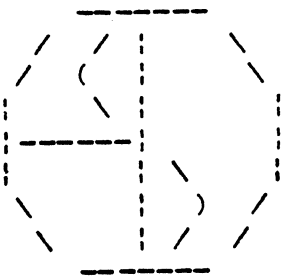
# MIGHTY MUX

## DMA MULTIPLEXER
## USER MANUAL

E D U C A T I O N A L  D A T A  S Y S T E M S

1682 Langley Avenue, Irvine, California 92714

(714) 556-4242



# EDSI MIGHTY-MUX

## PROGRAMMABLE DMA MULTIPLEXER

## USER'S MANUAL



EDS-1033-1
21 SEPT 77

Table of Contents

## Appendices

# Figures

# 0. Introduction

The EDSI MIGHTY-MUX is a complete I/O Communications System for any Nova type computer. It permits up to 128 devices to communicate concurrently with the mini-computer. These may be any combination of teletypes, typewriters, CRT terminals, printers, card readers, modems, and other RS-232 or current-loop devices. They may operate in half or full duplex, with or without automatic echo.

The MIGHTY-MUX is more than a conventional multiplexer in two fundamental respects. First, it provides program control on a port by port basis over Baud rate and most other port parameters. Second, it permits each port to operate in either character-at-a-time or automatic buffer mode. In automatic buffer mode, each port is assigned a buffer in the core memory of the computer. This may be of any size and have any location in core. The MIGHTY-MUX automatically services these buffers, inputting or outputting strings of characters without any interruption of the computer program until the entire string is completed. To accomplish this, it operates through the DMA channel of the computer.

The program gives the MIGHTY-MUX its I/O processing commands by storing appropriate control words in certain dedicated core locations. These I/O commands define, for each port:

- Length and location of the string buffer in core from/into which character strings are to be transferred;
- Baud rate for transmission and reception, from among 8 standard rates up to 9600 Baud;
- Character length - 5, 6, 7, or 8 bits;
- Parity mode - even, odd, or none;
- Number of Stop Bits - one or two;
- When interrupts are desired - for each character, for certain defined control characters, or only when the end of the specified buffer is reached;
- Whether or not incoming characters are to be automatically echoed.

The MIGHTY-MUX can also be used for synchronous communication, by means of the EDS-302 Synchronous Expander Board (see Appendix A).

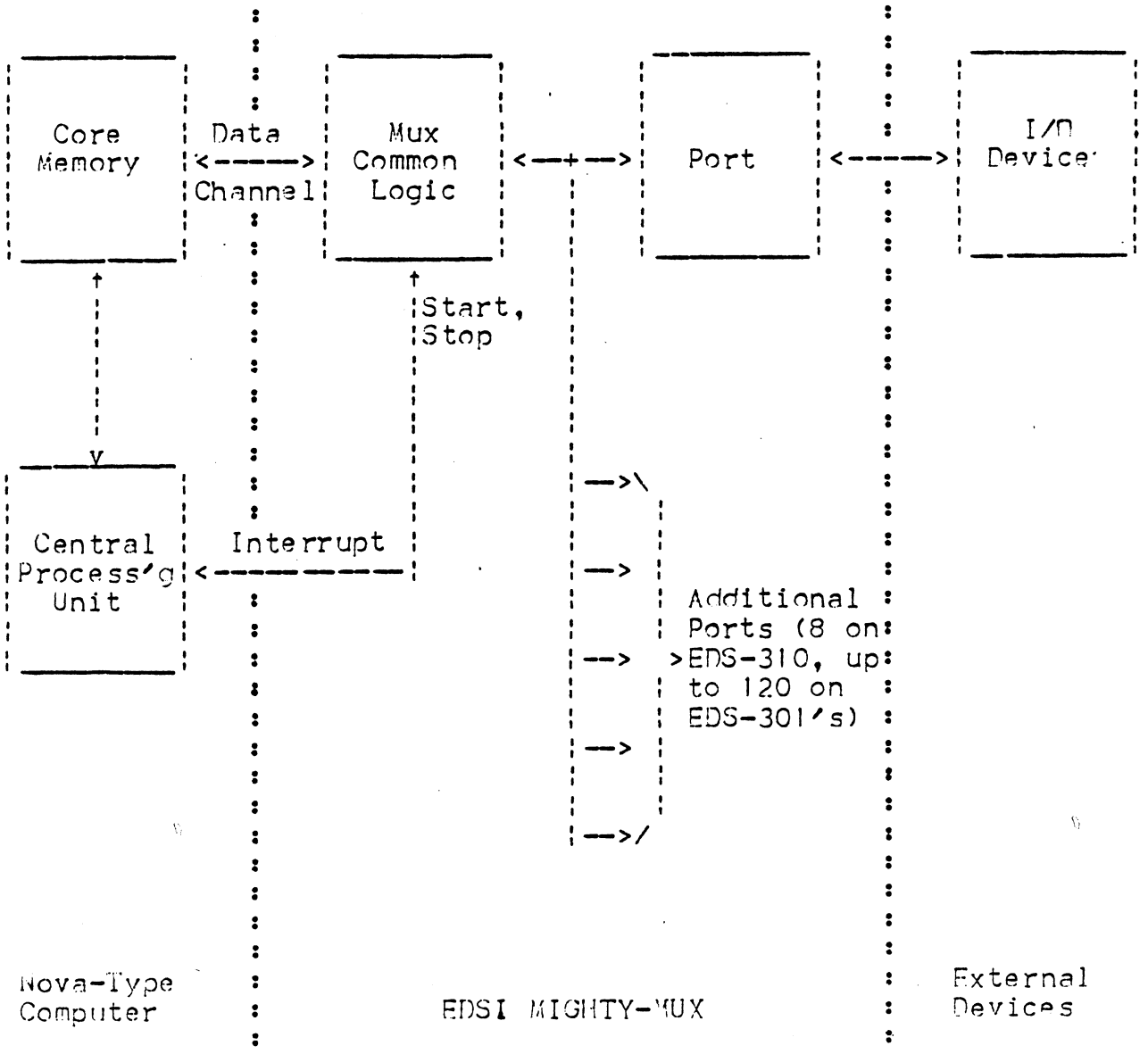



Figure 1. Block Diagram Showing EDSI MIGHTY-MUX Interfacing External Devices to Nova-Type Computer

Another very useful feature of the MIGHTY-MUX is that it may be used as the master terminal interface board, because when the multiplexing function is not turned on the MIGHTY-MUX responds to standard Device Code 10/11 type I/O software commands. Thus use of the MIGHTY-MUX obviates the need for a master I/O board, saving not only its cost but also a slot in the computer.

In addition to these broad programming capabilities, the MIGHTY-MUX has exceptional electrical interfacing capabilities. Each port has 4 signal lines: 2 data lines (incoming and outgoing), and 2 auxiliary lines - the outgoing Device Control line and the incoming Device Status line. These auxiliary lines may be programmed for any desired purpose. For example, for a full-duplex modem such as a Bell System 103 Type Data Set, Device Control may be used as the Data Terminal Ready line (to enable automatic hang-up), and Device Status may be the Received Line Signal Detector (permitting automatic log-off). For half-duplex control, Device Control would be the Request To Send, and Device Status, Clear To Send. For certain line printers, Device Control may be used as the Print Line command, and/or Device Status may indicate the printer's Ready/Busy status.

All four of these lines operate at standard EIA RS-232-C voltage levels. Optionally, the two data lines may also be operated as current loops, so that standard teletypes may be plugged into the same connectors as EIA terminals. (See Appendix C).

These features enable the MIGHTY-MUX to control up to 128 devices, including combinations of local and remote terminals and peripherals of many different kinds. Further, all these devices may be operated at any standard Baud rate up to 9600 Baud, under software control.

## 1. General Description

The MIGHTY-MUX Multiplexer (hereafter sometimes referred to as Mux) interfaces up to 128 I/O devices to a Nova type computer. The heart of the system is the EDS-310 board which contains all the Common Logic and 8 interface Ports (see Figure 1). Each Port contains the necessary control logic and buffering for full-duplex operation. The Common Logic contains a sequencer which interrogates the ports in turn, and logic to service them when they need it, providing direct access to core memory via the DMA channel and generating interrupts to the program when appropriate. Up to 120 more ports may be added by means of EDS-301 expansion boards. Each 301 board contains 8, 16, or 24 ports. Thus a full 128 port system requires one 310 and five 301 boards.

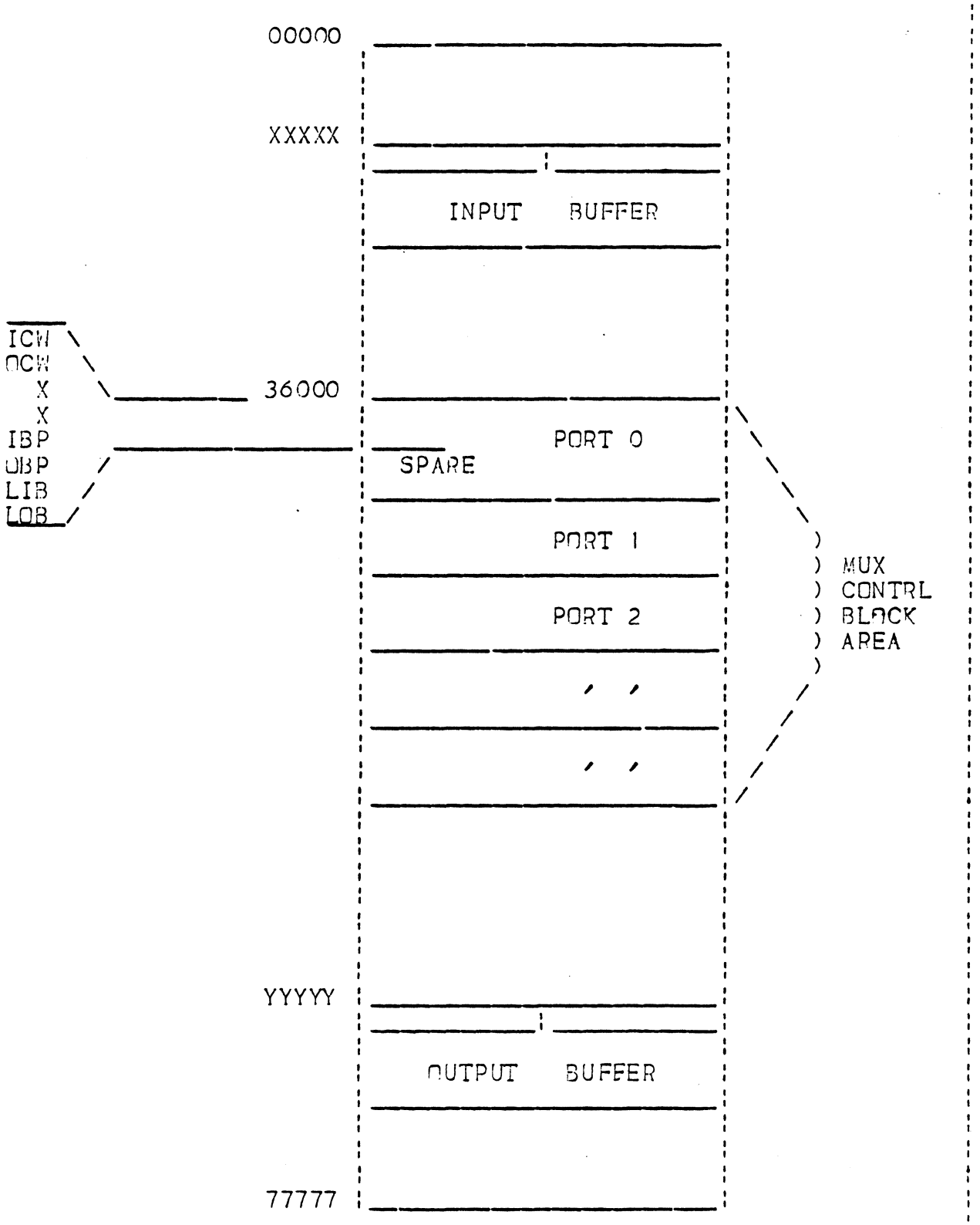



Figure 2. Core Allocation for Control Block and Input and Output Buffers

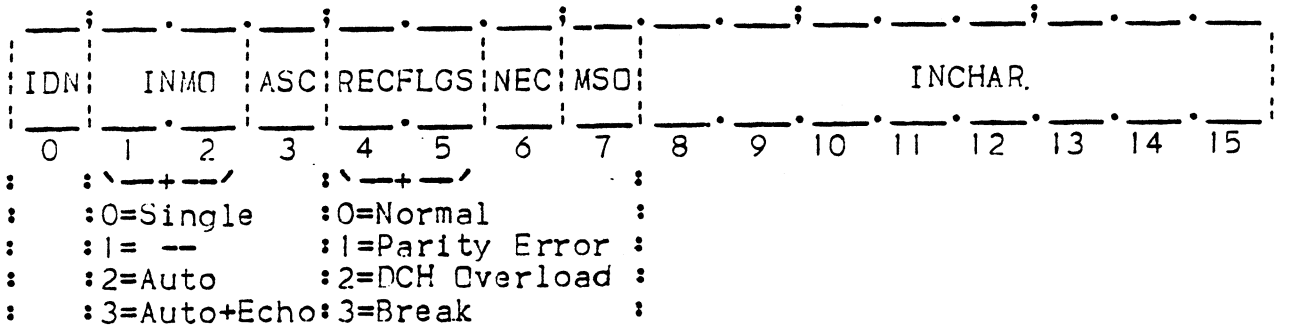
## 2. Core Allocation Requirements

To control the MIGHTY-MUX, the program stores appropriate I/O command words in a certain dedicated area in core, consisting of a control block for each port used. Core Allocation of Mux Control Block and of Input and Output Buffers is as shown in Figure 2. All control blocks are contiguous in core, and may start at any multiple of 400 (octal). This starting location is under software control. In the absence of appropriate software control, the Mux control block starting location will be set by hardware to a predetermined default value which has been preset by jumpers on the Mux board (see Appendix B, Section B.5). All control blocks must be the same size. This size may be 8, 16, or 32 words (10, 20, or 40 octal).

