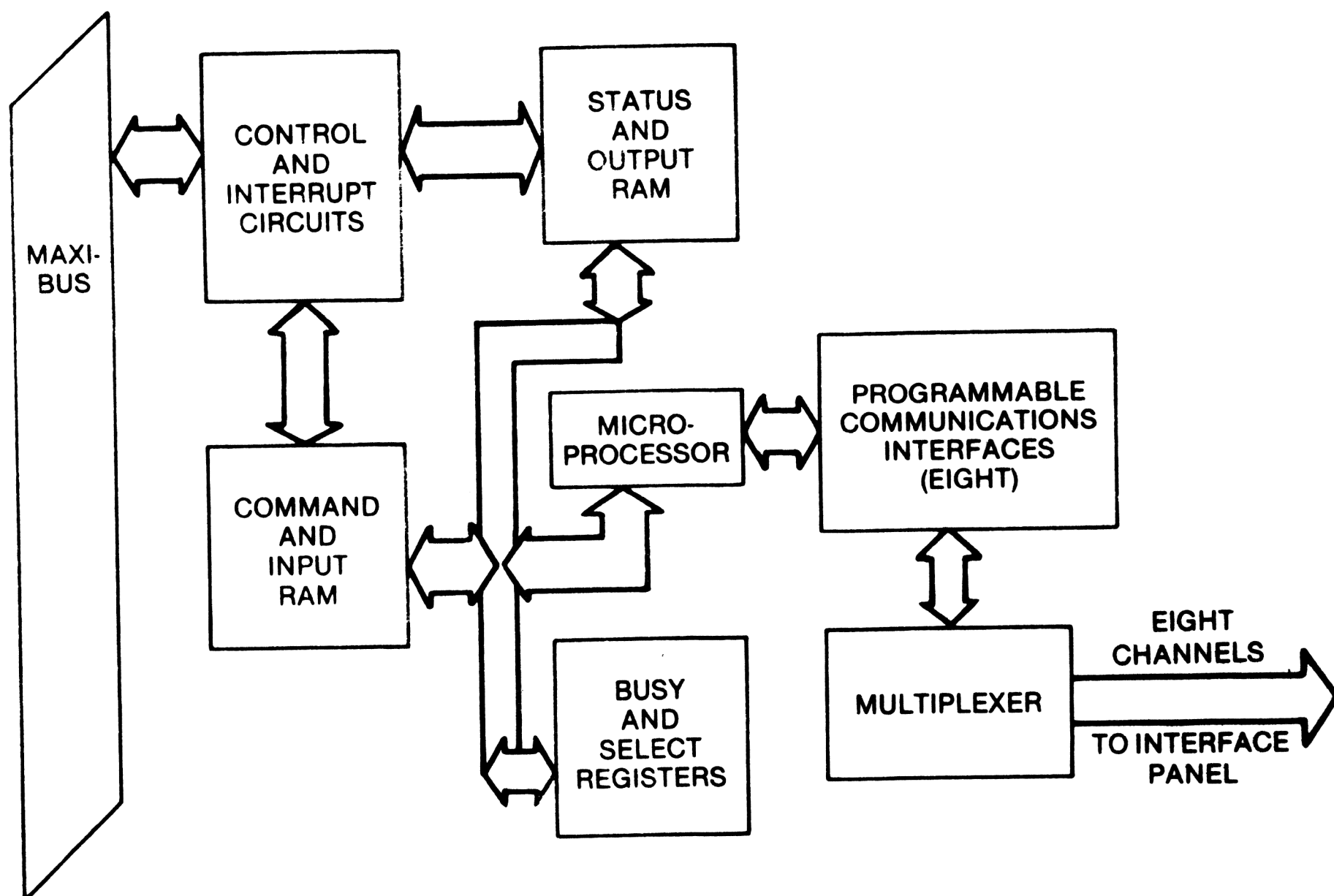


Figure 2-1. Asynchronous Multiplexer Functional Block Diagram

2.1.2 Interrupt Section

The interrupt section generates the interrupts for data service and End-of-Block service routines. To initiate data transfer, an interrupt for data service is sent to the CPU. When the interrupt is recognized the interrupt section sends a vector to the CPU giving the address of the memory location containing the Automatic I/O instruction block for data service. Data is then transferred between the Multiplexer and the system memory. When a buffer in system memory empties or fills, the Automatic I/O instruction causes the CPU to return an ECHO signal to the Multiplexer. The interrupt section responds by issuing an interrupt for EOB service. When the CPU recognizes the interrupt, the interrupt section sends a vector to the CPU with the address of the first word of the EOB instruction block in memory. Channel status is read as part of the EOB routine. Data service and EOB interrupts can be selectively disabled using CPU status word group interrupt inhibit bits 12 and 13, respectively. Disabling the EOB interrupt may block all further data service requests since a single interrupt flip-flop is used to generate both types of interrupt.

2.1.3 Busy and Select Registers

These registers contain a total of 32 bits: four bits per channel, one busy and one select bit per half channel. The bits are designated Receive Select (RXSEL), Receive Busy (RXBSY), Transmit Select (TXSEL), and Transmit Busy (TXBSY). The channel select bits are set by a command word with the corresponding Transmit or Receive Select bit set. Setting a select bit clears all message states on the half-channel to which it corresponds and disables the half-channel receiver or transmitter, making the half-channel ready to receive a new command.

The channel RXBSY or TXBSY bits are set when a channel Receive or Transmit command is loaded, and cleared when an EOB interrupt for the command is acknowledged. A Busy bit is immediately cleared when the corresponding select bit is set, if a command is not being loaded at the same time.

The BUSY bits are part of the channel status word and can be read one channel at a time by a CPU input status instruction. BUSY indicates the validity of the portion of the status word (transmit or receive) it refers to. If a BUSY bit is set, the corresponding section of the status word is disregarded. When the BUSY bit is cleared, the corresponding portion of the status word becomes valid following an EOB interrupt (3.2.2).

2.1.4 Input RAM

Commands to the multiplexer and data for transmission are loaded into the Input RAM. The RAM is divided into two registers. Transmit, eight words by eight bits, and Receive, eight words by four bits. The wider transmit word allows more transmit commands than receive, and the use of the transmit command register as an input data register. The RAM can be loaded 4, 8, or 12 bits at a time by the MAXI-BUS control and can be read 4 or 8 bits at a time by the microprocessor.

2.1.5 Output RAM

Received data and status information are read by the CPU from the Output RAM.

The RAM is divided into transmit and receive registers, and is used for status or for data. The Receive register is eight words by eight bits, while the Transmit register is eight words by four bits. Owing to its greater width, the Receive register can store more status information than the Transmit register, and is also used as an output data register.

The Output RAM may be written by the microprocessor four or eight bits at a time, but is read onto the MAXI-BUS eight or twelve bits at a time. The channel BUSY bits are read with status to comprise a fourteen-bit status word.

2.1.6 Modem Control Register

There are two 8 x 4-bit register files used for modem control: a command register file and a receive register file. The command register file is written only and the receive register file is read only. There is one register for each direction for each of the eight channels for a total of 16 registers. The state of the bits in the modem control registers is used to establish communication between the CPU and the attached modem.

2.1.7 Channels 0-7

Channels are assigned priorities, the priority decreasing as the channel number increases. The CPU receives data from and transmits to the Multiplexer on a channel priority basis. The receive operation takes priority over transmit. The channels are polled in sequence to determine if there is a receive operation pending. If there is, it is executed and the receive lines repolled. The transmit lines are only polled when no receive is pending. Each channel has a Programmable Communications Interface (PCI). The PCI can be programmed for the number of stop bits, full-or half duplex mode, character length, baud rates, and parity.

2.1.8 Option Tables

Each channel has a five 8-bit byte option table associated with it. The first four bytes of each table are programmed to establish channel configuration and PCI operating parameters. A Read or Write operation may be performed with these bytes. The fifth byte is the PCI status byte, and may be Read only.

The individual bytes are explained in the following sections. A bit default value is the value assumed unless the bit has been changed to a different value by programming or status conditions. Bit default values are assumed after a board reset.

2.1.8.1 Byte 0-Channel Configuration

Programming this byte establishes channel operating characteristics, the most important of which is the mode, full or half-duplex. The meaning and setting of all bits is explained below. The format for byte 0 is shown in Figure 2-2.

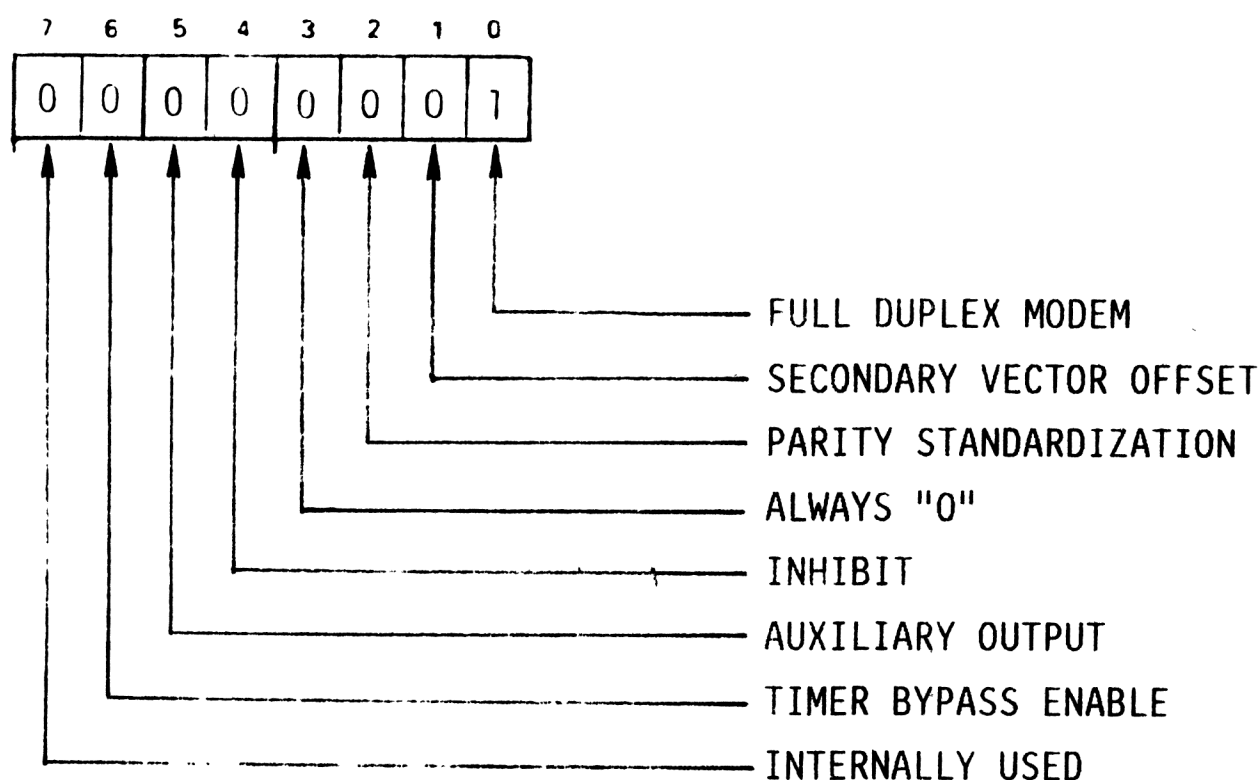


Figure 2-2. Option Table Byte 0 Format

Bit 0, the channel modem mode bit, determines whether the modem is in full- or half-duplex mode. Bit 0 = 1 gives full duplex. This is the setting normally used. A modem may be configured in full- or half-duplex independent of the channel configuration.

Bit 1 is the off-set vector bit. When bit 1 = 1, the secondary (receive) interrupt vectors are off-set by 64, for use when the channel is in full-duplex mode. The use of secondary vectors is explained in 3.1.2.

Bit 2, the parity standardization bit, determines parity when 7-bit characters are used. Bit 2 = 1 gives a parity standardization of one. Bit 2 = 0 gives a parity standardization of zero, which is used with NAKED MINI Family computer software.