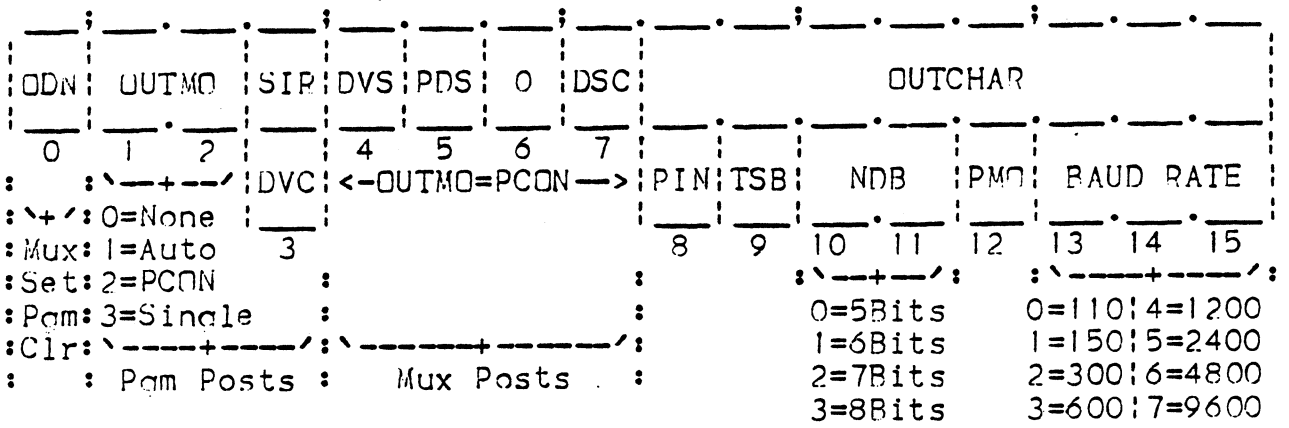
Within each port control block (PCB), 6 words are used for Mux control - namely words 0, 1, 4, 5, 6, and 7; the remainder may be used for any other purpose. The three even-numbered words control input, the odd-numbered ones control output. In each set of three, one word is a general control word and the other two are pointers to the beginning and end of the desired buffer areas in core. The sizes and locations of these buffers are therefore entirely under software control. Several ports may transmit from the same buffer area at one time, since each keeps track of its own pointers. If the port is operated in single-character mode, the pointer words are not read by the Mux and may be used for any other purpose. The meanings and formats of the six control words are shown in Figure 3 and explained in Section 4.

The MIGHTY-MUX is normally delivered wired for 32-word control blocks. This may easily be changed to 16 or 8 by rewiring some jumper wires as described in Appendix B, Section B.2.

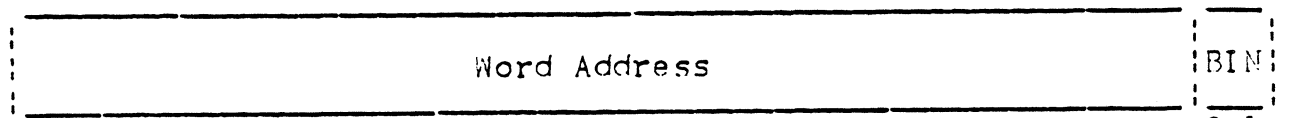
Word 0 - ICW - Input Control Word



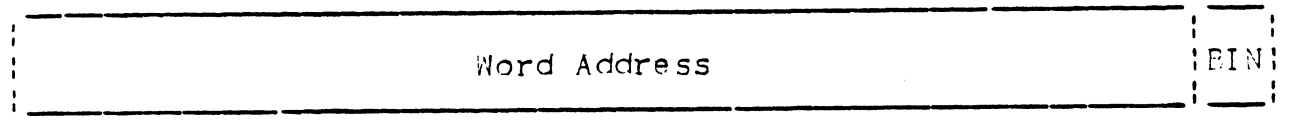

Word 1 - OCW - Output Control Word



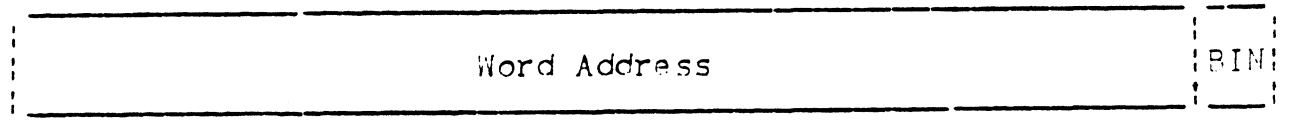

Word 4 - IBP - Input Byte Pointer




Word 5 - OBP - Output Byte Pointer




Word 6 - LIB - Last Input Byte




Word 7 - LOB - Last Output Byte

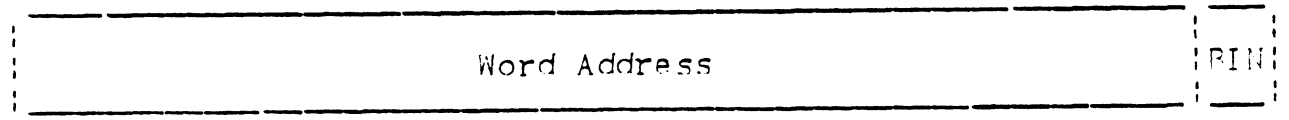



Figure 3. MIGHTY-MUX Control Words

## 3. Overall Sequence of Operation

In order to start the Mux, it is necessary for the program to give a DOC -,MUX instruction (the data output is immaterial). The Mux then cyclically tests all ports, checking for an Input Request (i.e., a character has been received) or an Output Request (i.e., port is ready to accept next output character). When one of these requests is found, the Mux stops at that port and reads the Input Control Word or Output Control Word belonging to that port. Depending on the instructions contained in this control word, the Mux then carries out the indicated operations, including accessing the automatic buffer if appropriate. When all required actions for the sensed Input or Output Request are completed, the Mux continues on to test the next port, etc. If both Input Request and Output Request are true simultaneously, Input takes precedence and Output is deferred to the next time that that port is inspected. These steps are shown in the Flow Chart comprising Figure 4.

When an Input or Output process is completed, the Mux sends an interrupt to the CPU. An interrupt may also be given by the Real-Time Clock on the MIGHTY-MUX, once each 10 msec (100 times per second). The program, by giving a DIAS-, MUX instruction, can then clear the interrupt and simultaneously read the Mux Status Word, which indicates what type of interrupt was given.

## 4. Definition of Control Words (See Figure 3)

### 4.1 ICW - Input Control Word (Word 0 of each control block)

Bit 0 - Input Done (IDN). Set by the Mux under the following conditions:

1) If in Single Character Input mode, when an incoming character is received and stored in Bits 8-15.

2) If in Automatic Input mode (with or without automatic echo) when the assigned input buffer is full; i.e., after an incoming character is placed in the byte specified by LIB.

3) If any of the following three cases is detected by the receiver (see Bits 4 and 5 below).

   Parity error - if Parity Inhibit is 0 (see OCW Bit 8)
   Data Channel overload
   Break (framing error)

4) If in Automatic Input With Echo mode but automatic echo was not accomplished (see Bit 6 below).

5) If Special Interrupt Request is enabled, and a "Special" character (see Bit 3 below) is received.

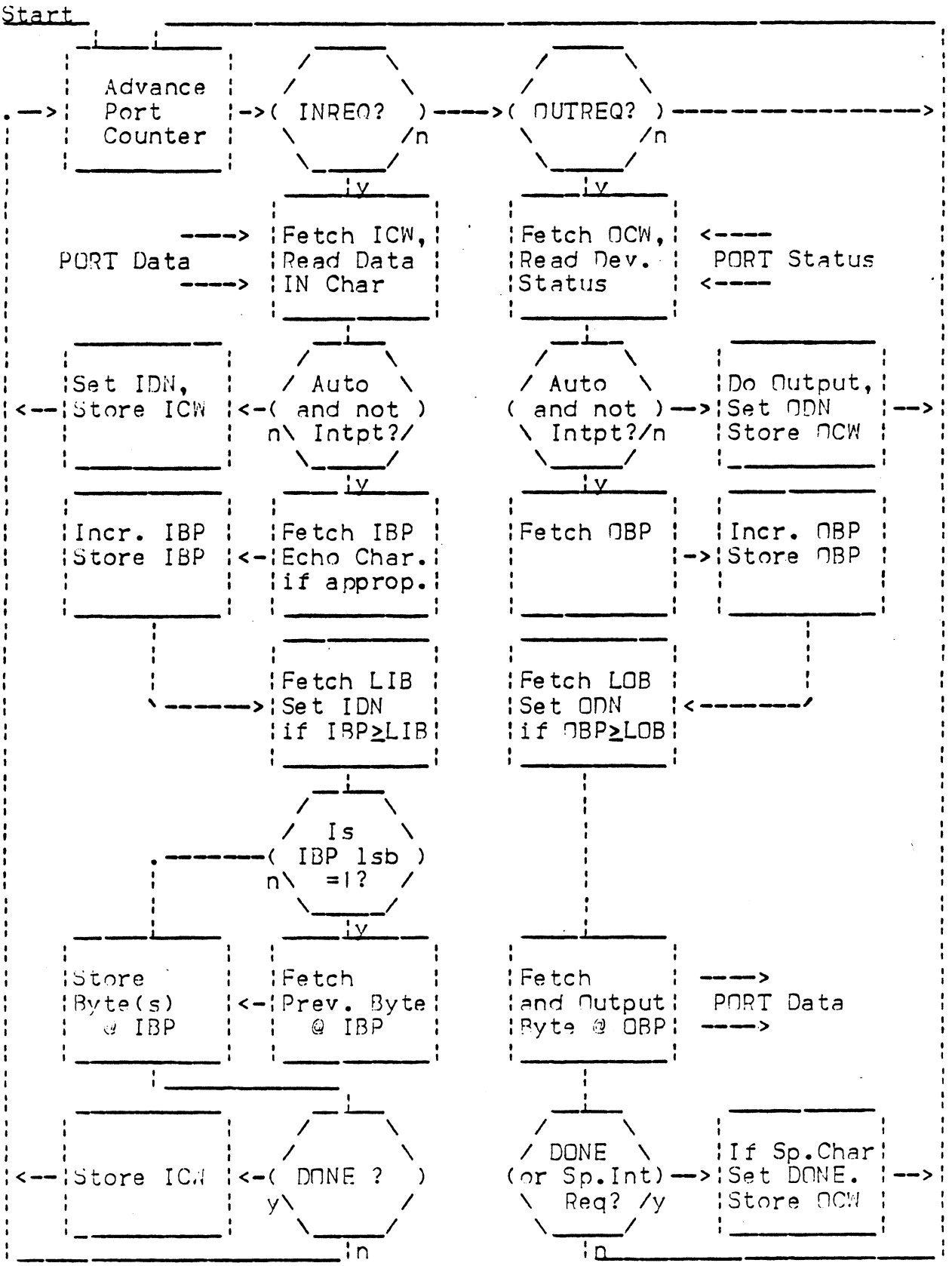



Figure 4. Flowchart Showing Processing of Inputs and Outputs.

At the same time that IDN is set, an interrupt is also generated.

Input Done must be cleared by the Mux interrupt service program. There is approximately one character time available for this. If another input character is received while Input Done is still set, it will override the character in Bits 8-15 (in this case automatic input is not permitted), and Bit 7 is set indicating that an incoming character has been lost, but a second interrupt is not generated. If it is desired to inhibit any interrupts from a particular port, this can be accomplished by setting IDN to 1.

Bits 1 and 2 - Input Mode (INMO). Written by the program and read by the Mux to determine what type of input will be done.

| 1 | 2 | | |
|---|---|---|---|
| 0 | 0 | = | Single Character Input mode - i.e., each incoming character is placed in Bits 8-15 of ICW, IDN is set, and an interrupt given. |
| 1 | 0 | = | Automatic Input - i.e., incoming characters are placed in the input buffer defined by IBP and LIB, as long as no interrupt conditions are encountered (see Bit 0 for interrupt conditions). |
| 0 | 1 | = | Not used (illegal). |
| 1 | 1 | = | Automatic Input With Echo - same as Automatic Input except that each character which is placed in the input buffer is also automatically echoed (output). Note that any character producing an interrupt (other than buffer-full) is not automatically echoed. |

Bit 3 - 7-bit ASCII Mode Special Interrupt Request (ASC). Written by the program.

If this bit is set, the Mux will examine every incoming character and generate an interrupt if it is a "Special" character. The set of special characters is determined by a PROM (Programmable Read-Only Memory) on the 310 board and may be any set desired by the user. The standard set consists of all characters below octal 40 or above octal 173; for available optional sets see Appendix B, Section B.7. Special characters are not stored in an automatic buffer but in ICW and produce interrupts. This allows immediate program response to such characters as backspace, carriage return, end-of-message, etc.

A second effect of the ASC bit is that when this bit is on, the Mux sets the most significant bit of each incoming character to a one. The intent is to allow the software operating system to distinguish incoming 7-bit ASCII data characters from non-data by means of the msb. If this feature is not desired, see Appendix B, Options, Section B.8.

Bits 4 and 5 - Receiver Flags (RECFLGS). Posted by the Mux each time an input is received.

| 4 | 5 | | |
|---|---|---|---|
| 0 | 0 | = | Normal - i.e., none of the three cases below. |
| 0 | 1 | = | Parity Error - if Parity is not inhibited (see OCW Bit 8 below) |
| 1 | 0 | = | Data Channel Overload - i.e., an input character was received before the previous character from that port could be stored away by the Mux. This can only happen if another interface on the computer's data channel, having higher priority than the Mux, has been taking most of the core memory cycles. |
| 1 | 1 | = | Break, or Framing error - i.e., a Stop Bit was not received. This is used for detecting the Break character, which is a 0 character with no Stop Bit. |

Each of the latter three cases produces an interrupt and sets Input Done. If more than one of these occurs at the same time, only the one with the highest number is recorded.

Bit 6 - Not Echoed (NEC). Posted by the Mux if in Automatic Input With Echo mode, and if automatic echo was not possible because the output circuitry was still busy with a previous output. This can only happen if the program started an output while the Mux was in Automatic Input With Echo mode - a condition that should never occur. At the same time that this bit is posted, Input Done is also set and an interrupt generated.

Bit 7 - Mux Service Overload (MSO). This bit is set by the Mux if at the time an input character is stored the Input Done flag is still on from a previous input. The length of time available for the Mux service program to service an input interrupt before the next input could come in on the same port is on the average equal to one character transmission time (about 1 msec for 9600 Baud data rate), but could in the worst case be as much as 300 usec less than that.

Bits 8-15 - Input Character (INCHAR). Every incoming character which produces an Input Done is stored in these bits (see "Exception" below). In automatic buffer mode characters which are stored in the automatic buffer are not stored in ICW, except that the last character, which produces the buffer-full condition, is stored in both places (see "Exception" below).

Exception: If an input buffer ends in mid-word, the byte placed in ICW is the right half of the last buffer word and not the last incoming character. The software must therefore check for "Buffer-full" before it checks for "Special character".

## 4.2 OCW - Output Control Word (Word 1 of each control block)

Bit 0 - Output Done (ODN). Set by the Mux under the following conditions:

1) If in Single Character or Port Control Output mode, when the output byte is read for transmission to the port. The program then has a minimum of one character time, and typically almost two character times, to reload OCW if uninterrupted output is desired.

2) If in Automatic Buffer Output mode, when the last byte of the automatic buffer is read by the Mux for transmission. Again, if the output control words are reloaded within 1-2 character times, uninterrupted output will result.

3) If in Automatic Output with Special Interrupt Request mode, and if the transmitted character is a Special Control Character (see Section 4.1, Bit 3). In this event automatic output is terminated after transmitting the special character.

4) Regardless of output mode, if the Device Status Changed bit (Bit 7) becomes a 1.

At the same time that ODN is set, an Interrupt is also generated. Output Done must be cleared by the program before another output can take place on that port. Leaving ODN set inhibits further interrupts from that port, even if Device Status changes.

Bits 1 and 2 - Output Mode (OUTMD). Written by the program and read by the Mux to determine what type of output will be done.

| 1 | 2 | | |
|---|---|---|---|
| 0 | 0 | = | None - no output. |
| 0 | 1 | = | Automatic buffer output - i.e., Mux will output from automatic buffer defined by OBP and LOB. |
| 1 | 0 | = | Port Control Output (PCON) - i.e., the data in Bits 3 and 8-15 are sent to the port as control data, to do their assigned control functions. |
| 1 | 1 | = | Single character output - i.e., the data byte in Bits 8-15 is sent to the port for transmission to the external device. |

Bit 3. The meaning of Bit 3 depends on the Output Mode. If OUTMO = 00 (None) or 11 (Single), Bit 3 has no effect.

If OUTMO = 01 (Auto):

Bit 3 - Special Interrupt Request (SIR). Written by the program. If this bit is a 1, the Mux will test each outgoing character to determine if it is a Special Control character (see Section 4.1, Bit 3). If it is, the Mux will still transmit it, but also set ODN, produce an interrupt, and terminate automatic output. This is useful in driving printers which require some special service after a Carriage Return character, such as a Line Feed, or a delay time, or a Print Line command.

If OUTMO = 10 (PCON):

Bit 3 - Device Control (DVC). Written by the program and transferred by the Mux to the port when the Port Control Output is done. This bit is stored by a flip-flop in the port circuitry and applied as an EIA level on the Device Control Line going to the external device. It may be used for any desired function - it does not interact with the functioning of the port in any other way. For Bell System 103 Type Data Sets, this signal is used as the Data Terminal Ready line (circuit CD). For half-duplex control, this signal is used as the Request To Send (circuit CA).

1 = + 10 volts = EIA Positive = Function On

0 = - 10 volts = EIA Negative = Function Off

Bit 4 - Device Status (DVS). Written by the Mux each time OCW is inspected whether any output is done or not.

This is simply the current value of the Device Status line coming from the external device. This line may be used for any desired function - it does not interact with the port in any other way. For 103 Type Data Sets, this line is used for the Received Line Signal Detector (circuit CF). For half-duplex control, this line is used for the Clear To Send (circuit CB).

1 = EIA Positive, i.e. > +3 volts

0 = EIA Negative, i.e. $\leq$ 0 volts, or open

Note: Actually, any voltage above +1.3 v will produce a 1, and any voltage below +0.7 v will produce a 0; voltages between these limits are indeterminate.

Bit 5 - Previous Status (PDS). Written by the Mux each time OCW is inspected whether any output is done or not. The value read from Bit 4 is rewritten here. This bit is of no significance to the programmer.

Bit 6 - Unused. Always is made zero by the Mux. (Exception: see Appendix A, "EDS-302 Synchronous Port Card").

Bit 7 - Device Status Changed (DSC). Written by the Mux whenever the device status line does not equal the Device Status bit (Bit 4) read from the OCW. When this bit is set, Bit 0 (Output Done) is also set and an interrupt is generated. Note that this imposes a requirement on the Mux Service Program, whenever it writes an OCW, to leave Bit 4 the way it was - otherwise the Mux will think Device Status has changed and produce an interrupt.

Bits 8-15 - Output Character (OUTCHAR). If Output Mode = 11 (Single character), this byte is sent to the port for transmission to the external device. If OUTMO = 00 or 01 (Inactive or Automatic), this byte is ignored. (Exception: In Automatic Output with Special Interrupt Request, this byte must not be a Special Character - see Section 4.1, Bit 3 for definition of "Special Character"; otherwise the Special Character detection logic will never permit automatic output to get started.) When an ODN interrupt is produced, the Mux will store the character that produced the interrupt in this byte. If OUTMO = 10 (Port Control Output), this byte is sent to the port as a control character, governing both output and input. In this case, the meanings of these eight bits are as follows.

Bit 8 - Parity Inhibit (PIN).

0 = Parity is generated (transmit) and checked (receive)
1 = No Parity

Bit 9 - Stop Bit Selector (SBS).

0 = One Stop Bit
1 = Two Stop Bits

Note: This governs output only. In input, one Stop Bit (in fact, just over one-half Stop Bit) is always adequate.

Bits 10 and 11 - Number of Data Bits (NDB).

| 10 | 11 | | |
|---|---|---|---|
| 0 | 0 | = | 5 Bits |
| 0 | 1 | = | 6 Bits |
| 1 | 0 | = | 7 Bits |
| 1 | 1 | = | 8 Bits |

This field governs the data bits only; parity (if not inhibited) is an additional bit. If less than 8 bits are selected, the character will be right justified, and the unused most significant bit(s) will be set to zero on input (exception: ASC mode - see Input Control Word Bit 3). For output, the character must be right justified, but the unused most significant bits are ignored.

Bit 12 - Parity Mode (PMO) (has effect only if Bit 8 = 0).

0 = Odd Parity
1 = Even Parity

Bits 13-15 - Frequency of transmission/reception.

| 13 | 14 | 15 | | |
|---|---|---|---|---|
| 0 | 0 | 0 | = | 110 Baud (standard Teletype) |
| 0 | 0 | 1 | = | 150 Baud |
| 0 | 1 | 0 | = | 300 Baud |
| 0 | 1 | 1 | = | 600 Baud |
| 1 | 0 | 0 | = | 1200 Baud |
| 1 | 0 | 1 | = | 2400 Baud |
| 1 | 1 | 0 | = | 4800 Baud |
| 1 | 1 | 1 | = | 9600 Baud |

4.3 IBP - Input Byte Pointer. (Word 4 of control block). This word and its partner (LIB) are only used if in Automatic Input mode. IBP must be set up by the program to 1 less than the first byte address (see Section 7.3) of the automatic input buffer. Each time the Mux stores an incoming byte, it will increment IBP. The Mux then uses the most significant 15 bits of the IBP as a core word address and the least significant, (bit 15) as the Byte Indicator (BIN); i.e. the byte will either be stored in the left or right half of the word addressed, depending upon the value of BIN (BIN = 0 = right byte, BIN = 1 = left byte). Thus IBP always points to the last input byte stored unless no input bytes have been received yet. If IBP $\geq$ LIB, an interrupt will be generated by the first incoming character, but that character will be stored at IBP.

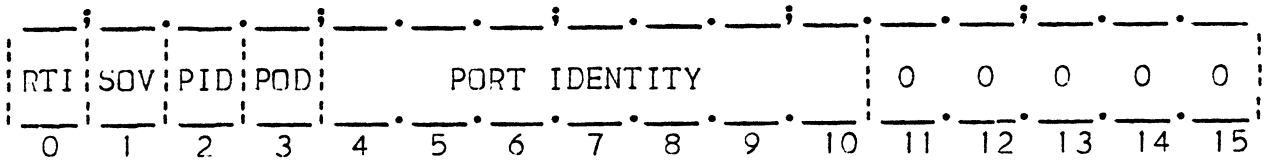
Note: When storing the first (left) byte in a word, the Mux writes that byte in both halves of the word (unless this is the last byte in the buffer). When storing the second byte, it overwrites the right half of the word. If an input buffer ends in mid-word, the right half of that word is correctly preserved.

4.4 OBP - Output Byte Pointer. (Word 5 of the control block). This word and its partner (LOB) are used only if in Automatic Output mode. OBP must be set up by the program to 1 less than the byte address of the first byte of the automatic output buffer. Each time the Mux is ready for an output byte, it will increment OBP and then fetch the byte for transmission from the appropriatehalf of the word address given by the most significant 15 bits of the OBP (word address) in conjunction with the Byte Indicator (BIN). Thus OBP always points to the last byte that has been transmitted, if any.

4.5 LIB - Last Input Byte. (Word 6 of control block). Set up by the program to the last byte address of the auto input buffer. The Mux will generate an Input Done interrupt when a byte is stored at this address. IBP will then be equal to LIB.

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

4.7 Mux Status Word (MUXSTWD).




This is not one of the control words in each port control block, but it is the word which is read in when the program gives a DIA -,MUX instruction. The Mux Status Word contains information indicating what type of interrupt was given. The Mux contains a stack (FIFO - First In, First Out) where up to 40 interrupts can be queued up. This stack is popped when a START pulse is given by the program. Therefore, by means of a DIAS -,MUX instruction the program can "simultaneously" read the Mux Status Word, and pop the stack. If the stack is then not empty, the Mux DONE flag will remain set and another interrupt will be generated. If the stack is empty, a DIA -,MUX instruction will read in a word with bits 0-3 all equal to 0, but bits 4-10 may not be 0.