Bit 3 is always zero.

Bit 4, the inhibit bit, inhibits automatic line setup, PCI status and error checking when set to 1. Bit 4 = 0 is the normal setting for this bit. If the bit is set to 1, modem controls must be used for line configuration.

Bit 5 controls an output available for whatever purpose the user desires. It is programmed using transmit commands.

Bit 6, is the timer bypass bit. When bit 6 = 1, the 10 ms internal timer normally enabled when the option table is read, is bypassed.

Bit 7 is reserved. Programming this bit has no effect because the bit setting is internally controlled.

2.1.8.2 Byte 1 - PCI Mode Register 1

Programming this byte establishes the number of bits per character, parity, and the number of stop bits. The meaning and setting of all bits is explained below. The format for byte 1 is shown in Figure 2-3. Values shown in the bit chart are default values.

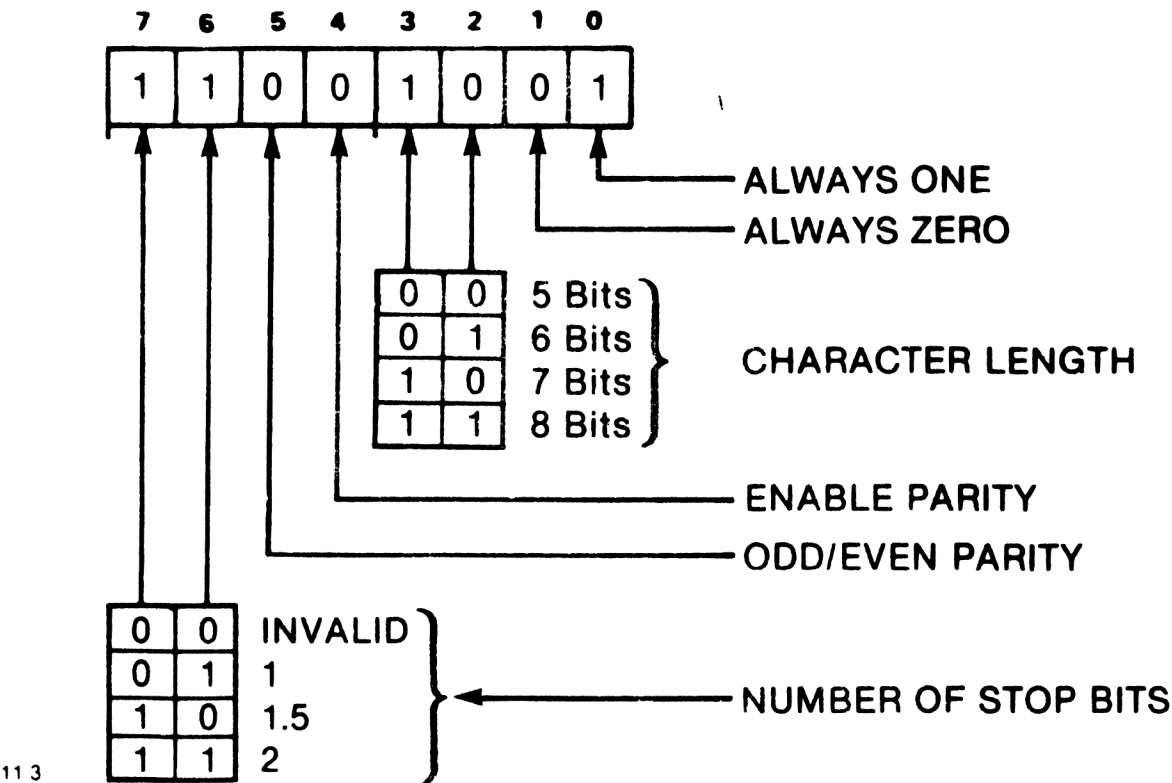


Figure 2-3. Option Table Byte 1 Format

Bit 0 is always one.

Bit 1 is always zero.

Bit 2 and 3 are the character length bits. Values for all combinations of bit settings are given in the chart. Normally the character length is seven bits, which means bit 2 = 0 and bit 3 = 1.

Bit 4 is the parity bit. When bit 4 = 1, parity is enabled.

Bit 5 is the parity type bit. This bit is valid only when parity is enabled (bit 4 = 1). When bit 5 = 1, parity is even. When bit 5 = 0, parity is odd.

Bits 6 and 7 are the stop bits. Values for all combinations of bit settings are given in the chart. Normally the number of stop bits is 2, which means bits 6 and 7 both equal one.

2.1.7.3 Byte 2 - PCI Mode Register 2

Programming byte 2 determines the baud rate. The bit meanings and settings are explained below. The format for byte 2 is shown in Figure 2-3. The values shown in the bit chart are default values.

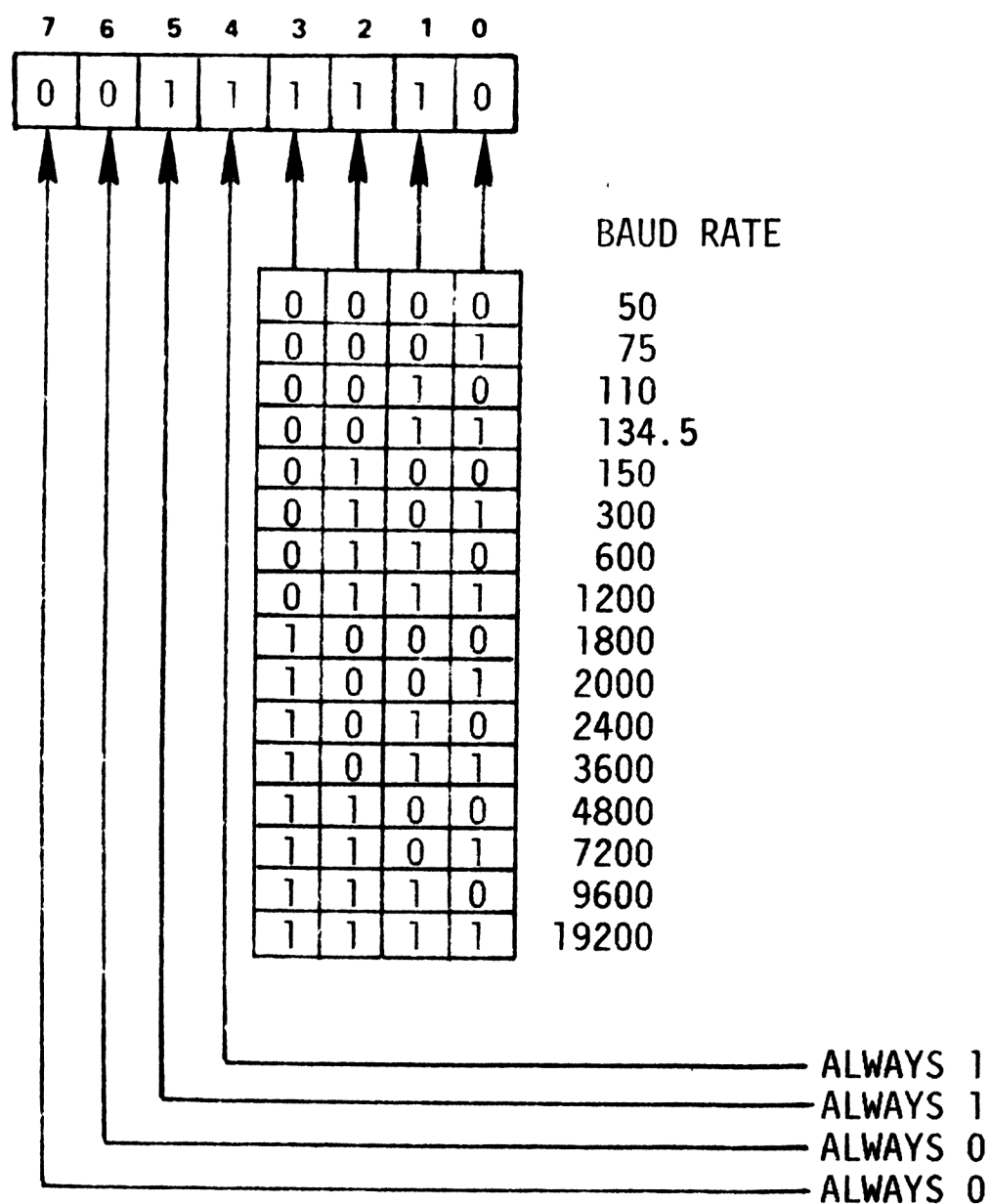


Figure 2-4. Option Table Byte 2 Format

Bits 0-3 determine the baud rate. The default value corresponds to a rate of 9600 baud.

Bits 4-7 always take the values shown.

2.1.8.4 Byte 3 - PCI Command Register

Programming byte 3 controls transmitter and receiver operation and associated functions. The bit meanings and settings are explained below. The format for byte 3 is shown in Figure 2-5. The values shown in the bit chart are default values.

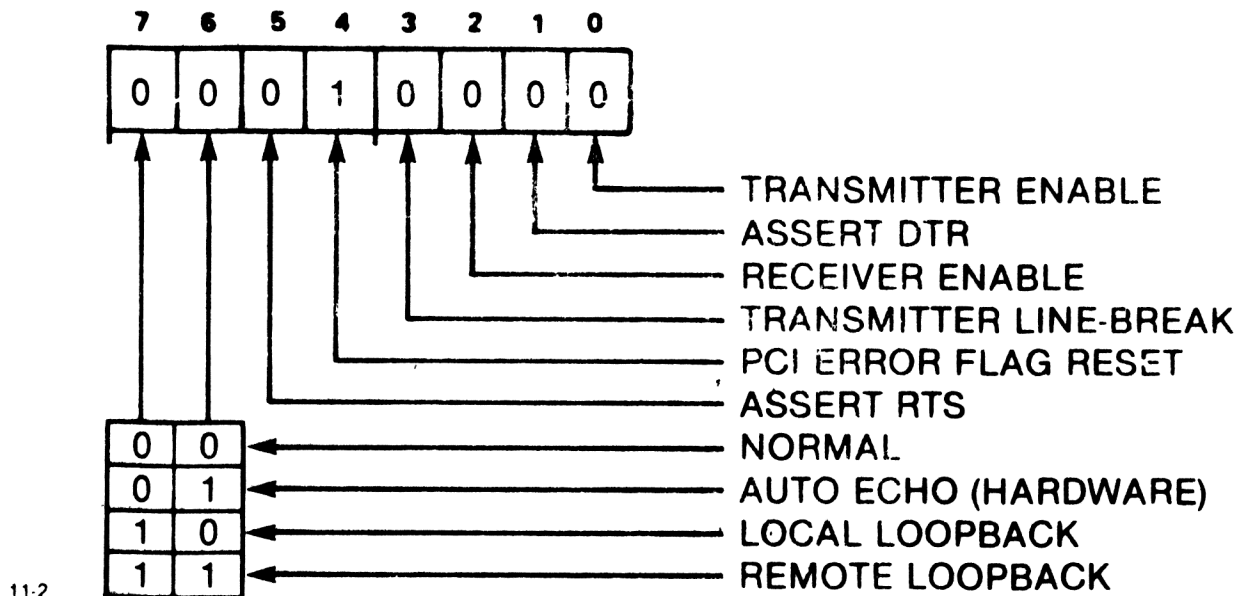


Figure 2-5. Option Table Byte 3 Format

Bit 0 is the transmitter enable bit. When bit 0 = 1, the transmitter is enabled.

Bit 1 is the Data Terminal Ready (DTR) bit. When bit 0 = 1, DTR is asserted.

Bit 2 is the receiver enable bit. When Bit 2 = 0, the receiver is enabled.

Bit 3 is the line-break bit. When bit 3 = 1, a transmitter line break is forced.

Bit 4 is the reset error bit. When bit 4 = 1, the error flags in the PCI status byte are reset.

Bit 5 is the Request To Send (RTS) bit. When bit 5 = 1, RTS is asserted.

Bits 6 and 7 determine the mode and the meanings of all combinations of bit settings are given in the chart.