Bit 0 - Real-Time Interrupt (RTI - 100 Hz). If the Real-Time Clock is enabled, once each hundredth of a second the Mux will give an interrupt with this bit on (1). Any additional interrupts will have this bit off (0). This bit is cleared by the DIA -,MUX instruction which reads it, whether accompanied by an S pulse or not.

Bit 1 - Stack Overflow (SOV). If this bit is a 1 it indicates that the Mux interrupt stack has overflowed ($\geq$ 40 interrupts pending), and therefore the software must inspect all Input and Output Control Words to determine which ones require service. In this case the remainder of the Mux Status Word is meaningless. If this bit is 0, then the following definitions apply.

Bit 2 - Port Input Done (PID). The port whose identity appears in Bits 4-10 has generated an IDN interrupt.

Bit 3 - Port Output Done (POD). The port whose identity appears in Bits 4-10 has generated an ODN interrupt.

Bits 4-10 - Port Identity. This field identifies the port reporting an IDN or ODN interrupt. The "identity" of a port consists of the 7 bits which specify the 128 possible port control blocks: i.e. bits 4-10 of the port's control word addresses if the control block size is 32 words, or bits 5-11 for 16-word control blocks, or bits 6-12 for 8-word control blocks.

Bits 11-15. Always 0.

## 5. CPU Control Over Mux

The CPU controls Mux operation by means of the following I/O instructions.

**IORST or CLEAR (eg NIOC)** Turns off both Mux and Real-Time Clock, clears DONE and BUSY flags, and terminates any output that may have been in process by setting all data output lines to -10 volts. IORST also has the following additional effects:

a) Sets all Device Control lines to +10 volts,

b) Sets Baud rate of all ports to 110 Baud,

c) Sets Port 0 to 8 bit character length, no parity, 2 stop bits,

d) Sets the Port Control Block base address to its default value.

The CLEAR pulse does not have these additional effects, in order to permit the software to set up any desired port parameters, then shut down the Mux and let Port 0 be the master terminal interface (Device code 10/11 type - See Appendix B, Section B.3).

**IOPLS (eg NIOP)** Starts Real-Time Clock. Has no effect on Mux action.

**START (eg NIOS)** Pops Mux Status Word stack. Thus, a DIAS -,MUX instruction will "simultaneously" read and pop the MUX Status Word stack. If the stack is then empty, the Mux DONE flag will be cleared. If stack overflow has occurred, the START pulse clears the stack.

Note: Some CPU's (such as the Nova 2) may allow a data channel cycle between the DIA and S pulses of a single DIAS instruction. To guard against this, the DIAS should be preceded by a DOB -,MUX and followed by a DOC -,MUX.

**DOA -,MUX** If given before the Mux is turned on, this instruction sets up the base address of the PCB area. Only the 8 msb are used; therefore the PCB area must start at a multiple of 400 (octal). Once the Mux has been turned on, the effect of the DOA instruction (regardless of data output) is to "prompt" the Mux to inspect all OCW's for any pending output commands (in the absence of the DOA, the OCW's of all inactive ports are inspected once each 50 milliseconds). This can be used to speed up output to line printers, for example.

DOC -,MUX  Starts the multiplexer proper. Thus, DOCP -,MUX will start both the Mux and the real-time clock. (Only the DOC signal is used - it does not matter what data is output by this instruction.)

DOB -,MUX  Stops all Mux data channel transfers, but does not interrupt any input or output already in process. Because the Mux takes successive data channel cycles when processing a particular port, the DOB instruction will always stop the Mux after completion of one port service and before beginning another. The DOB/DOC pair may be used whenever a control word has to be written, to guard against the possibility that the Mux has just written into that control word. For example, the following sequence may be used in the Mux Output Routine when a new OCW is to be written.

```
LDA     3,BASE   ;base address of port control block
LDA     1,C4000  ;mask for bit 4, Device Status
LDA     2,NOCW   ;next OCW desired to be written
INTDS            ;disable intpts. for min. Mux pause
DOB     0,MUX    ;pause Mux operation
SKPDN   MUX      ;is Mux trying to interrupt?
JMP     .+4
DOC     0,MUX    ;  yes - turn Mux back on
INTEN            ;    and let it interrupt
JMP     .-6      ;    try again after the interrupt
LDA     0,1,3    ;  no - pick up OCW
AND     0,1,SZR  ;pick out the Device Status bit
ADD     1,2      ;insert device status in new OCW
STA     2,1,3    ;store new OCW
DOC     0,MUX    ;continue Mux operation
INTEN            ;enable interrupts
```

This sequence guards against the possibility of the Device Status changing just before the program stores a new OCW. A similar sequence may be used in the Mux Input Service Routine to guard against an incoming character being accidentally overwritten by the program.

The DOB/DOC pair should also be used around a DIAS -,MUX instruction (see Note under "START" above).

DIA -,MUX  Reads in the Mux Status Word (see Section 4.7), and clears the Real-Time Clock interrupt bit if it was on.

6. Initialization (when power is first turned on)

1) Give an IORST, or NIOC -,MUX.

2) Give a DOA -,MUX if it is desired to change the Core Location of the Control Block Area from its hard-wired default value (generally 36000). See Appendix B, Section B.5 .

3) Set up each OCW for initial Port Control Output (see Section 4.2). For example, for a data set ready for automatic answering at 110 Baud with two Stop Bits and using 7-bit characters with even parity, OCW would be 50150:

| Out Don | OUTMO =PCON | Dev Con | Dev Sta | Pre Sta | 0 | Sta Chg | Par Inh | Two Stp | Char.L. =7 bits | Evn Par | Freq. =110 Baud |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 0 | 1 | 0 0 0 |
| 0 | 5 | | | 0 | | | 1 | | 5 | | 0 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 11 | 12 | 13 14 15 |

4) Set up each ICW for the desired input mode. Initially this will generally be Single Character Input. Therefore, ICW would be set to 0 (or 10000 if the msb is desired to be set to 1).

5) Give a DOC -,MUX. This starts the Mux, setting its BUSY flag. BUSY remains set from here on until an IORST, CLEAR, or DOB-,MUX is given: interrupts cause DONE to come on but do not clear BUSY. The Mux will now do all the indicated port control outputs, setting the Output Done bit for each and also setting the Mux DONE flag.

6) Enable Interrupts. As soon as the Mux DONE flag is set, an interrupt will result. The interrupt service will then, by means of DIAS -,MUX instructions, pop Mux Status Words off the interrupt stack until all Output Done bits are processed.

## 7. Input and Output

### 7.1 To Input in Single Character Mode

1) Set ICW to: (see Figure 3)

   10000 for msb = 1
   0 for normal input (unmodified)

2) When an incoming character arrives, it will be stored in ICW, the IDN bit will be set, and an interrupt will be given.

### 7.2 To Output in Single Character Mode

1) Set OCW to:

   6x000 plus the desired character, where x = 4 or 0 according as the Device Status bit (Bit 4) is 1 or 0.

2) An interrupt will be given and ODN set when the character is picked up for transmission.

### 7.3 To Input Automatically into a String Buffer

1) Set IBP to 1 less than the byte address of the first byte in the string. Note: Byte Address equals Word Address shifted one place to the left with the least significant bit = 0 for left byte or = 1 for right byte.

2) Set LIB to the byte address of the last byte of the string buffer. Note: The string buffer may end in the middle of a word without destroying the contents of the right half of that word.

3) Set ICW to:

   40000 for no echo and no ASC
   50000 for no echo but ASC
   60000 for automatic echo and no ASC
   70000 for automatic echo and ASC

4) An interrupt will be given and IDN set if:

| Condition | Indication |
|---|---|
| Buffer full | IBP = LIB |
| Parity Error | ICW Bits 4,5 = 0 1 |
| Data Channel Late | ICW Bits 4,5 = 1 0 |
| Break | ICW Bits 4,5 = 1 1 |
| Auto echo ordered but not accomplished | ICW Bit 6 = 1 |
| Special Character if ASC set (Special Interrupt Req) | Character is in ICW Bits 8 - 15 |

Note: After the buffer is full, if IDN is cleared but ICW is left set up for automatic input, then if another character is received, it will produce an interrupt but it will also be stored in the next byte address (i.e. produce buffer overflow).

7.4 To Output Automatically from a String Buffer

1) Set OBP to 1 less than the byte address of the first byte to be output.

2) Set LOB to the byte address of the last byte to be output.

3) Set OCW to:

2x000 for regular automatic output
3xyyy for interrupts on special control characters, where

x = 4 or 0 according as Device Status = 1 or 0
yyy = any value that is not a "Special" character (see Section 4.1, Bit 3)

Note: If x is set to the wrong value, an immediate interrupt will be produced without transmitting even one character.

4) An interrupt will be produced and ODN set if:

| Condition | Indication |
|---|---|
| Buffer empty | OBP = LOB |
| Device Status changed | OCW Bit 7 = 1 (OBP points to last character transmitted) |
| Special character if special interrupt requested (SIR) | Character is in OCW Bits 8-15 (OBP also points to it in buffer) |

7.5 To Change Port Control Parameters

A new Port Control mode may be set up at any time by simply storing the desired Port Control Word (see Section 4.2) in OCW. Thus, for example, Baud rate, or parity mode, or Device Control may be changed whenever the program wishes.

If it is desired to issue a new Port Control Word (PCON) after completing a transmission, it is necessary to add a null character at the end of the desired output string. The program must then wait for the Output Done condition resulting from this null character being picked up for transmission, before it stores the new Port Control Word in OCW. This is due to the double buffering in the port logic, and applies to both single character mode and automatic output mode.

8. Interrupt Service

1) Give a DIAS -,MUX (see Note under "START" in Section 5). This reads in the Mux Status Word (see Figure 2), and clears the Mux DONE flag (unless there is another interrupt pending).

2) Test MUXSTWD Bit 0. If 1, do real-time interrupt service. This step is, of course, not needed if the real-time clock has not been turned on.

3) Test Bit 1. If 1, check all ICW and OCW to determine if service is required, then go to 6. If 0, continue.

4) Test Bit 2. If 1, do Input Service for the port whose identity appears in bits 4-10, then go to 6.

5) Test Bit 3. If 1, do Output Service for the port whose identity appears in bits 4-10.

6) Give another DIAS -,MUX. If bits 0-3 are all 0, exit interrupt service, otherwise go back to 2.

APPENDIX A

# EDS-302 SYNCHRONOUS PORT CARD

## SPECIFICATION

The EDS-302 is a 15" x 15" board that occupies one slot in any Nova-type computer and serves as an expansion board for an EDS-310 MIGHTY-MUX data channel multiplexer. The EDS-302 contains up to 8 synchronous ports in increments of 2 (options -S2, -S4, -S6, or -S8). Each of the synchronous ports offers the following capabilities.

### A.1 Functional Capabilities

1) Receive and transmit synchronous data at any rate up to 50,000 baud (clocked by modem)

2) CRC (Cyclic Redundancy Code) generation and checking performed by hardware

3) Compatible with ADCCP/HDLC/SDLC:

   a. Flag sequence, Zero-bit insertion/extraction, and Frame Check Sequence implemented in hardware

   b. Address, Control, and Information Fields under software control

4) Characters (8 bits each) are packed 2 per word (left/right) in core memory

5) Three outgoing modem control lines under program control:

   Request To Send
   Data Terminal Ready
   Secondary Request To Send

6) Five incoming modem control lines available to program:

   Clear To Send
   Data Set Ready
   Ring Indicator
   Received Line Signal Detector (Carrier)
   Secondary Received Line Signal Detector

7) All data and control lines operate in accordance with EIA spec RS-232-C

8) Full or Half Duplex, with on/off secondary channel

## A.2 Theory of Operation

Figure A-1 contains a functional block diagram of the synchronous port, and shows the data flow between the modem and the MIGHTY-MUX Common Logic.

In transmission, data output coming from the Mux in 8-bit parallel form is converted to serial form by the Parallel-Serial Converter. When the port is in the idle state, it sends a stream of Flags and presets the CRC generator to all 1's. When the Mux sends the port data to transmit, the CRC generator starts generating the CRC, and the Zero-bit Insertion logic scans the bit stream, inserting a zero bit after each sequence of 5 ones. When the Mux sends the port an EOF (End of Frame) signal, the port transmits the complement of the 16-bit CRC (Frame Check Sequence) followed by at least two Flags (01111110 binary, or 176 octal), and reverts to its idle state. The CRC polynomial is x16 + x12 + x5 + 1, as specified by ADCCP. The Mux may also send the port an Abort signal, in which case the port sends out 8 ones (377 octal)* followed by Flags. The Zero-bit Insertion logic is deactivated when sending Abort or Flag, and is active at all other times.

In the Receiver section, the Zero-bit Extractor logic always extracts a zero following five ones, and also contains Flag and Abort recognition logic. When first turned on (idle state) the CRC generator is cleared to all ones and remains so as long as only Flags or Aborts are received. When some other character is received (following a Flag and framed by it) the CRC generator is enabled. The incoming serial data is converted into parallel characters by the Serial-Parallel Converter. Each assembled character is sent to the Mux. This process can end in one of two ways. If a Flag is received, the port will check the contents of the CRC generator and send an EOF signal to the Mux together with an indication of whether the CRC was good or bad. If an Abort is received*, the port sends the Mux an Abort signal. The Receiver section then reverts to its idle state. Note that while the Transmitter sends at least two Flags between successive frames, the Reciever requires only one. In addition to the above, the software has access to the following EIA control lines:

CA Request To Send (RTS)
CB Clear To Send (CTS)
CC Data Set Ready (DSR)
CD Data Terminal Ready (DTR)
CE Ring Indicator (RING)
CF Received Line Signal Detector (CARR)
SCA Secondary Request To Send (SRTS)
SCF Secondary Received Line Signal Detector (SCAR)

---

* The 302 board implements only one kind of Abort. It does not provide for the "Abort and Idle Link State" defined in Section 3.6 of ADCCP.

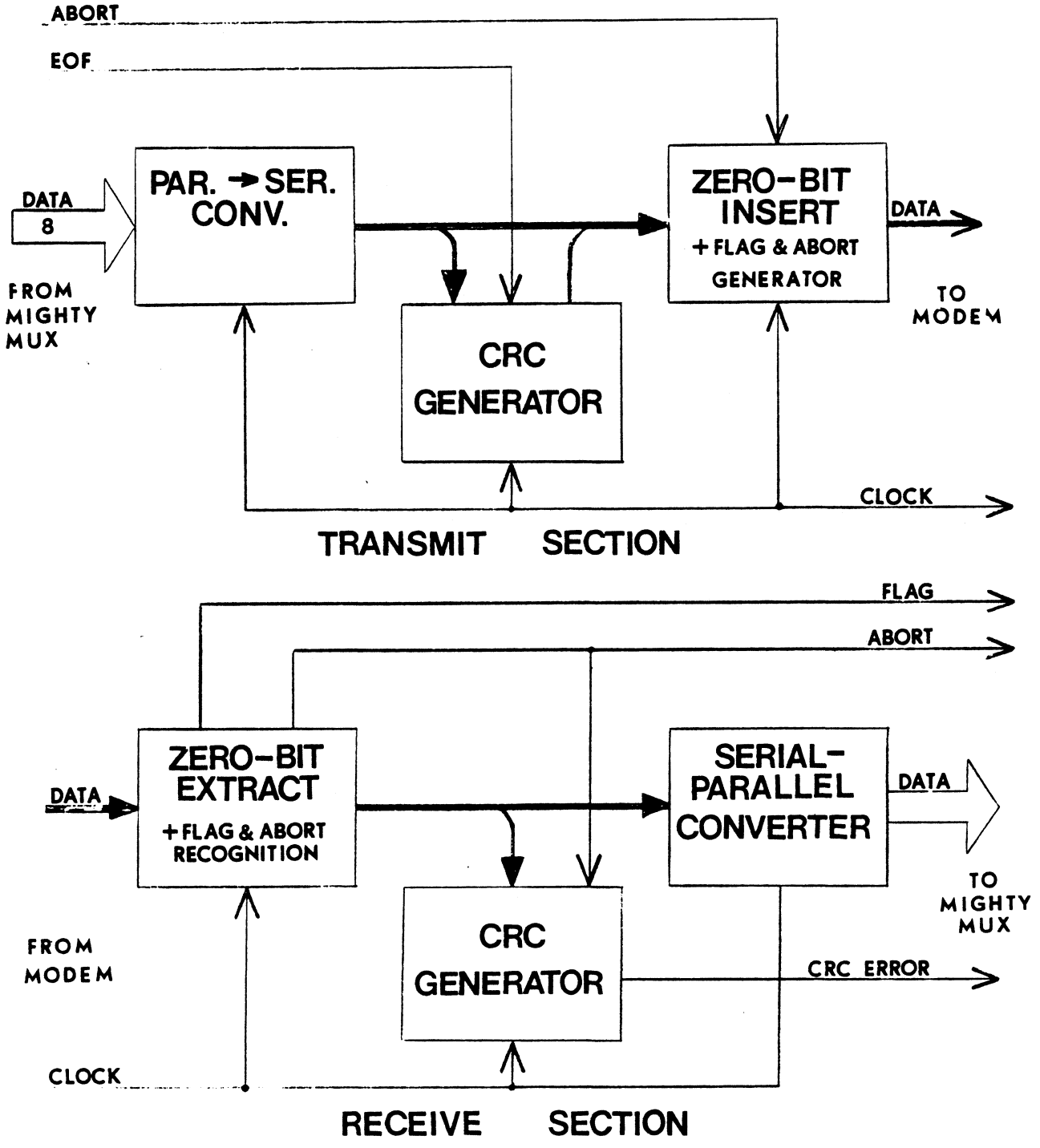



Figure A-1. Functional Block Diagram of ADCCP Port

## A.3 Software Interface

Everything described in the body of this manual for asynchronous ports applies to the synchronous ports, with the exception of certain control bits in the Input Control Word (ICW) and Output Control Word (OCW). These are depicted in Figure A-2. Only the control bits having special meanings for synchronous ports are described below.

### 1) ICW for Sync Port

Bit 4 - Data Channel Overload Same meaning as in asynch ports (see Sect 4.1, Bits 4-5)

Bit 5 - Status Input, i.e. Bits 8-15 contain the Receiver Status Byte (see below). The Status Byte is read in, and produces an interrupt, when any of the following conditions occurs.

a) DSR (Data Set Ready) changes state
b) RING Indicator changes state
c) CARRier changes state
d) SCAR (Secondary Carrier) changes state
e) Flag Sequence or Abort is received
f) Status byte was requested by READ STAT in OCW

Status Byte (Bits 8-15 if Bit 5 = 1):

Bit 8 - DSR (Data Set Ready). Circuit CC of RS-232-C.

Bit 9 - RING (Ring Indicator). Circuit CE of RS-232-C.

Bit 10 - CARR (Received Line Signal Detector). Circuit CF of RS-232-C.

Bit 11 - SCAR (Secondary Received Line Signal Detector). Circuit SCF of RS-232-C

Bit 12 - EOF (End of Frame). Indicates that a Flag sequence (01111110) has terminated a message frame.

Bit 14 - ABORT. Indicates that an Abort Code (7 or more 1-bits in a row) has terminated a message frame.

Bit 15 - CRC ERROR. Indicates that the last 16 bits preceding a closing flag sequence did not contain a valid CRC code for the transmitted frame. Can only be on when Bit 12 is on. Thus a correctly received frame is indicated by Bit 12 = 1 and Bit 15 = 0.

2) OCW for Sync Port

Bits 1 and 2 - OUTMO (Output Mode)

0 0 = None
0 1 = Automatic Buffer Output
1 0 = PCON (Port Control Output)
1 1 = Single Character Mode Output

If OUTMO = PCON, Bits 8-15 contain the Port Control Byte and are defined below.

Bit 3 - The meaning of Bit 3 depends on OUTMO.

| OUTMO | Meaning of Bit 3 |
|---|---|
| AUTO | SIR (Special Interrupt Request) |
| PCON | RTS (Request to Send). Circuit CA of RS-232-C. |
| SINGL | EOF (End of Frame): i.e. send CRC and closing flag after this character. |

Bit 4 - CTS (Clear to Send). Circuit CB of RS-232C.

Bit 5 - Previous CTS. Of no significance to the software.

Bit 6 - ABORT SENT. Indicates that an Abort has been transmitted, because no data was available to transmit (Data Channel Late or Mux Service Overload).

Bit 7 - CTS has changed. This also produces an interrupt.

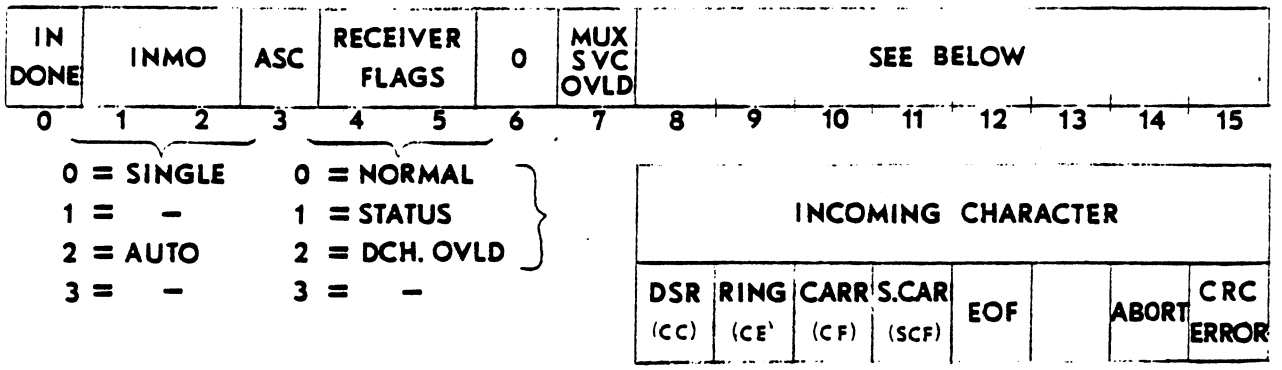
PCON Byte (Bits 8-15 if Bits 1,2 = PCON = 1,0):

Bit 12 - DTR (Data Terminal Ready). Circuit CD of RS-232-C.

Bit 13 - SRTS (Secondary Request to Send). Circuit SCA of RS-232-C.

Bit 15 - READ STAT. Causes the Receiver Status Byte to be read into ICW.

## ICW FOR SYNC PORT




## OCW FOR SYNC PORT

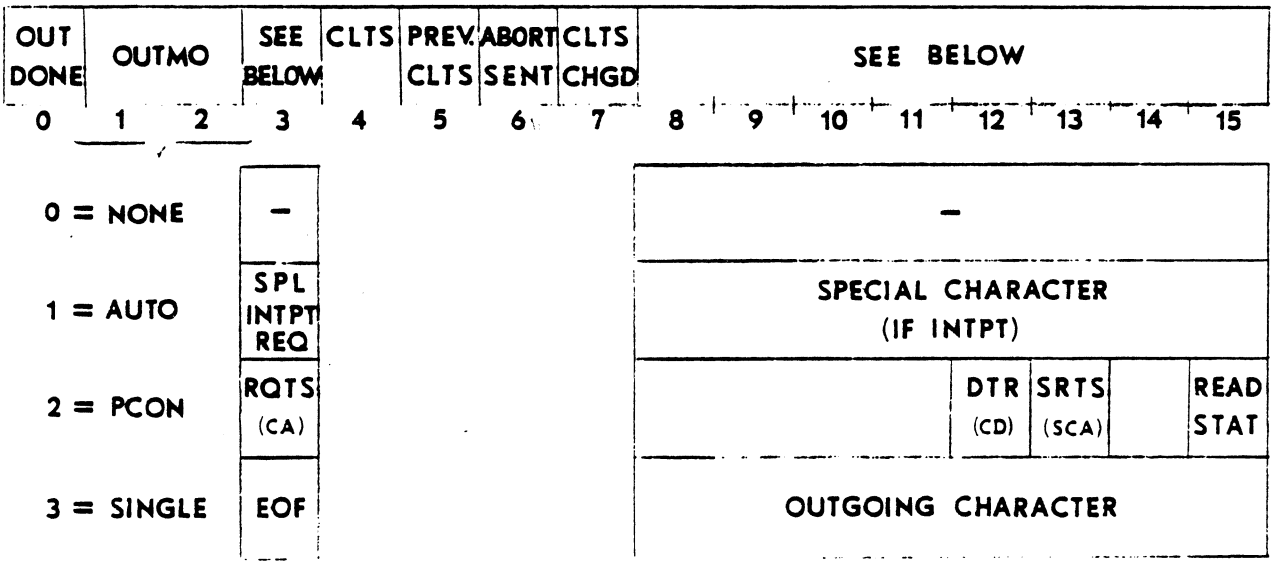



Figure A-2. Input and Output Control Words for Synchronous Ports

## A.4 To Transmit on a Synchronous Port

1) Turn on Request To Send and Data Terminal Ready, by storing the appropriate Port Control Word in OCW.

2) When ODN (Bit 0 of OCW) = 1 (and an interrupt occurs), check Clear To Send (CTS = Bit 4 of OCW). If CTS = 1, go on to Step 3. If CTS = 0, clear the ODN bit and wait for the next interrupt, or for ODN = 1, at which time CTS should be 1.

Note: Steps 1 and 2 are for initialization or line turn-around, and may be omitted when transmitting on an active Full Duplex channel.

3) Set up OBP (Output Byte Pointer) and LOB (Last Output Byte) for the desired automatic output buffer, and then set up OCW for auto output mode.