- In normal mode the transmitter and receiver operate independently as determined by mode and status registers.
- Auto Echo mode is a hardware setting which disables the CPU-transmitter link while permitting normal CPU-receiver operation to continue. The echoed received data is automatically directed to the PCI transmit line.
- Local loopback allows testing of the PCI internal circuitry by connecting transmitter output to receiver input.
- Remote loopback allows testing of the external channel by placing data assembled by the receiver in the transmitter holding register and retransmitting.

2.1.8.5 Byte 4 - PCI Status Register

This byte may be Read only. The bit meanings are explained below. The byte 4 format is shown in Figure 2-6.

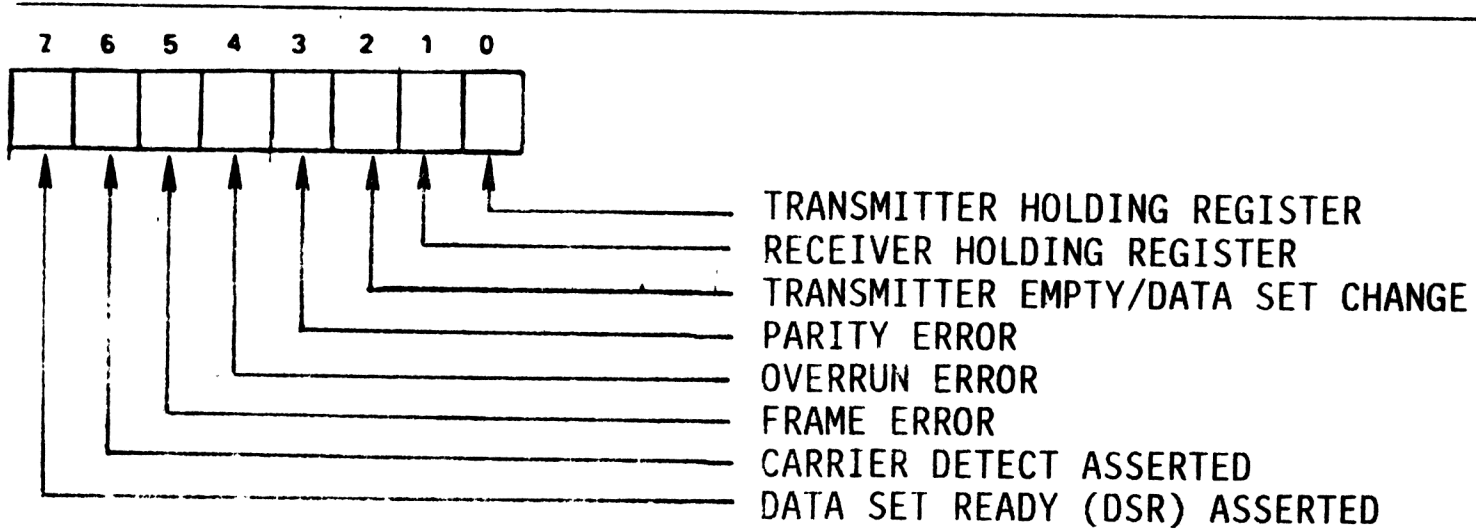


Figure 2-6. Option Table Byte 4 Format

Bit 0, the transmitter holding register bit, is valid only when the transmitter is enabled. When bit 0 = 0, the transmitter holding register has been loaded, but transmission has not taken place. When bit 0 = 1, the holding register is ready to accept data from the system.

Bit 1, the receiver holding register bit, is valid only when the receiver is enabled. When bit 1 = 1, a character has been loaded into the receiver holding register and can be read by the CPU. After a Read operation, bit 1 = 0 if no new character has been loaded.

Bit 2 = 1 to indicate one of three situations: 1) The transmitter shift register is empty following transmission and no new character is ready in the transmit holding register; 2) Data Set Ready or 3) Carrier Detect has changed state.

Bit 3, the parity error bit, is valid only when the enable parity bit has been set (bit 4 in byte 1 = 1). When bit 3 = 1, a parity error has been received. Bit 3 is cleared by disabling the receiver or by setting the reset error bit (bit 4 = 1 in byte 3).

Bit 4 is the overrun error bit. Bit 4 = 1 when the receiver holding register received a new character before the previous character was read by the CPU. This bit is cleared by disabling the receiver or by setting the reset error bit in byte 3.

Bit 5 is the framing error bit. When bit 5 = 1, the received character was not framed by the programmed number of stop bits. (If 1.5 stop bits are programmed, only the first stop bit is checked.) This bit is cleared by disabling the receiver or by setting the reset error bit in byte 3. Line Break also sets this bit to 1.

Bit 6 is the Carrier Detect indicator bit. When Carrier Detect is asserted, bit 6 = 1.

Bit 7 is the Data Set Ready indicator bit. When Data Set Ready is asserted, bit 7 = 1.

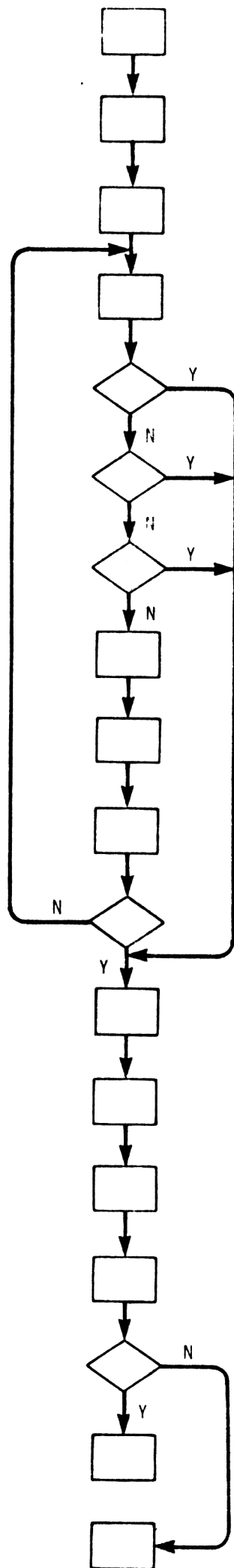
2.2 RECEIVE MESSAGE SEQUENCE

The receive message sequence is initiated by issuing the Receive Buffer command. This command is not a part of the computer instruction set, but must be configured in a specified register. The Receive Buffer command word is output by a Channel I/O instruction to the Multiplexer board and addressed channel (3.2.2).

The Receive Buffer command initializes the Multiplexer Input sequence for the addressed channel. The channel instructions establish communication with the channel modem by raising the Data Terminal Ready (DTR) line. The modem responds by raising the Data Set Ready (DSR) line. The Multiplexer can receive data when the Carrier Detect (CD) bit is 1 in the modem control register, indicating that a communication link with the polled station has been established. The Receive Buffer command also sets the Receiver Enable bit in the PCI command register.

If the polled station has data to transmit, the data is sent over the communications line and loaded in the Programmable Channel Interface (PCI) receiver holding register. The Receive Ready (RXRDY) line is tested to determine if data is present in the PCI receiver holding register. If there is, the microprocessor reads the data and the PCI status. If an error is indicated in the PCI status byte the receive operation is terminated and an End-Of-Block (EOB) interrupt is issued. An EOB interrupt is also issued if a special character or range match is detected, if either or both of these options is enabled. If none of these conditions are detected, a data service interrupt is issued to the CPU. When the interrupt is recognized the address vector for the data service routine is sent to the CPU. The data is then transferred to memory using Automatic I/O, one byte per interrupt. When the transfer byte count reaches zero, the CPU signals the Multiplexer by sending an ECHO signal. The Multiplexer then issues an interrupt to the CPU for an EOB routine. When the interrupt is recognized, the address vector for the EOB routine is sent to the CPU. The EOB routine is executed and channel status is read as part of the routine. If no error is indicated in status, the receive message sequence is completed. If an error occurred, a retransmission of the data must be requested.

The above description of the receive message sequence applies only if the Inhibit bit is not set (bit 4 in option table byte 0 is not equal to 1). If the bit is set, the microprocessor controlled modem initialization sequence, PCI status and error checking are disabled. Appropriate Write Option Table, Read Modem Register and Write Modem Register commands must be issued to assert DTR, RTS, check for DSR, and enable receiver and transmitter (3.5.2 and 3.5.4). Figure 2-7 shows the flowchart for the receive message sequence.



Load command word with Receive Buffer code in specified register.

Output command word with I/O instruction having required device address and function code of 1.

Receive Buffer code initializes the channel microprocessor controlled I/O sequence. If the peripheral has data to transfer, the data is loaded in the PCI receiver holding register.

Microprocessor reads data and status.

Is an error recorded in the PCI status register?

Is character match detected (if this option is enabled)?

Is range match detected (if this option is enabled)?

A data service interrupt is issued to the CPU.

When the interrupt is recognized, the address vector for the data service routine is issued to the CPU.

The data is transferred using Automatic I/O.

Was a byte count zero (ECHO) signal received from the CPU?

Channel status is read into the Output RAM.

An End-of-Block interrupt is issued to the CPU.

When the interrupt is recognized, an address vector for the End-of-Block routine is issued to the CPU.

The End-of-Block routine is executed. Channel status is read as part of this routine.

Was an error detected?

Request retransmission.

Receive Buffer sequence completed.

Figure 2-7. Receive Message Flowchart

2.3 TRANSMIT MESSAGE SEQUENCE

The transmit buffer sequence is initiated by issuing the Transmit Buffer command. This command is not a part of the computer instruction set but must be configured in a specified register. The Transmit Buffer command is output by a Channel I/O instruction to the Multiplexer board and addressed channel (3.2.2).

The Transmit Buffer command initializes the Multiplexer Output sequence for the addressed channel. The channel instructions establish communication with the channel modem by raising the Data-Terminal-Ready (DTR) and Ready-To-Send (RTS) lines. The modem responds by raising the Data-Set-Ready (DSR) and Clear-To-Send (CTS) lines, and establishing a carrier link to the modem attached to the receiving station. In half-duplex mode the Multiplexer checks the status of the Carrier Detect bit prior to raising RTS. If the CD bit is a 1, the other station on the line has control. Transmission cannot begin until the CD bit is 0 in the modem control register of the transmitting channel.

Once the DTR, RTS, DSR, and CTS lines are high, (and, in half-duplex mode, the CD line is low,) the Multiplexer of the transmitting station issues a data service interrupt to the CPU. When the interrupt is recognized, the Multiplexer sends the address vector for the data service routine to the CPU, and the data is transferred to the Input RAM using Automatic I/O, one byte per interrupt.

The Transmit Ready (TXRDY) line is then tested to determine if the PCI transmitter holding register is ready to accept a character. If it is, the data is loaded into the register for automatic serialization by the PCI onto the line. When the transfer byte count equals zero, the CPU signals the Multiplexer by sending an ECHO signal. The Multiplexer then issues an interrupt to the CPU for an EOB routine. When the interrupt is recognized, the address vector for the EOB routine is sent to the CPU. The EOB is executed and (usually) channel status is read as part of the routine. If no error is indicated in status, the transmit message sequence is complete. If an error occurred, the data must be retransmitted.

The above description of the transmit message sequence applies only if the Inhibit bit is not set (bit 4 in option table byte 0 is not equal to 1). If the bit is set, the microprocessor controlled modem initialization sequence, PCI status and error checking are disabled. Appropriate Write Option Table, Read Modem Register and Write Modem Register commands must be issued to assert DTR, RTS, and test the state of CD (3.5.2 and 3.5.4). Figure 2-8 shows the flowchart for the transmit message sequence.

2.4 AUTO ANSWER

The Receive Buffer and Transmit Buffer commands are normally used to raise the DTR line and initiate the I/O sequence. The DTR line may also be raised in response to a Ring signal. This is done by setting the Auto Answer (ANS) bit in the Modem Control Register. The bit is set by issuing a Write Modem Register command with a code of :1008. When the Auto Answer bit is set, receipt of a Ring signal causes the Multiplexer to generate an EOB interrupt. When the EOB interrupt is recognized and executed, a Write Modem Register command is issued using a code of :1009 to set the DTR bit and keep the ANS bit set. The modem responds with DSR. After an EOB routine is executed, a Receive or Transmit Buffer command can be issued to the channel.

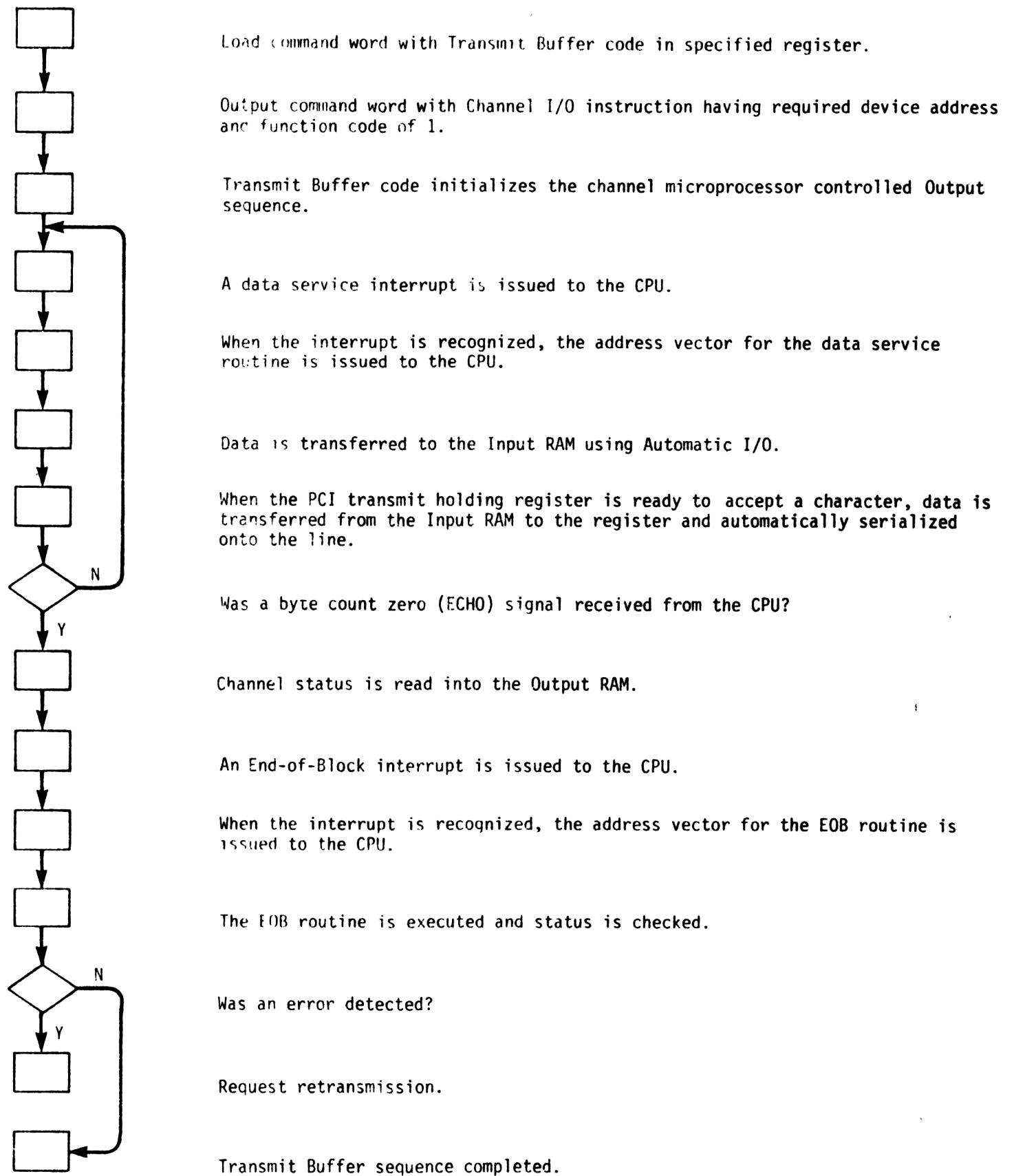


Figure 2-8. Transmit Message Flowchart

SECTION 3

PROGRAMMING INSTRUCTION SET

The Asynchronous Multiplexer conforms to the general I/O programming philosophy presented in the computer reference manuals. The multiplexer accepts the standard I/O instructions from the NAKED MINI 4 computer family.

NOTE

The Asynchronous Multiplexer fully decodes the instruction set detailed in this section. Other instructions, output with the Asynchronous Multiplexer device address, may be partially decoded and the resulting operation is unpredictable.

3.1 DEVICE AND INTERRUPT VECTOR ADDRESSES

The CPU issues instructions to the Asynchronous Multiplexer using 16-bit instruction words. Data transfer is brought about by interrupt vectors generated by the Multiplexer. The instruction words must include the Multiplexer base device address, which is hardware-selected on the Multiplexer board. The base interrupt address, which provides the starting address for the block of data service and End-of-Block routines in system memory, is also hardware-selected on the Multiplexer board. Once the device and interrupt vector addresses are selected, they must be established in software--the base device address in order to issue instructions and commands to the device, and the base interrupt vector address in order for the interrupt routines to be loaded into system memory.

3.1.1 Device Address

All instructions to the Asynchronous Multiplexer must include the base device address. When a specific channel is being addressed, the channel address must be included with the base device address. The base device address is a 4-bit number, established by the setting of a four element switch on the multiplexer board. The channel address is the number (0 through 7) of the desired channel.

Once the base device address is selected, it must be established in software. The EQU directive can be used to establish the base device address and channel number as conveniently used mnemonics. Using ADBAS to represent the base device address, the directive is as follows:

ADBAS

EQU

Device address base determined from
system installation information.

The Asynchronous Multiplexer may have a device address between :1 and :F. Designating the device address as :7, the EQU directive takes the form

ADBAS EQU :7

For Select instructions, which do not require a channel address, the final device address is established as a mnemonic using the EQU directive to combine the base device address with the four bit position shift. Using ADSEL to represent the final Select instruction device address, the directive is as follows:

ADSEL EQU ADBAS%4

When using an assembler that does not recognize the %(shift) sign, the four bit position shift may be achieved using the EQU directive as follows:

ADSEL EQU ADBAS*16

To address a specific channel, a Channel I/O instruction is used. This instruction requires the base device address and the channel address. Using CH as the mnemonic for any channel, and DEVAD as the address of a device attached to one of the channels, the final device address is established using the EQU directive as follows:

DEVAD EQU ADBAS%3 + CH%1

When using an assembler that does not recognize the % sign, the final device address is established using the EQU directive as follows:

DEVAD EQU ADBAS*8 + CH*2

3.1.2 Interrupt Vectoring

The Asynchronous Multiplexer generates interrupts to the CPU. Once an interrupt is recognized, the Multiplexer must then provide a memory address to the computer for use as the next instruction address. This process is called interrupt vectoring.

Transmit and Receive operations both require the issuing of two types of interrupts: data service and End-of-Block. The placement of an interrupt in system memory depends on the channel mode. A channel programmed for full duplex operation locates Transmit interrupts in primary memory locations and Receive interrupts in secondary. A half duplex channel vectors both transmit and receive interrupts to a primary memory area.

The actual address provided by the Asynchronous Multiplexer is controlled by the hardware. The following conventions determine the hardware assignment of the interrupt vector addresses:

The primary interrupt vector addresses are assigned within a block of 64 consecutive memory locations.

The secondary interrupt vector addresses are assigned within the next block of 64 consecutive memory locations.

A full duplex channel is allocated eight consecutive memory addresses in the primary address block and eight consecutive memory addresses in the secondary address block.

A half-channel is allocated eight consecutive memory addresses.

Each vector points to a block of four contiguous words.

An interrupt for data service is vectored to the channel low order memory address.

An interrupt for End-of-Block service is vectored to the channel low order memory address plus four.

The location of a 128 word address block is selectable - in 128 word increments -using an 8-bit switch on the multiplexer board.

Figure 3-1 shows the interrupt vector addresses for the primary 64 address block beginning at the standard interrupt location. This block starts at memory address :0100 and goes through :0140.

Figure 3-1 also shows the format of the eight consecutive memory addresses for channel 0. The format for the remaining channels is the same. If all channels are in full duplex mode, then the block shown shows the location of all transmit (primary) vectors, and would be followed by an identical block of 64 addresses for the receive (secondary) vectors. If the channel is in half duplex mode, the primary vector addresses shown would be used for both transmit and receive. If a different interrupt base address is chosen, the vector addresses would change but the numerical relationship between addresses would remain the same. Since, in practice, some channels may be operating in full-duplex while others are in half-duplex, the distribution of vectors throughout the memory areas may not be uniform. This fact must be remembered when loading the interrupt instructions into memory.

CHANNEL	0	1	2	3	4	5	6	7
Data Service Vector Address	:0100	:0108	:0110	:0118	:0120	:0128	:0130	:0138
End-of-Block Vector Address	:0104	:010C	:0114	:011C	:0124	:012C	:0134	:013C

:0100	Auto I/O Opcode and Device Address
:0101	Byte Count (Negated)
:0102	Memory Buffer Pointer (Buffer-1)
:0103	Unused
:0104	Jump and Store (JST*\$\$+1)
:0105	EOB Subroutine Address
:0106	Unused
:0107	Unused

Figure 3-1. Interrupt Vector Addresses

3.1.3 Interrupt Addresses

Interrupt addresses must be established in software so the required instruction words can be loaded in memory. The base interrupt address can be established using the following mnemonic.

INTBAS	EQU	Interrupt address base determined from system installation information.
--------	-----	--

Assuming the base interrupt address :0180 then:

INTBAS	EQU	:0180
--------	-----	-------

3.2 SELECT AND CHANNEL I/O INSTRUCTIONS

Two types of instructions are sent by the CPU to the Asynchronous Multiplexer: Select and Present instructions and Channel I/O instructions. The first type is used to establish operating conditions on the Multiplexer board. The second type issues a command or data I/O instruction to a specified channel.

3.2.1 Select and Present Instructions (SELP)

Select and Present instructions are used to reset the Multiplexer board and enable or disable interrupts on the board. Four function codes are used with Select instructions:

- Function code 4 resets the Multiplexer board and enables interrupts. The Multiplexer interrupt enable (MIE) flip flop, the data group enable (DGIE) flip flop, and the End-of-Block group enable (EGIE) flip flop are set; the interrupt request (IR) flip flop and the interrupt pending (IP) flip flop are cleared. The Busy and ECHO flip flops, modem control lines and the microprocessor are cleared, and the option table bytes assume default values. Use function code 4 in the form

SELP	R,ADSEL+4	(R = A, X, Q, or Y Registers)
------	-----------	-------------------------------

One of the registers, A, Q, X, or Y, must be specified.

- Function code 5 sets the MIE flip flop. Use function code 5 in the form

SELP	R,ADSEL+5
------	-----------

- Function code 6 resets the MIE, disabling all Multiplexer board interrupts. Use function code 6 in the form

SELP	R,ADSEL+6
------	-----------

- Function code 7 resets the IR flip flop. Use function code 7 in the form

SELP	R,ADSEL+7
------	-----------

3.2.2 Channel I/O Instructions

Channel I/O instructions address specific channels to execute control and data transfer functions. The I/O instructions descriptions are presented in assembly format for easy use. The label (if used), mnemonic, and operand with an appropriate function code appears for each instruction. The instructions used to communicate with the Asynchronous Multiplexer and the addressed channel are:

IN INPUT TO A SPECIFIED REGISTER

OUT OUTPUT FROM A SPECIFIED REGISTER

AIB AUTOMATIC INPUT BYTE

AOB AUTOMATIC OUTPUT BYTE

3.2.2.1 Control Usage

Channel I/O instructions with a function code of 1 initiate Asynchronous Multiplexer operations or request the status word from the addressed channel. The CPU initiates an Asynchronous Multiplexer operation by using the OUT instruction to output a command word from a specified register to the Multiplexer. The required command word must be loaded into the specified register prior to the execution of the OUT instruction.