The Mux will now transmit the contents of the specified buffer, without further intervention from the program.

The transmission of data ends when any of the following occurs, at which time an interrupt is given and ODN is set to 1.

a) Buffer Empty, indicated by OBP = LOB. In this case the port automatically transmits the 16-bit CRC followed by at least two Flags. The software may set up the next output immediately after the ODN interrupt: if so, that output will begin after two Flags, otherwise the port continues to send Flags until the next output is initiated.

b) Special Interrupt was requested (via Bit 3) and a special character was sent - indicated by the special character being in OCW Bits 8-15. To continue uninterrupted transmission, the software must set up the next output within approximately one character time. Otherwise an Abort results.

c) Data Channel Late, or (if in single character mode) Mux Service Slow. Results in aborting the outgoing message, and is indicated by Bit 6 = 1.

d) Clear To Send became low - indicated by Bit 4 = 0. This immediately terminates transmission.

If it is desired to transmit in Single Character mode, this may be done in normal fashion. The interrupt service must be fast enough the produce the next character within approximately one character time of the interrupt from the previous character, otherwise an Abort results. When the last character of a frame is put in OCW, Bit 3 should be set so that the CRC and closing Flag sequence will be transmitted.

## A.5 To Receive on a Synchronous Port

1) Turn on Data Terminal Ready and - for Half Duplex - turn off Request to Send, by storing the appropriate Port Control Word in UCW. Note: This may be omitted when receiving on an active Full Duplex channel. The above assumes that the Data Set is set up for automatic answering, otherwise Data Terminal Ready should not be turned on until the Ring Indicator interrupt occurs.

2) Set up IBP, LIB, and ICW for the desired automatic input buffer.

The Mux is now ready to receive data and to store it away in the specified input buffer. Automatic input continues until one of the following occurs, at which time an interrupt is given and IDN is set to 1.

| Condition | Indication |
|---|---|
| Buffer full | IBP = LIB |
| Data Channel Late | ICW Bit 4 = 1 |
| DSR went low | ICW Bit 5 = 1 and Bit 8 = 1 |
| Carrier Lost | ICW Bit 5 = 1 and Bit 10 = 1 |
| SCAR changes state | ICW Bit 5 = 1 and Bit 11 = 1 |
| *A complete frame is received with good CRC | ICW Bit 5 = 1 and Bits 12-15 = 1000 |
| A complete frame is received with bad CRC | ICW Bit 5 = 1 and Bits 12-15 = 1001 |
| An Abort is received | ICW Bit 5 = 1 and Bits 12-15 = 0010 |
| Special Character in SIR (Spl Intpt Rqst) mode | ICW Bits 4,5 = 0,0 and character in Bits 8-15 |

*Indicates the proper termination in normal reception.

References.


ADDCP = Advanced Data Communication Control Procedures, Proposed ANSI Standard (X3S34/589), 5th Draft, 9 April 1976

HDLC = High Level Data Link Procedures, Proposed International Standard (ISO DIS 3309)

SDLC = Synchronous Data Link Control, IBM Document GA 27-3093-0, 18 November 1974

# APPENDIX B

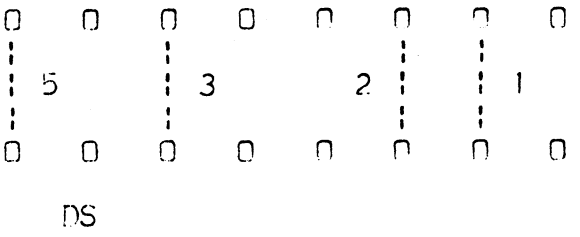
## Options available on the EDSI 310 MIGHTY-MUX board

B.1 Device Code. (These options preserve device code 10/11 for TTY mode - see Section B.10 below for TTY mode device code 50/51 option)

Standard: 25

Option: 4, 5, 14, 15, 20, 21, 24, 30, 31, 34, 35.

Implementation: At location 3B of the 310 board, there are four jumpers (labelled DS) which control bits 1, 2, 3 and 5 of the six-bit Device Code. Bits 0 (the msb) and 4 are permanently wired to zero. The jumper block at location 3B is as shown below:-




The numbers represent the Device Code bits; a jumper on the left makes that bit a 1, a jumper on the right makes it a 0. Thus, the Standard Device Code has bit 5 = 1, bit 3 = 1, bit 2 = 0, bit 1 = 1.

| 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 | 1 |

Together with bits 0 and 4 both = 0, this makes the Device Code = 010,101 = 25 Octal. To change Device Code to any of the available options, decode which bits need to be changed, cut the appropriate jumpers and install their opposites.

NOTE: When changing the Device Code to which the 310 board responds, it is also necessary to change the code with which the 310 responds to an INTA instruction. That code is controlled by jumpers (labelled I) at location 10F on the 310 board. Therefore, for proper response of the 310 board to the new Device Code, the jumpers at location 10F should be changed in identical fashion to those changed at location 3B.

Effect on Diagnostic: Diagnostic must be re-assembled to allow for the new Device Code.

B.2 Mask Bit (to disable interrupts).

Standard: Bit 5 (counting the msb as bit 0).

Option: Any bit, 0 through 15.

Implementation: Mask Bit jumper (M) is located next to pin 12 of I.C. 9C and as standard is connected so that bit 5 is the Mask Bit. To change Mask Bit to some other bit, jumper M should be cut, and feed-thru closest to pin 12 of I.C. 9C jumpered to the appropriate I.C. pin to give the required Mask Bit as specified by the table below:-

| Reqd. Mask Bit | Jumper To |
|---|---|
| 0 | I.C. 14E pin 13 |
| 1 | I.C. 14E pin 6 |
| 2 | I.C. 14E pin 10 |
| 3 | I.C. 14E pin 3 |
| 4 | I.C. 12E pin 13 |
| 6 | I.C. 12E pin 10 |
| 7 | I.C. 12E pin 3 |
| 8 | I.C. 13E pin 13 |
| 9 | I.C. 13E pin 6 |
| 10 | I.C. 13E pin 10 |
| 11 | I.C. 13E pin 3 |
| 12 | I.C. 11E pin 13 |
| 13 | I.C. 11E pin 6 |
| 14 | I.C. 11E pin 10 |
| 15 | I.C. 11E pin 3 |

Effect on Diagnostic: Requires patch for new Mask Bit. For example, for Mask Bit = 3, change the word at location EIMUX-2 (location 216 in MUXDP0428) from 175777 to 167777, i.e. instead of bit 5 = 0, bit 3 is now = 0. The corresponding checksum at CKS1 (260 in MUXDP0428) must then also be changed.

B.3 Master TTY Mode (using device code 10 & 11).

Description: IORST initializes Mux into TTY mode (unless disabled). The port which is Port 0 under Mux control is the TTY port. The initial port parameters are set up as follows:-

Baud rate = 110
Character length = 8 bits
Parity = none
Number of stop bits = 2

Standard Device Code 10/11 software interface applies, and there is no Mux action on ports 1 through 7.

Input. When a character is assembled, DONE is set and an interrupt produced if not masked out (Mask bit = bit 14). If BUSY was on it is turned off. DIA -,TTI reads in the character (may be repeated). S or C pulse clears DONE; S sets BUSY and C clears BUSY. Paper tape reader control (RS-232) is driven from BUSY.

Output. DOA -,TTO transfers the outgoing character to the Mux. S pulse starts the actual transmission of the character, also sets BUSY and clears DONE (if on). When transmission is completed, BUSY is cleared and DONE set producing an interrupt if not masked out (Mask bit = bit 15 = lsb). C pulse clears both BUSY and DONE.

TTY ==> MUX. DOC -,MUX turns off TTY mode and turns on MUX mode, starting standard data channel Mux action. Mux mode immediately clears TTI and TTO BUSY and DONE flags but allows any outgoing character to be completed.

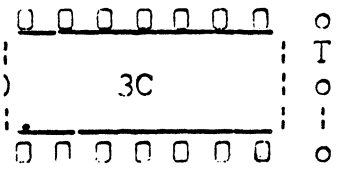
MUX ==> TTY. Giving a C pulse to the Mux (e.g. NIOC MUX) turns off Mux mode and turns on TTY mode, without changing port control parameters. Therefore the software can go to mux mode, set Port 0 to a higher baud rate, then revert to TTY mode and thus use Master TTY I/O interface logic at any desired baud rate.

Real-Time Clock. The Real-Time Clock on the Mux may be used while in TTY mode, by giving an NIOP MUX to turn it on. The Mux BUSY and DONE flags, interrupt logic, and Mux Status Word will then all work in normal manner but for RTC only - no data channel action will occur and ports 1 through 7 will remain inactive.

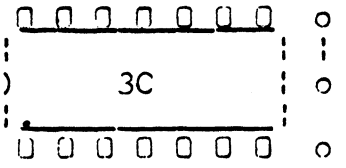
Standard: Disabled.

Option: Enabled following an IORST or NIOC, disabled when MUX is turned on by a DOC -,MUX.

Implementation: Master TTY Mode is enabled by the removal and re-installation of a jumper wire close to pins 7 and 8 of I.C. 3C as shown below:-




Standard (Master TTY disabled) is with jumper wire as shown above. To enable Master TTY, remove jumper and re-install over the T as shown below.




Effect on Diagnostic: See Appendix D of EDSI MIGHTY-MUX DIAGNOSTICS manual for complete description of how to test the MIGHTY-MUX with TTY option enabled.

B.4 Port Control Block Size.

Standard: Port Control Block size = 40 words (octal).

Option: Port Control Block size may be 20 or 10 (octal) words .

Implementation: Port Control Block Size on model 310 Mux is controlled by jumpers at location 13B. To change Port Control Block Size to either 20 or 10 words (octal), cut etches as follows:-

1. At location 13B cut etches between pins 1&18, 2&17, 3&16,...etc... 8&11, and 9&10.

2. Cut etches next to location 13B passing between pins 2&3 and between pins 4&5.

3. Cut etch between feed-thru and pin 3 of chip 14B.

4. Cut etch passing between pins 9&10 of chip 14D and pins 7&8 of chip 13D.

For 20 Word Port Control Block Size

5.0 At location 13B jumper pins 1to17, 2to16, 3to15, ...etc... 8to10, and 9to11

6.0 Jumper 14B pin 4 to 12F pin 2, 14B pin 3 to 12F pin 5 and 14B pin 28 to 11B pin 11.

Effect on Diagnostic: None. Run diagnostic as usual and test for correctness of Port Control Block Size by confirming Block Length typed out (in octal = 20).

For 10 Word Port Control Block Size

5.1 At location 13B jumper pins 1to16, 2to15, 3to14, ...etc... 7to10, 8to18 and 9to17.

6.1 Jumper 14B pin 4 to 12F pin 14, 14B pin 3 to 12F pin 5, 14B pin 2 to 12B pin 14 and 14B pin 28 to 11B pin 11.

Effect on Diagnostic: None. Run diagnostic as usual and test for correctness of Port Control Block Size by confirming Block Length typed out (in octal = 10).

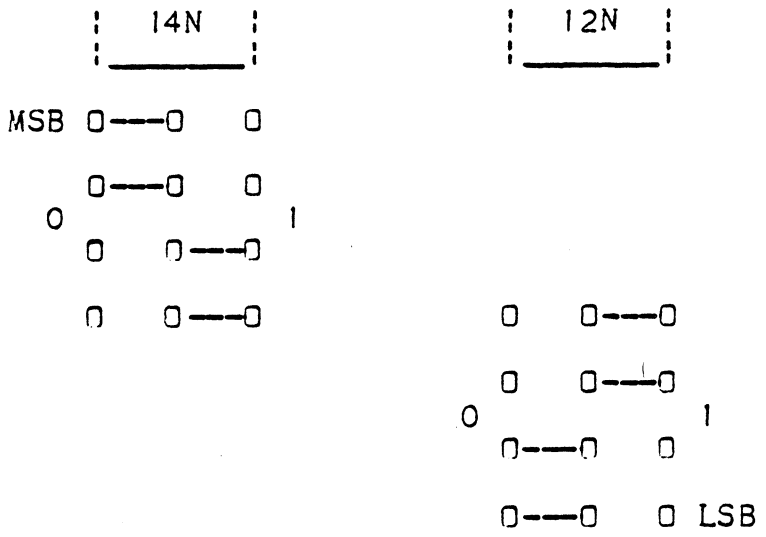
B.5 Core Location of Control Block Area.

All Port Control Blocks are contiguous in core. The beginning of the entire Control Block Area is initialized (by an INRST) to a hardware-controlled default value, after which it may be changed by software (before the MUX is turned on).

Standard: Default value of beginning of control area = 36000.

Option: Default value may be any other multiple of 400.