To output the contents of the specified register as control information, use function code 1 in the form

OUT 4Reg,DEVAD+1

where 4Reg is a standard NAKED MINI 4 assembler documentation convention meaning that one of the A, Q, X, or Y registers must be specified.

The CPU obtains the channel status word by issuing an IN instruction. To store the status word in a specified register, use a function code of 1 in the form

IN DEVAD+1,4Reg

3.2.2.2 Data Transfer Usage

Channel I/O instructions with a function code of 0 transfer data between the computer and the addressed channel via the Multiplexer. Data transfer via the Asynchronous Multiplexer is accomplished by Automatic I/O. Automatic I/O instructions are always initiated by an interrupt. The Automatic I/O uses the AOB and AIB instructions to transfer data between system memory and a peripheral device without altering the CPU registers. The programming for the AOB and AIB instructions is the same except for the instruction mnemonic. The AOB instruction is shown in the following example.

To transfer data using the Automatic I/O, the Automatic I/O instruction words must be loaded at the interrupt address using function code 0. In addition, it must be specified whether the byte address is in the upper or lower 64K bytes. These operations are accomplished as follows:

- *Enter the word address in the X register.
- *Enter the byte count in the Q register.
- *Jump to the subroutine by entering:

	JST	SETUP	
	.		
	.		
SETUP	ENT		
	COPY	INST,A	
	COPY	A,INTBAS	SETUP AUTOMATIC I/O INSTRUCTION
	NEG	Q,Q	Q=NEGATIVE BYTE COUNT
	SUB	=1,X	
	SHIFT	X,LO,1	CONVERT WORD ADDRESS TO BYTE ADDRESS
	JF	OV,\$+2	TEST FOR OVERFLOW. JUMP IF ABSENT
	RBIT	15,Q	RESET BIT 15 OF THE BYTE COUNT
	ADD	=1,X	CONVERT TO BYTE ADDRESS MINUS 1
	COPY	X,INTBAS+2	STUFF IN AUTOMATIC BYTE BLOCK
	COPY	Q,INTBAS+1	SET NEGATIVE BYTE COUNT IN AUTOMATIC
			I/O BLOCK
	JMP	*SETUP	
	.		
	.		
INST	AOB	DEVAD+0	

When using a stack, the following changes are made in the program:

JST	SETUP is replaced by JSK	SETUP	
SETUP	ENT is replaced by SETUP	EQU	\$
JMP	*SETUP is replaced by RSK		

A NOP instruction must never be loaded in an interrupt vector address. The Asynchronous Multiplexer does not cancel the interrupt when a NOP instruction is executed. This results in a constant interrupt to the CPU, preventing progress of the system program.

3.3 END-OF-BLOCK INTERRUPT

The Asynchronous Multiplexer provides an End-of-Block interrupt at the completion of a data transfer sequence. The End-of-Block interrupts are vectored to the channel address plus four. An appropriate End-of-Block routine must be provided at a convenient location in memory. The transfer of program control to the End-of-Block routine is set up in the form

ABS	DEVAD+4	Channel End-of-Block Interrupt Vector Address
JSK	*\$+1	
DATA	EOB	Address of End-of-Block Routine

The programmer must establish a stack for use with this interrupt before using the JSK instruction. If no stack is available JST should be substituted for JSK.

3.4 UPDATE INTERRUPT STATUS (UIS) INSTRUCTION

The Update Interrupt Status Instruction outputs the CPU processor status word to all I/O devices in the system. The instruction is used to selectively enable or disable data service and EOB interrupts. Data service and EOB interrupts are disabled by setting bits 13 and 12 respectively, to 1. The processor status word is output by UIS in the assembler instruction command field.

NOTE

Care must be taken in using these inhibits. Since there is only one set of interrupt hardware for both data service and EOB, it is possible for an EOB inhibit to lock out Data Service Interrupts, and vice versa.

3.5 COMMAND WORD

Command words are used to issue control codes to the Multiplexer. The control codes initiate the microprocessor operations that control the Multiplexer. Command words are not part of the computer instruction set but must be loaded into a register specified by the programmer and output using Channel I/O instructions with a function code of 1. Command words have the following hexadecimal skeletons, n representing a variable field:

Reset Receive Half-Channel	:8000
Load Receive Command Field	:4n00
Reset Transmit Half-Channel	:2000
Load Transmit Command Field	:10nn

3.5.1 Reset Receive Half-Channel (:8000)

This command sets the Receive Select (RXSEL) bit for the addressed channel in the Select register. Setting the RXSEL bit provides a command which partially initializes the receive half-channel it is directed toward. All message states are cleared, the receiver is disabled, and the half-channel is immediately ready for a new command.

3.5.2 Load Receive Command Field (:4n00)

This command loads the receive command field into the Multiplexer Receive Command buffer. The variable field in the command word may be configured in eight ways, to give the following eight commands:

Read Modem Register	:4000	Receive Message CM,RM	:4B00
Read Option Table	:4100	Receive Message AE	:4C00
Receive Message	:4800	Receive Message AE, CM	:4D00
Receive Message RM	:4A00	Receive Message AE, CM, RM	:4F00

where AE, CM, and RM mean Auto Answer, Special Character Match Enable, and Range Match Enable, respectively. These terms are explained in 3.5.2.3.

3.5.2.1 Read Modem Register

Read Modem Register causes the 4-bit modem receive register to be read into the receiver status register of the addressed channel. The meaning of the bits is explained below. The Format of the receiver status register is shown in Figure 3-2.

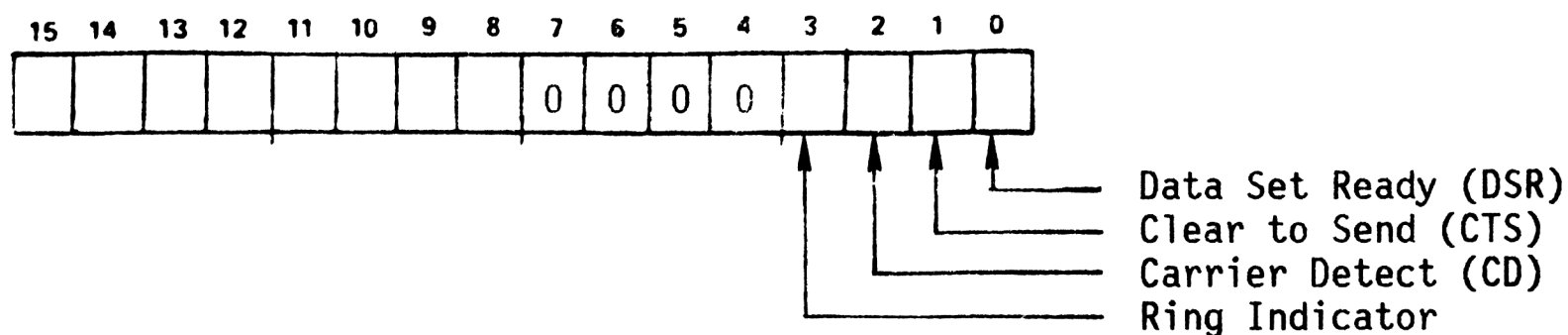


Figure 3-2. Receiver Status Register

Bit 0, the Data Set Ready bit, indicates the status of the data set (modem). Bit 0=1 when the modem is on line in a Ready condition. The bit indicates the state of the modem only, and not that of an attached peripheral.

Bit 1, the Clear-to-Send bit, indicates the response of the modem to a Request-to-Send signal sent by the CPU. When bit 1=1, the modem is ready to transmit data from the computer.

Bit 2, the Carrier Detect bit, indicates that the modem on the opposite side of the line has a message to communicate to the modem attached to the Multiplexer. When bit 2 = 1, a carrier has been established by the modem on the transmitting side to the modem attached to the Multiplexer.

Bit 3, the Ring Indicator bit, is used in Auto Answer mode to bring about a software controlled rise of the DTR Line (2.4 in Functional Description). In this mode the Ring Indicator bit is a 1 in response to a Ring signal. In other than Auto Answer mode the Ring Indicator bit is not a part of the modem initialization sequence and has no functional significance.

3.5.2.2 Read Option Table

Read Option Table causes an Automatic I/O read of the option table associated with the addressed channel. The five bytes are read beginning with byte 0, followed by a read of the two special characters. To carry out this operation an Automatic I/O instruction and buffer in system RAM must be provided. The read will terminate only when ECHO is returned from the CPU indicating a full buffer. All information read beyond the two special characters is however, irrelevant. Characters are spaced by an internal 10 ms timer to avoid overloading the multiplexer. The timer can be bypassed by setting bit 6 to a 1 in option table byte 0.

3.5.2.3 Receive Message

Receive Message is used to activate the receive half of the channel and allow the multiplexer to pass characters from the channel to the system RAM via Automatic I/O. Three options are enabled by bits in the command. Depending on how many bits are set, the half channel may operate in any of six Receive Message configurations. The three options are:

- AE - Auto Echo. This option is used only when the modem is operating in full duplex. If there are no errors, the incoming data is sent back automatically through the transmit half of the channel. The auto echo indicates that the data was received by the multiplexer - but not necessarily by the CPU. Matched characters that cause EOB are not echoed.
- RM - Range Match Enable. This allows the multiplexer to terminate (with an End-of-Block interrupt) if a received character - without parity bit - falls in the range from zero to :1F. The character received is recorded in status.
- CM - Special Character Match Enable. This allows the multiplexer to terminate (with an EOB interrupt) if a received character matches one of the two stored special characters for that channel. The characters are individually programmable. When Range Match Enable and/or Special Character Match Enable are used the channel status register will indicate the reason for the interrupt (3.6.1).

3.5.3 Reset Transmit Half-Channel (:2000)

This command sets the Transmit Select (TXSEL) bit for the addressed channel in the Select register. This provides a command which partially initializes the transmit half channel it is directed toward. All message states are cleared, the transmitter is disabled, and the half-channel is immediately ready for a new command.

3.5.4 Load Transmit Command Field (:10nn)

This command loads the transmit command field into the Multiplexer Transmit Command buffer. The two variable fields in the command word can be configured to give seven basic options. In practice, the Transmit Command Field is divided into two sub-fields; a five bit data field and a three bit opcode. The format is shown in Figure 3-3.

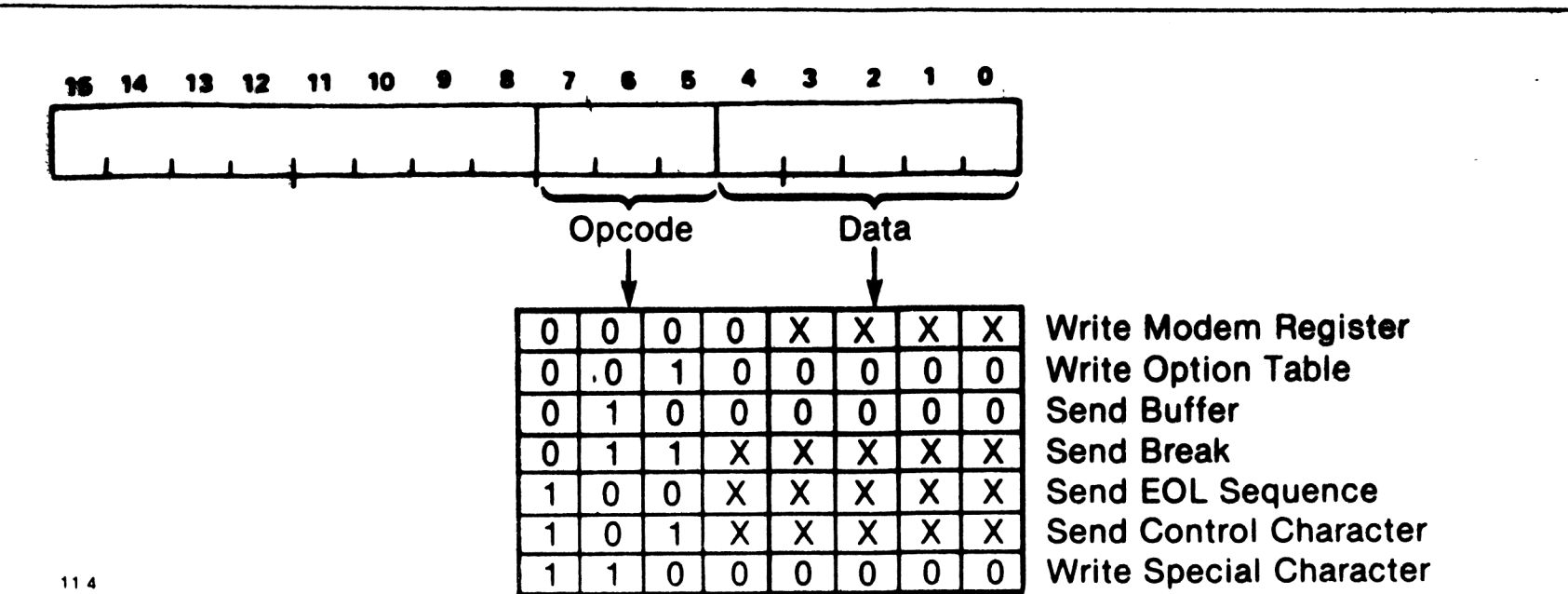


Figure 3-3. Load Transmit Command Field Command Format

3.5.4.1 Write Modem Register

Write Modem Register is used to configure the four signal bits in the command file section of the modem control register (2.1.6 in Functional Description). The format of the data field for this command is shown in Figure 3-4.

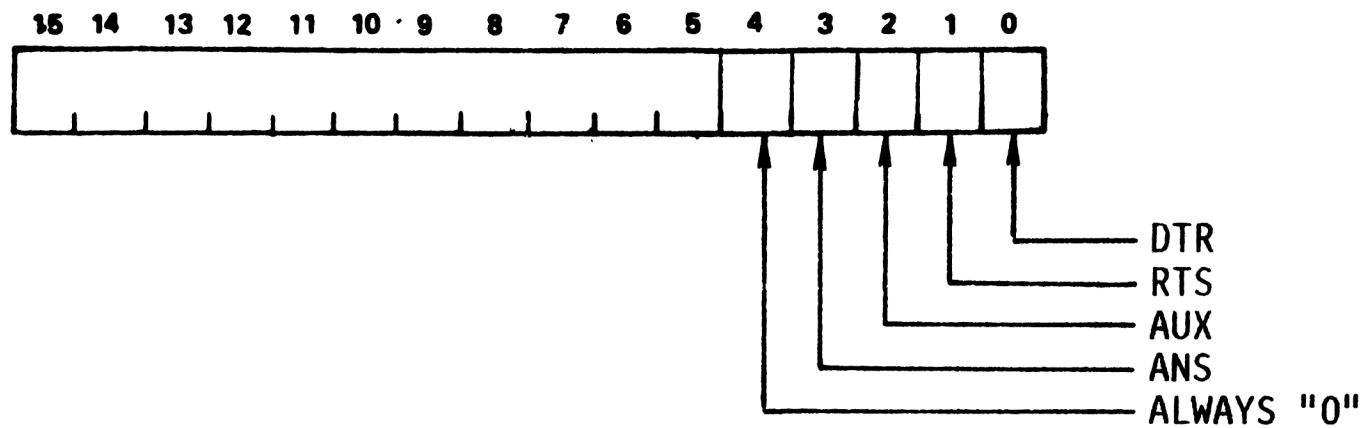


Figure 3-4. Data Field Format for Write Modem Control Register Command

Bit 0, is the Data Terminal Ready bit. When bit 0=1, the Multiplexer is ready to receive a message from the modem.

Bit 1 is the Request to Send bit. When bit 1=1, the Multiplexer is ready to transmit a message to the modem.

Bit 2 is the Auxiliary (AUX) bit. When bit 2=1, AUX is asserted. The bit is used for whatever purpose the user specifies.

Bit 3 is the Auto Answer (ANS) bit. When bit 3=1, the End-of-Block interrupt is held off until one of the bits in the modem receiver register (Clear to Send, Data Set Ready, Carrier Detect, or Ring Indicator) changes state.

3.5.4.2 Write Option Table

Write Option Table writes up to four bytes of the option table starting with byte 0. The data is transferred via Automatic I/O. If less than four bytes are provided, the remaining bytes remain unaffected.

3.5.4.3 Send Buffer

Send Buffer is the main operational transmit command used to send a message onto the communication channel. The Send Buffer operation is terminated when an ECHO signal (byte count zero) is received from the CPU, indicating that the system buffer is empty or at an end.

3.5.4.4 Send Break

Send Break transmits a break, the duration of which is 10 msec times the value set in the data field. A default value (zero in the data field) transmits a 220 ms break. A break is a continuous space on the communication line used to interrupt (or get attention for) the transmitting terminal. No buffer is required.

3.5.4.5 Send EOL

Send EOL Sequence sends a carriage return, followed by as many nulls as the value set in the data field, followed by a line feed. No Buffer is required.

3.5.4.6 Send Control Character

Send Control Character transmits a character comprised of the 5 bits in the data field padded with zeros to the character length. No buffer is required.

3.5.4.7 Write Special Character

Write Special Character writes the special control characters of the addressed channel from a buffer in system RAM into the multiplexer special character register.

3.6 STATUS

Each channel has a status register which can be read by the CPU at any time but whose contents are valid only under certain conditions. The status register format is shown in Figure 3-5.

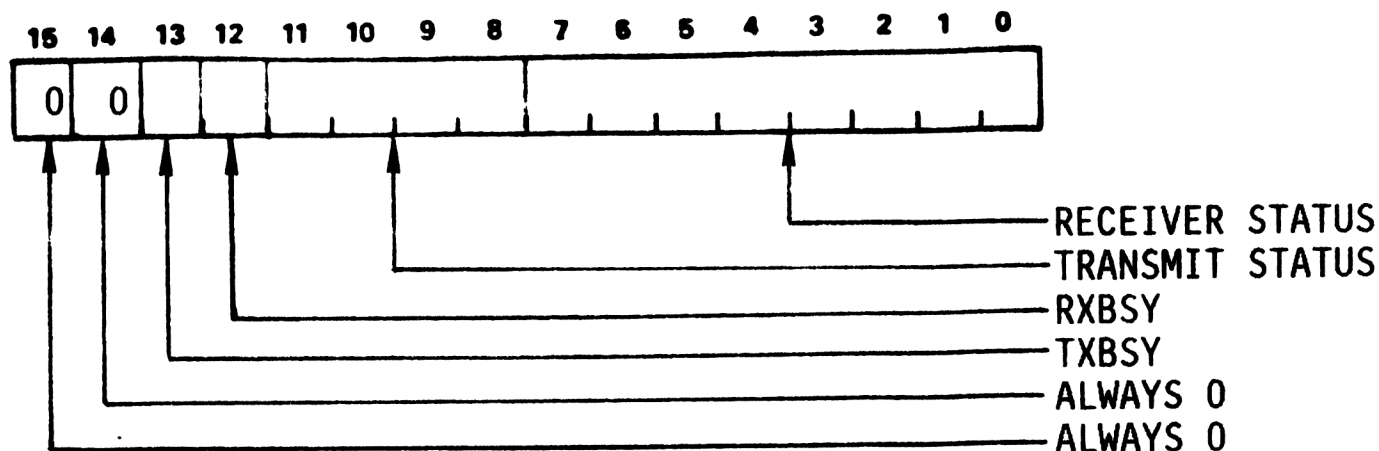
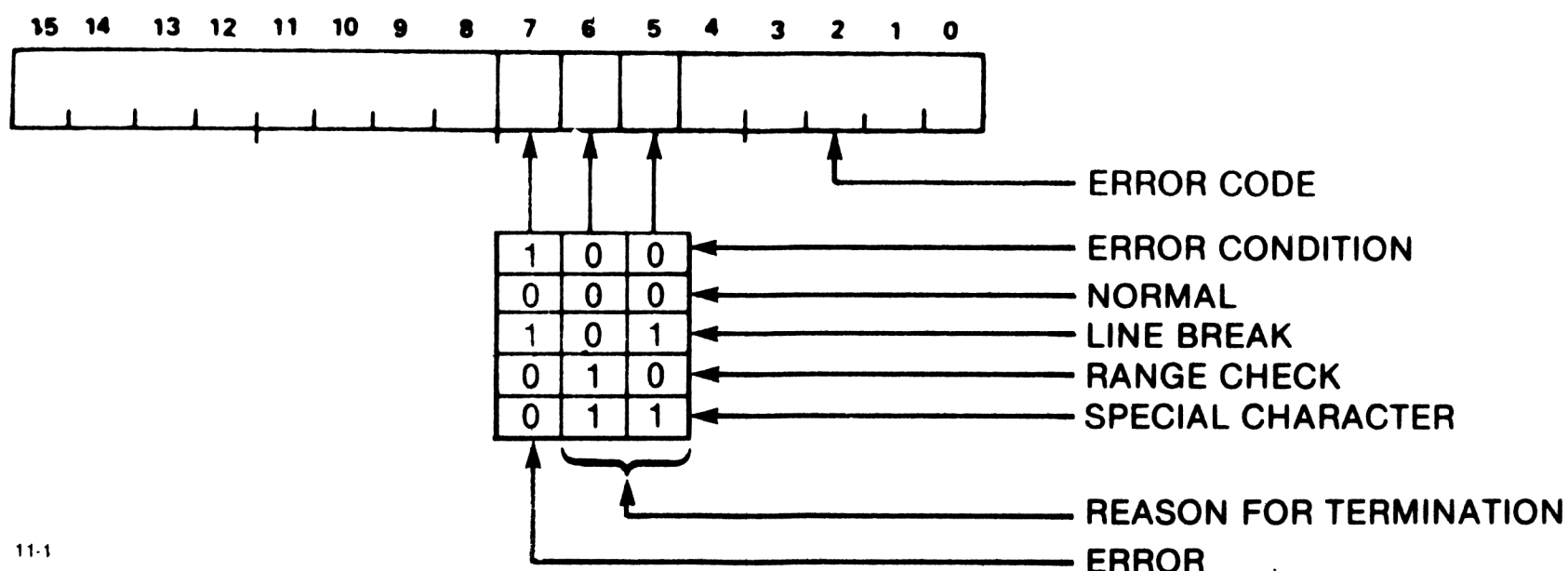


Figure 3-5. Status Register Format

3.6.1 Receiver Status Field

The status register receiver status field is valid only when an EOB has occurred on completion of a receive command, the RXBSY bit is not set, and a partial reset has not taken place. The Receiver Status field format is shown in Figure 3-5.



11-1

Figure 3-6. Receiver Status Field Format

Bit 7 is the Error bit. When bit 7=1, an error occurred during the receive operation. In an error condition the rest of the bits are defined as follows:

- bits 5 and 6 are 0, except if a Line Break has been detected. In a Line Break condition bit 5=1 and bit 6=0.
- bits 0 through 4 are the Error Code bits, and may assume a hexadecimal value of 0 to :1F. The value indicates the cause of the error. The error code is shown in Table 3-1.

Table 3-1. Receive Error Code

Error Code Value	Explanation
0	Lost DSR
1	Lost CD
2	Framing Error
3	Priority Error
4	Overrun Error
5 - :1F	Not Used

When bit 7=0, no error has occurred and bits 5 and 6 indicate the reason for termination of the receive operation, as follows:

- If bits 5 and 6 are both 0, termination is normal.
- If bits 5 and 6 are 0 and 1, respectively, termination is due to range check.
- If bits 5 and 6 are both 1, termination is due to the receipt of one of the special characters. The number of the character received is indicated in bit 0 (see below).

When bit 7 = 0, the meaning of bits 0 through 4 depends on the reason for termination and the command executed, as follows:

- If termination is normal, bits 0 through 4 are 0, except if a Read Modem Control Register command has been executed. In this case bits 0 through 4 contain the contents of the Modem Control Register.
- If termination is due to range check, bits 0-4 record the range check character.
- If termination is due to the receipt of one of the special characters, bit 0 indicates which special character was received. Bit 0 = 1 to indicate special character one; bit 0 = 1 to indicate special character two. Bits 1 through 4 are 0.