Implementation: The hardware default value for the Control Block Area base address is specified by the use of jumpers near locations 14N and 12N as shown below:-




A jumper on the left signifies that a particular bit = 0; and a jumper on the right signifies a bit = 1. MSB corresponds to the most significant bit (bit 0) and LSB corresponds to the least significant bit (bit 7). Thus, by utilizing these hardware jumpers, it is possible to specify the eight most significant bits of the Control Block Area base address, i.e. to specify the Control Block Area base address to some multiple of 400 octal. The Control Block Area base address specified by the jumpers shown above is then:-

| MSB | | | | | | | | LSB | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | X | X | X | X | X | X | X | X |
| 0 | . | 3 | | . | 6 | | . | X | | . | X | | . | X | |

Where the "X" values are supplied by the Mux control logic to specify the different addresses of each of the Port Control Blocks within the Control Block Area. That is, Control Block Area base address = 36000, and each individual Port Control Block within the Control Block Area has an address = 36000 + some value XXX.

To change to some non-standard hardware default value, it is necessary to decode the required address (octal to binary), then cut and install the appropriate jumpers:-

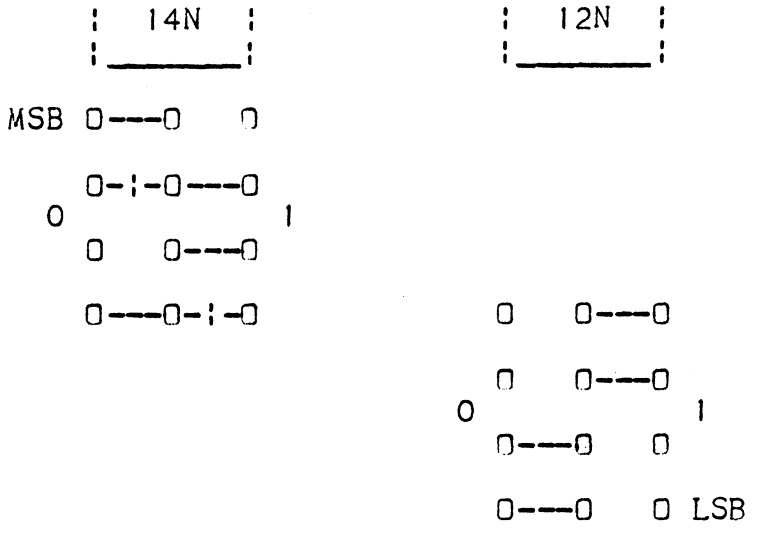
For example, to set up a Control Block Area base address = 66000, the appropriate bits will decode as follows,

```
MSB                         LSB

 0   1  1  0   1  1  0   0

 0 .   6   .   6   .   0
```

i.e. bits 1 and 3 require to be changed by cutting the standard etches for bits 1 and 3, and replacing them with their opposites, as shown below.



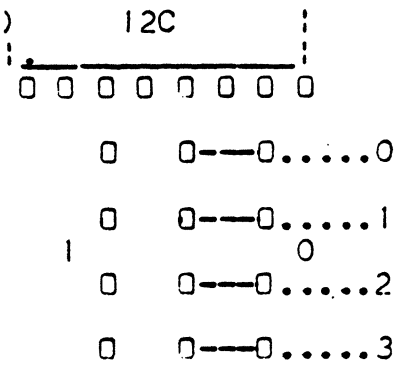

Effect on Diagnostic: None. Run diagnostic as usual, and test for correctness of Control Block Area Base address by confirming Mux Control Area Base typed out. (In above example, Mux Control Area Base type out in octal should be = 66000).

B.6 Location of Port Control Blocks within the Control Block Area (not applicable unless using expansion boards, 301 or 302). Each 8 ports (or fraction thereof) on a 310 or 301 or 302 board constitute a set whose Port Control Blocks must be contiguous in core. As the Mighty-Mux is expandable up to 128 ports, there may be up to 16 sets.

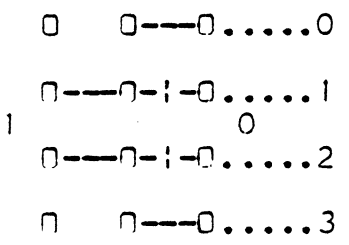
Standard: The 8 ports on the 310 constitute set 0; i.e. their Control Blocks are at the beginning of the Control Block Area.

Option: The 310 ports may be any set from 1 through 15.

Implementation: The 310 Ports Control Block set is specified by jumpers situated next to the I.C. at location 12C, and is set up as shown below:-




Standard jumpers, positioned as shown (all zeroes), specify the 310 Port Control Blocks as set 0. To specify sets 1 through 15, simply cut etches and insert jumpers as required to change appropriate bit(s) from 0 to 1, so that with bit 3 = lsb, and bit 0 = msb, the number thus created is the binary equivalent of the required set number, i.e. for 310 Port Control Blocks = set 6, the zero jumpers of bits 1 and 2 would be cut, and their opposites inserted as shown below:-




To give: 0 1 2 3

0 1 1 0 = 6 octal

The above description of Port Control Block set-up applies equally well to expansion boards, except that these boards are supplied with DIP switches rather than jumpers, to facilitate ease of set-up, since expansion boards have no standard set values.

For example, the switches on a 24-port 301 board, (3 sets of 8 ports) which are located at 5C, 10C, and 15C, might typically require to be set up for ports 8 through 16, 17 through 24, and 25 through 32 (with ports 0 through 7 on the 310 board); i.e. the 301 expansion board requires to be set up for port control block areas corresponding to sets 1, 2, and 3 (310 board is set 0) and this could be implemented as shown below:-

| | 15C | | | 10C | | | 5C | | |
|---|---|---|---|---|---|---|---|---|---|
| MSB | 0 | 1 | 0 on | 0 | 1 | 0 on | 0 | 1 | 0 on |
| | 0 | 2 | 0 on | 0 | 2 | 0 on | 0 | 2 | 0 on |
| | 0 | 3 | 0 off | 0 | 3 | 0 off | 0 | 3 | 0 on |
| LSB | 0 | 4 | 0 off | 0 | 4 | 0 on | 0 | 4 | 0 off |
| | 0011 = set3 | | | 0010 = set2 | | | 0001 = set1 | | |

Note: In this case, "on" denotes a logical 0, and "off" a logical 1, with switch 4 being the least significant, and switch 1 being the most significant bit of the set number.

Effect on Diagnostic: None. Run diagnostic as usual, and test for correctness of Mux Control Block Area Base address typed out. (For above example, assuming Port Control Block size = 40 octal, Mux Control Block Base address will be = 36000 + 40x6x10 = 41000 octal, <u>assuming that diagnostic is run on 310 board alone</u>; otherwise, if diagnostic is run on 310 board in conjunction with 301 or 302 boards whose Port Control Block Areas constitute sets 0 thru 5, Mux Control Block Area Base address typed out will still be = 36000 as before.

B.7 Special Character Set. A 256x4 PROM is used to determine which characters will produce interrupts (when requested by the software). The character itself is used as an address into the PROM and one of the outputs is used to tell if that character is a special character or not. The options relate to which output is used and also whether the msb of the address is the msb of the character (full 8-bit character set) or some other control signal in the MUX - resulting in two different 7-bit character sets, such as one for input and the other for output, or one for 310 ports and the other for expansion ports.

For example, when we require different special character sets for input and output, we can use the control signal QO+ as the most significant bit of the PROM address, so that for input the most significant bit will be high, specifying one set of PROM addresses (i.e. one group of 4 special character sets) and for output the most significant bit will be low, specifying another set of PROM addresses, and therefore another 4 character sets.

Note that each PROM address specifies a 4-bit memory location, where each bit may or may not be set, corresponding to whether or not this particular PROM address (character) is to be a special character in each of these 4 special character sets.

In those cases where both input and output require the same special character set, then the most significant bit of the PROM address can be set either high or low all the time and the special character set will be the same for both input and output. If this is the case, then if the most significant bit was set high (or low), then by setting the most significant bit low (or high) instead, we will specify another set of PROM addresses, which we may use to define another special character set assuming again that both input and output require the same special character set. For an example of this condition see Options 2 and 2a below.

Standard (Option #0): All ASCII characters <40 or >173 (testing only 7 bits) are "special" characters, for both input and output.

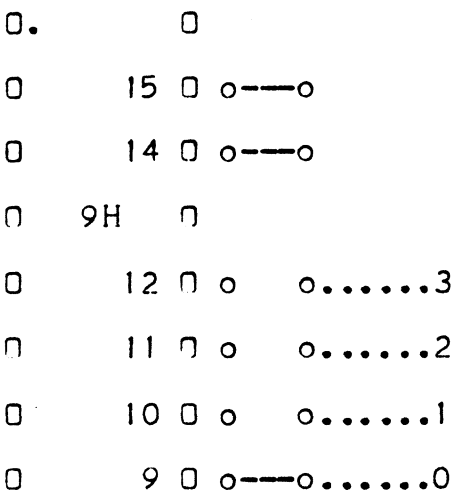
Option #1: 3,4,31. (Both Input and Output)

Option #2: 0,4,15,21,23. (Both Input and Output)

Option #2a: 136. (Both Input and Output)

Option #3: Input: 0,3,4,5,6,10,14,15,25,27,30,31,32,33.
Output: 0,3,4,5,6,14,15,25,27,30,31.

Implementation: Standard is as shown below. The jumper next to pin 15 of I.C. 9H corresponds to the most significant digit of the PROM address being tied to QO+ (normally used to distinguish between input and output special character sets, but in the standard case is there only to accommodate possible different PROMs and character sets in future builds); the jumper next to pin 14 of I.C. 9H is connected to one of the PROM enable inputs, and is always grounded. The jumper is there only to allow possible future expansion to 512x4 PROM. The jumpers next to pins 9 thru 12 of I.C. 9H are used to specify the appropriate option 0 thru 3 as shown.




To change from Standard to some other Special Character Set, cut the jumper next to pin 9 (for Option #0) and replace it with the required Option jumper 1,2 or 3. To distinguish between Option #2 and Option #2a, the jumper next to pin 15 of I.C. 9H must also be cut and replaced by a jumper from the pad next to pin 15 to either ground (for Option #2) or to a pull-up (for Option #2a). A convenient pull-up is located at pin 5 of I.C. 10H ; pin 14 of I.C. 9H is grounded by the jumper next to it, as mentioned earlier.

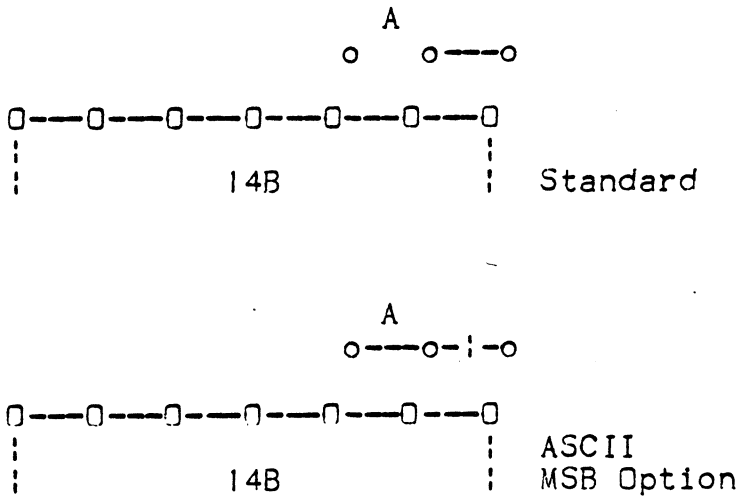
Effect on Diagnostic: Requires overlay or patch - contact EDS for the appropriate modification for a particular Special Character Set and/or ASCII mode option.

B.8 MSB in ASCII Mode (Input only). When special character interrupt is requested by the software, the MUX can automatically set the msb of the incoming character to 1. This will affect all incoming characters, whether special or not, but only on those ports where Special Character Interrupt is selected.

Standard: MSB is set to 1 on all incoming characters if Special Character Interrupt is requested.

Option: MSB is set to 0 if PCON was set up for 7-bit character length, or left unchanged if PCON was set up for 8-bit character length.

Implementation: To enable the option (i.e. to prevent the Mux from setting the most significant bit of the incoming character to a 1), the etch next to pins 8 and 9 of I.C. 14B should be cut, and a jumper installed at "A" as shown below:-




Effect on Diagnostic: Requires overlay or patch - contact EDS for the appropriate modification for a particular Special Character Set and/or ASCII mode option.

B.9 In-Board Power Option. (This is a factory-installed, extra cost option.)

Standard: Mux takes its ±12 volt power requirements from an external power supply, Model EDS-340-32 or 340-64. The power connection goes from the power supply to the 322 connector/cable assembly, and from there to the 310 board via ribbon cable.

Option: Mux generates its ±12 volt requirements from the CPU power supply. Mux contains all necessary voltage regulation, protection, and filtering circuitry.

Implementation: Customer must assure connection and availability of sufficient current from CPU power supply: VINH (pin B84) - 250ma, and -15v. (must be connected to pin B93 on Mux slot) - 250ma.