3.6.2 Transmit Status Field

This field is valid only when an EOB has occurred on completion of a transmit command and partial reset has not taken place (the TXBSY bit is not set). The Transmit Status field format is shown in Figure 3-7.

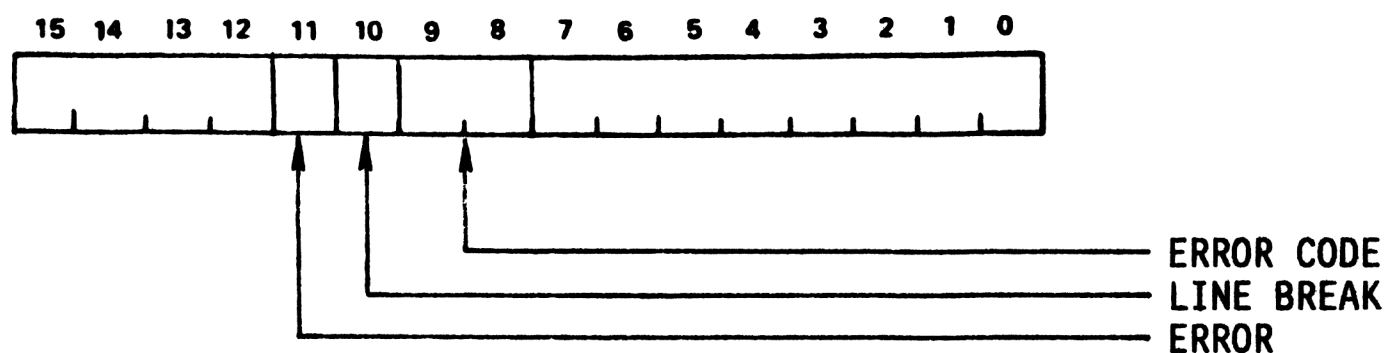


Figure 3-7. Transmit Status Field Format

Bit 11 is the Error bit. If bit 11-1 bits 8 and 9 indicate the error code, which may be from 0 to 3. The Transmit Error Code is shown in Table 3-2.

Table 3-2. Transmit Error Code

Error Code Value	Explanation
0	Lost DSR
1	Not Assigned
2	Not Assigned
3	Illegal Command

Bit 10 is the Line Break bit. Bit 10=1 when a Line Break is detected during a transmit operation.

SECTION 4

INSTALLATION

This section describes the installation considerations unique to the Asynchronous Multiplexer. For a description of the standard installation procedures, refer to the installation section of the appropriate computer handbook. The Asynchronous Multiplexer board is shown in Figure 4-1.

4.1 SYSTEM CONFIGURATION

The Asynchronous Multiplexer is a half-card device which may be inserted, according to interrupt priority, in any slot in the motherboard. A sixteen pin socket is provided to accommodate a half-card processor on the 100 side of the same full size slot.

If the multiplexer board is not jumpered directly to the processor, care must be taken to ensure that the priority chains between the processor and the multiplexer are not broken. All cards located between the processor and the Asynchronous Multiplexer must provide propagation of the priority chains, and in turn, the multiplexer propagates all MAXI-BUS interface priority chains. No open slots can exist between the multiplexer and the processor or the chain is broken.

4.2 ADDRESS ASSIGNMENTS

The Asynchronous Multiplexer must be assigned a base device address and a base interrupt address before it can be operated in the computer system.

4.2.1 Device Address Assignments

Base device address assignment is made using a 4-element switch on the Multiplexer board. The Asynchronous Multiplexer may be assigned any base device address from :1 through :F. When assigning the base device address, care must be taken to avoid an address already in use or dedicated to a specific use. Refer to the applicable computer reference manual for a listing of standard device address assignments and restrictions. To prevent a conflict base device addresses must be compared according to absolute binary (not hexadecimal) values. This is because bits 3 through 7 constitute the device address in a Select instruction, while bit 3 is the high level address bit in a Channel I/O instruction. If bits 3 through 7 assume identical values in an instruction word, devices attached to the address bus cannot distinguish between a Select instruction to one device and a Channel I/O instruction to another.

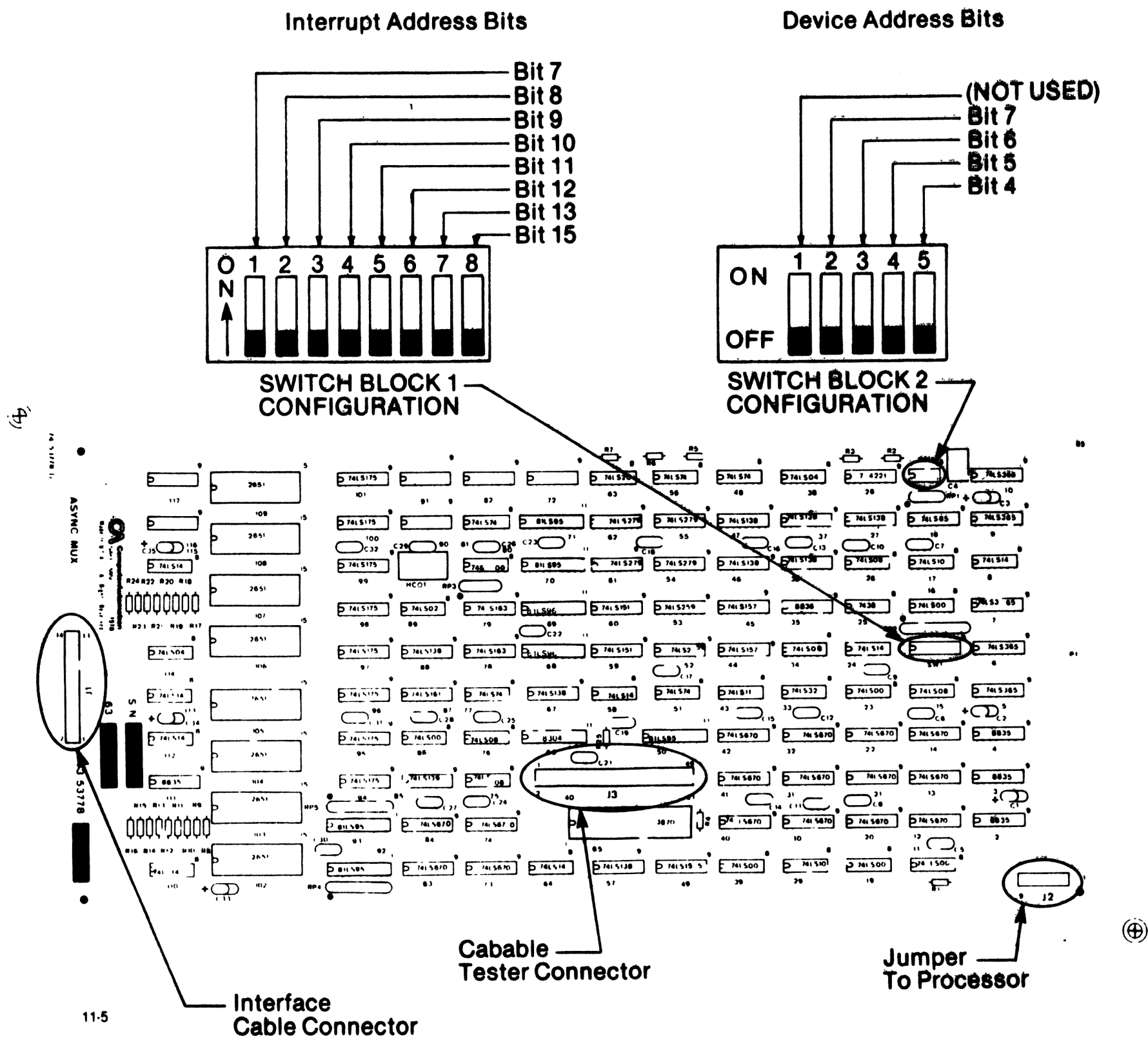


Figure 4-1. Asynchronous Multiplexer Board

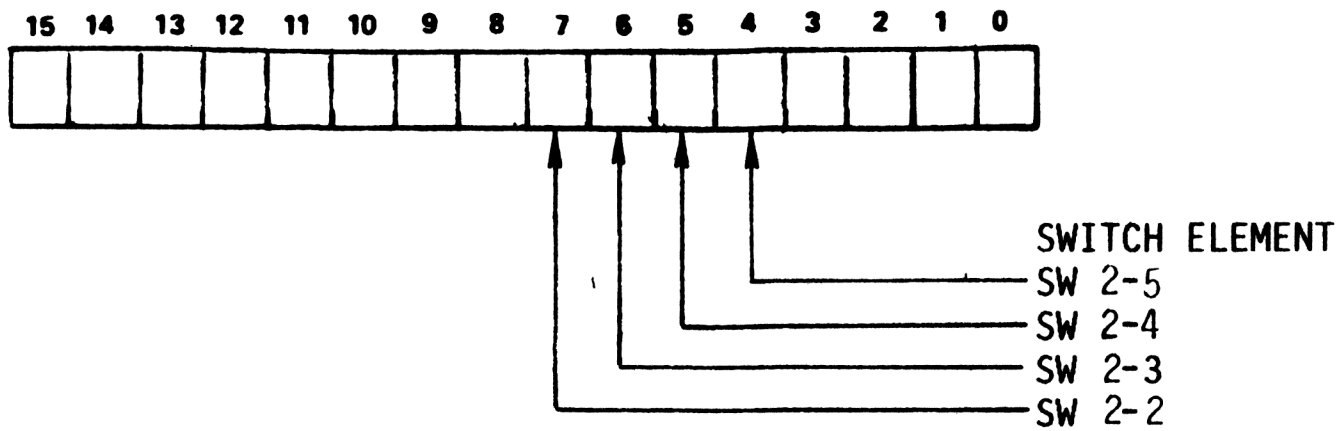


Figure 4-2. Device Address Bit Assignments

The correspondence between switch element settings and device address bit settings is shown in Figure 4-2. Bit positions apply to both Select instructions and Channel I/O instructions. The switch element settings for all 15 possible device addresses are listed in Table 4-1. The general device address zero is specifically dedicated and must not be used as the base address for the Asynchronous Multiplexer.

Table 4-1. Device Address Switch Settings

DEVICE ADDRESS	SWITCH ELEMENT NUMBER			
	2	3	4	5
:1	0	0	0	1
:2	0	0	1	0
:3	0	0	1	1
:4	0	1	0	0
:5	0	1	0	1
:6	0	1	1	0
:7	0	1	1	1
:8	1	0	0	0
:9	1	0	0	1
:A	1	0	1	0
:B	1	0	1	1
:C	1	1	0	0
:D	1	1	0	1
:E	1	1	1	0
:F	1	1	1	1

A "1" indicates the switch "on" position.

4.2.2 Interrupt Vector Address

The 128 word interrupt vector base address may be from zero to :3F80 or :8000 to :BF80. Care must be taken when assigning a base address that the address selected does not conflict with one previously assigned and that memory is large enough to include all vector addresses.

The base address is selected in 128 word increments-using an 8 bit switch on the multiplexer board. The 64 word secondary vector offset for use in full-duplex mode is chosen by setting bit 1 to a 1 in byte 0 of the channel option table. The format of an interrupt vector as sent to the MAXI-BUS interface by the multiplexer is shown in Figure 4-3.

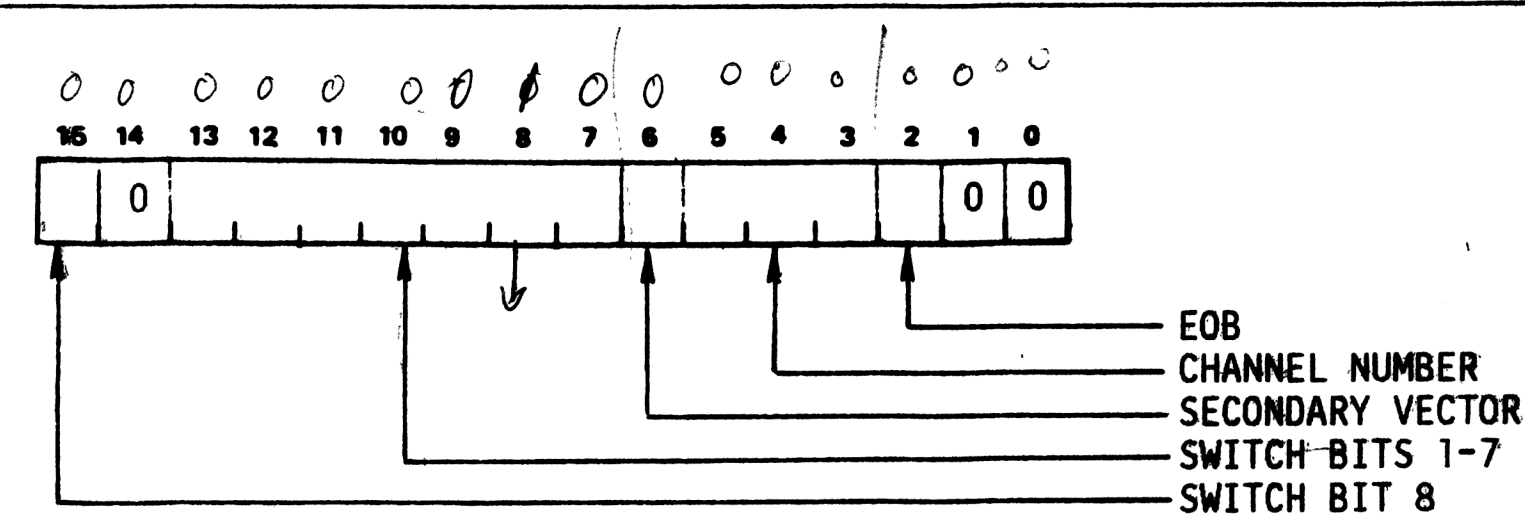


Figure 4-3. Interrupt Vector Format

APPENDIX A

RS-232-C INTERFACE PANEL

A.1 INTRODUCTION

The RS-232-C Interface Panel is a communications module designed for use with Computer Automation Inc. communications controllers. The panel provides drivers and receivers to make the controllers compatible with RS-232-C (CCITT V.24) modems and peripheral devices.

Panel options include:

73-53768-04, which allows for the connections of a maximum of four peripheral devices to one controller or, 73-53768-08, which allows for the connection of a maximum of eight peripheral devices to one controller.

The RS-232-C Interface Panel is connected to terminals using cable 45080-25, and to modems using cable 45081-15. Cable 72-53768-00 is used for test purposes.

A.2 SPECIFICATIONS

Power Requirements

All power is derived from the MAXI-BUS via the communications controller.

	Maximum
4-channel (45060-04)	
+5 Vdc	0.3A
+12 Vdc	0.1A
+12 Vdc	0.1A
8-channel (45060-08)	
+5 Vdc	0.5A
+12 Vdc	0.2A
-12 Vdc	0.2A

Environmental Requirements

Operating Temperature Range (Degrees Celsius)	0 to +50
Non-operating Temperature Range (Degrees Celsius)	-20 to +100
Relative Humidity (Percent, Non-condensing)	5 to 90

The RS-232-C Interface Panel uses 25-pin, D-type connectors.
The pin-list for the D-type connectors connecting to peripheral devices is shown in Table A-1.

Table A-1. RS-232-C Panel Interface Pin-List

PIN NUMBER	SIGNAL NAME
1	Protective Earth Ground
2	Transmit Data
3	Receive Data
4	Request to Send
5	Clear to Send
6	Data Set Ready
7	Signal Ground
8	Carrier Detect
14	Auxiliary Signal
20	Data Terminal Ready
22	Ring Indicator

A.3 INSTALLATION

The Interface Panel is to be installed in the 19 in. equipment rack using Mounting Bracket Kit No. 95-21207-00.

To install the panel, proceed as follows:

1. Remove hardware from package and check contents against Hardware List, Table A-1.
2. Position brackets on the standard 19 in. mounting cabinet rails. Brackets should be spaced 8.25 in. apart. (8.25 in.=15 hole spaces inclusive) See Figure A-1.
3. Attach brackets loosely, using No. 10 (A) screws and washers supplied in the mounting kit.
4. Position brackets on Interface Panel as shown in Figure A-1. Make sure the panel is positioned so the peripheral connector plugs are aligned vertically on the right side. Check the back of the panel to assure it is protected from contact with the mounting brackets by insulated tape.
5. Assemble and insert No. 6 (B) screws and hardware as shown in Figure A-2.
6. Tighten B screws and hardware as shown in Figure A-2.
7. Connect ribbon cable No. 72-21156-06 between connector J9 on Interface Panel and connector J1 on Multiplexer board.

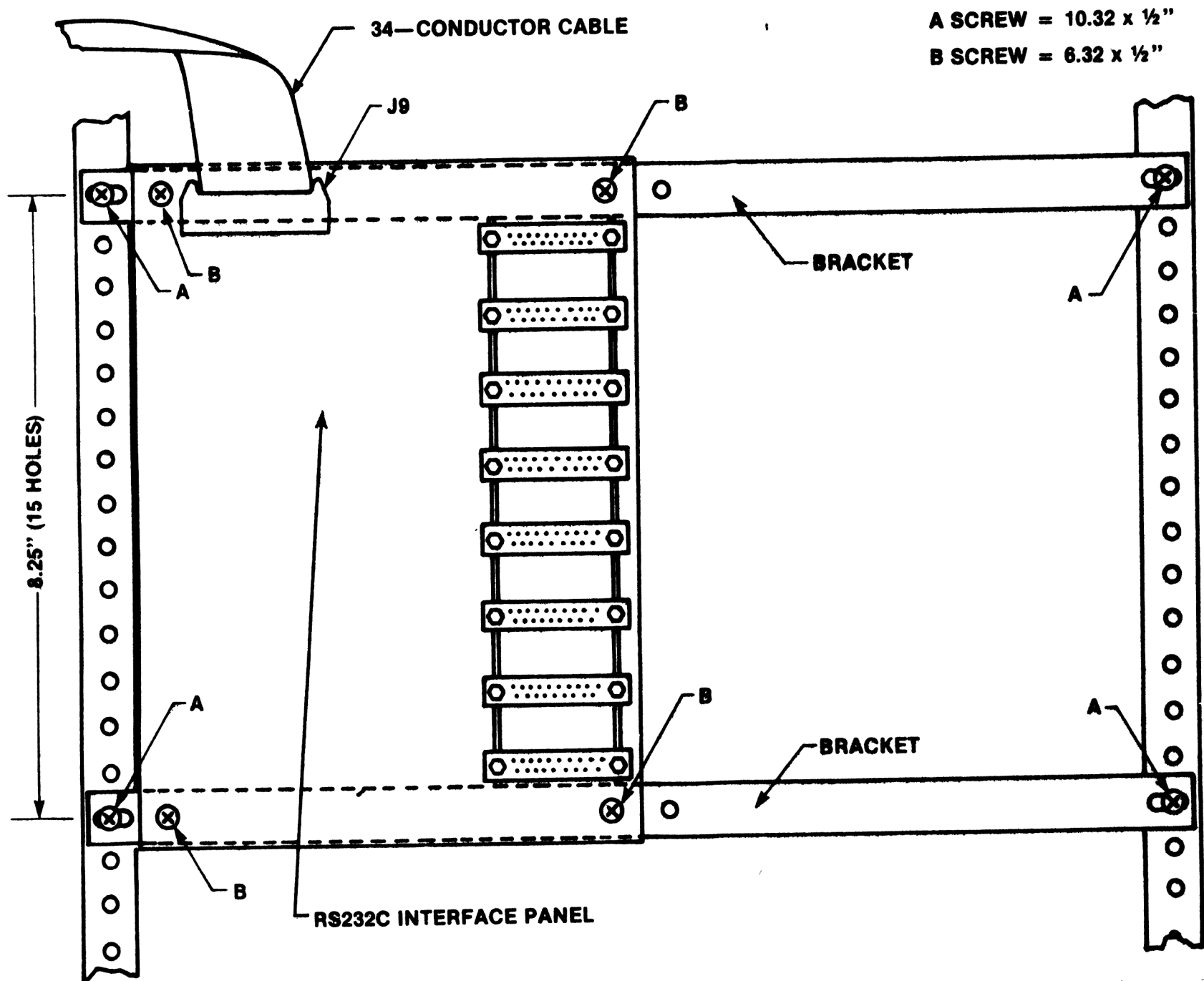


Figure A-1. RS-232-C Interface Panel Mounting on Equipment Rack

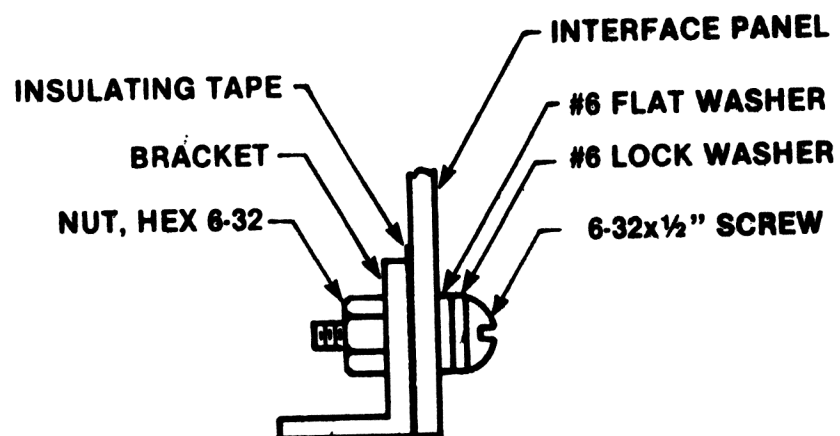


Figure A-2. Side View B Type Screw Configuration

Table A-1. Shipping Kit Hardware List

CAI PART NUMBER	QTY	DESCRIPTION
72-20567-00	2	BRACKET, ANGLE
41-49168-67	4	SCREW, PAN HD- PHILLIPS, 10-32 x 1/2 in.
41-49168-33	4	SCREW, PAN HD- PHILLIPS, 6-32 x 1/2 in.
42-49246-06	4	WASHER, FLAT, NO. 6
42-49248-06	4	WASHER, LOCK, NO. 6 SPLIT
42-49248-10	4	WASHER, LOCK, NO. 10 SPLIT
42-49328-06	4	NUT, HEX 6-32

CUSTOMER INFORMATION BULLETIN

CIB No. 1234 - 53778-XX Asynchronous Multiplexer

The maximum speed of transmission over a single line is listed as 19.2K Baud. What is unclear, is the throughput achievable over several lines running concurrently at different Baud-rates.

The figure of 19.2K Baud should be taken as a maximum throughput across all lines. For example, two lines could run at 9.6K Baud, four at 4.8K Baud or one at 9.6K Baud with two others at 4.8K Baud and so on.

If lines can be guaranteed NOT to run concurrently, then each can be configured for 19.2K Baud.

Under certain programming conditions, the Multiplexer could lose characters. This is due to the way the interrupt logic has been implemented, with only one set of logic to handle End-of-Block and Data interrupts.

If a UIS instruction is sent to inhibit the EOB interrupt, then Data interrupts as well will be inhibited. The Multiplexer is capable of holding one character per line, therefore if interrupts are blocked for longer than the time necessary to transmit a second character, that character may be lost or the attached device may hang-up.

In any real-time environment, blocking interrupts is never recommended. At the maximum throughput of 19.2K Baud, this represents a time period of 521 μ s between characters. Interrupts should never be blocked for greater than this period of time and it is highly desirable to limit blocking to the minimum period of time where such an event could disrupt the operation of the program.

CAI Limited
Technical Support Group
Naked Mini Division

July, 1979