** Note ** Data General "Eclipse" type CPU uses pin B93 for a signal line, and has -15v on pins A14 and A16 instead. Therefore, to use a 310 Mux with In-Board Power Option in an Eclipse, the 310 board must be modified by cutting the etch from pin B93 and jumpering the -15v supply from A14 and A16.

B.10 TTY Mode Device Code 50/51 (Mux Device Code 45)
(See also Section B.1 for Mux Device Code options preserving TTY Mode Device Code 10/11)

| | MUX | TTY |
|---|---|---|
| Standard: | 25 | 10/11 |
| Option: | 45 | 50/51 |

Implementation: Standard Device Code is produced having Bit 0 (msb) and Bit 4 permanently wired to zero. Mux Device Code = 45 and TTY Device Codes = 50/51 require bit 0 = 1. Therefore, in addition to standard jumper changes (for bit 1 = 0), at locations 3B and 10F, modification to bit 0 is also required as indicated below:-

(a) Cut etch between 2D pin 11 and edge connector pin A72.
Cut etch between 3A pin 7 and 2D pin 10.
Add jumper between 3A pin 7 and edge connector pin A72.

(b) Cut etch between 3B pin7 and 3B pin 10.
Add jumper between 3B pin 8 and 3B pin 9.

(c) Cut etch between 10E pin 8 and 13E pin 1.
Add jumper between 10E pin 8 and 13E pin 12.

(d) Cut etch between 10F pin 7 and 10F pin 10.
Add jumper between 10F pin 7 and 10F pin 12.

| This then gives: | 0 | 1 | 2 | 3 | 4 | 5 | | |
|---|---|---|---|---|---|---|---|---|
| Mux Device Code: | 1 | 0 | 0 | 1 | 0 | 1 | = | 45 octal |
| TTY Device Code: | 1 | 0 | 1 | 0 | 0 | 0 | = | 50 octal TTI |
| | 1 | 0 | 1 | 0 | 0 | 1 | = | 51 octal TTO |

# APPENDIX C

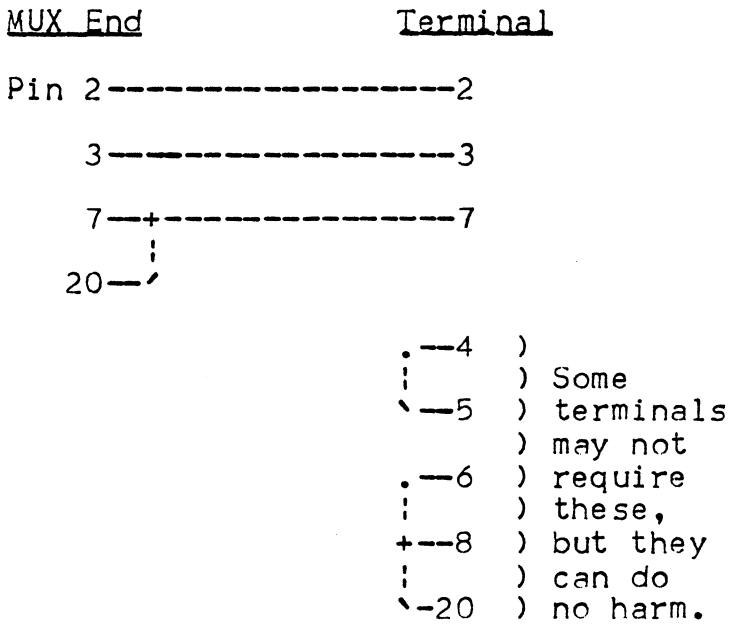
# Interface

## C.1 Junction Panel Connections (EIA)

EIA Standard RS-232-C defines the interface between a Data Set (e.g., modem) and a Data Terminal (e.g., CRT). The Mighty-Mux can communicate with both data sets and data terminals; however, the 25-pin connectors on the junction panel are wired to make the Mux look like a data set. Therefore, data terminals may be plugged into it directly, but data sets must be connected through a cable which interchanges certain signals. All connectors used are standard 25-pin D-type connectors. The connectors on the junction panel are 25-pin Cannon type D female connectors, model DB-25S. Mux cable, therefore requires male end. Terminals and modems typically require male end also. See below for sample Mux cable connections.

### C.1.1 MIGHTY-MUX to Terminal

Device status, pin 20, is grounded (pin 7) at Mux end, to inhibit automatic log-off. Alternatively, if terminal has a stable Data Terminal Ready output on pin 20, it may be connected from pin 20 at the terminal end through to pin 20 at the Mux end.



C.1.2 MIGHTY-MUX to Modem

Full Duplex. A full duplex modem should be employed when using IRIS operating system.

| MUX End | | Modem | |
|---|---|---|---|
| Pin 2 | ——— | 3 | Received Data |
| 3 | ——— | 2 | Transmitted Data |
| 7 | ——— | 7 | Signal Ground |
| 8 | ——— | 20 | Data Terminal Ready |
| 20 | ——— | 8 | Received Signal Carrier Detect |

Half Duplex. In half duplex systems, pin 20 of the Data Set must be connected to a +5 to +15 volt source, or jumpered (as shown) to Data Set pin 6 (Data Set Ready).

| MUX End | | Modem | |
|---|---|---|---|
| Pin 2 | ——— | 3 | Received Data |
| 3 | ——— | 2 | Transmitted Data |
| 5 | ——— | 4 | Request to Send |
| 7 | ——— | 7 | Signal Ground |
| | | 6 | Data Set Ready |
| | | 20 | Data Terminal Ready |
| 20 | ——— | 5 | Clear to Send |

## C.2 Junction Panel Connections for Current-Loop

The EDS-322-CL Cable Assembly allows the user the versatility of plugging cables requiring either EIA-level or current-loop interfacing into the same 25-pin connector. This is achieved by using pins 11, 18, and 25 (left "unassigned" by EIA spec RS-232-C) for the current-loop (C-L) connections (see Figure C-1). Pin 25 carries the C-L output from the Mux, and Pin 18 receives the C-L input to the Mux, with Pin 7 (Signal Ground) serving as the return for both. Pin 11 must be jumpered to Pin 2 whenever C-L is used; this is best done inside the 25-pin connector on the C-L cable.

The C-L circuitry on the 322-CL is driven by a +28 volt source, and is designed for 20 ma current in both input and output. This may be increased to 40 or 60 ma by adding two 1.4K (1 watt) or 680 ohm (2 watt) resistors on the 322 board next to each 25-pin connector for which the change is desired.

*** CAUTION ***

The Mighty-Mux Current Loop Interface (322-CL) is designed for use with terminals that are "passive" (no voltage source) in both input and output. Do not plug an "active" terminal (internal C-L voltage source) into the 322-CL. See schematics in Figure C-3.

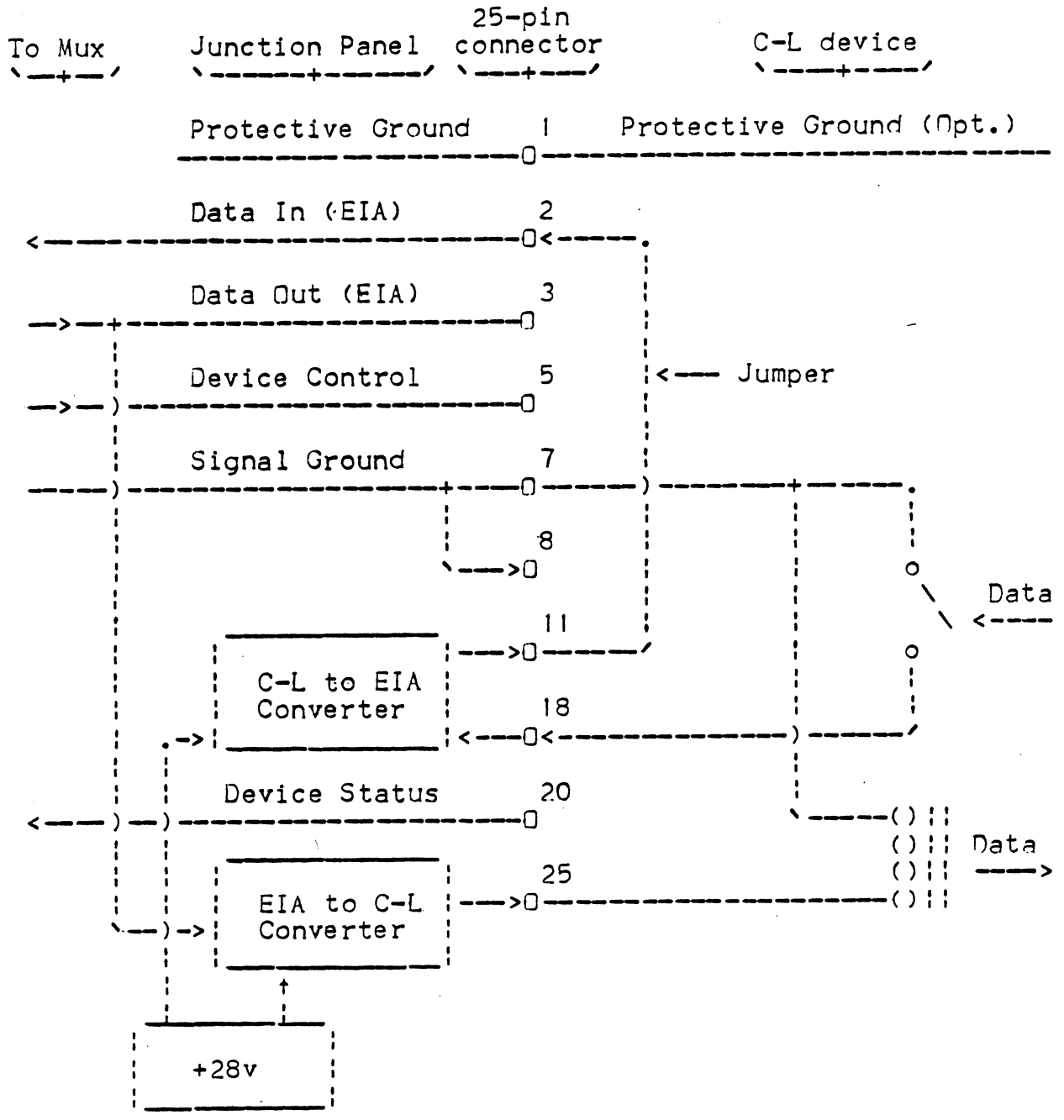



Figure C-1. Junction Panel Connections for Current-Loop Interfaces

## C.3 Junction Panel Connections for Synchronous Ports

The following connections are made on the 25-pin connector on the junction panel (EDS-323 boards), in accordance with EIA Spec RS-232-C.

| Pin | Circuit | Function | Direction |
|---|---|---|---|
| 1 | AA | Protective Ground | -- |
| 2 | BA | Transmitted Data | To Data Set |
| 3 | BB | Received Data | From Data Set |
| 4 | CA | Request To Send | To Data Set |
| 5 | CB | Clear To Send | From Data Set |
| 6 | CC | Data Set Ready | From Data Set |
| 7 | AB | Signal Ground | -- |
| 8 | CF | Received Line Signal Detector | From Data Set |
| 12 | SCF | Secondary Received Line Signal Detector | From Data Set |
| 15 | DB | Transmit Clock | From Data Set |
| 17 | DD | Receive Clock | From Data Set |
| 19 | SCA | Secondary Request To Send | To Data Set |
| 20 | CD | Data Terminal Ready | To Data Set |
| 22 | CE | Ring Indicator | From Data Set |

## C.4 Input and Output Timing and Voltage Specifications

The Mighty-Mux I/O timing specs are determined primarily by the Universal Asynchronous Receiver/Transmitter (UART) chips used in each Port and the electrical specs are determined by the TTL/EIA converter chips. The interrelationship between these circuits is shown in Figure C-2.

The basic timing source is a 6.7584 MHz ±.005% crystal oscillator. This is followed by a frequency divider producing 16 times the selected Baud rate, which in turn drives the UART chip. The UART receiver is so designed as to allow up to almost 47% time distortion, while the transmitter produces less than 1% distortion. This makes the Mighty-Mux compatible with virtually all asynchronous modems and terminals in use today.

The voltage levels of the Mighty-Mux standard I/O circuitry are in accordance with EIA Specification RS-232-C. The 1488 chips are driven from a ±12 volt source, and thus the outgoing levels are:

Data = 0 or Device Control = 1: +10 volts
Data = 1 or Device Control = 0: -10 volts

For incoming lines the 1489 chips give the following response:

> +1.3 volts: Data = 0 or Device Status = 1
< +0.7 volts or open: Data = 1 or Device Status = 0
Between +.7v and +1.3v: Indeterminate

The 1489 offers an input resistance of about 3.5K; the 1488 can drive any load above 2K.

The optional Mighty-Mux Current-Loop interface is driven from a +28 volt source through a current-limiting resistor of 1.4K for the 20 ma version (or 700 ohm for 40 ma or 470 ohm for 60 ma). These values apply to both the input and output circuits, as shown in Figure C-3.

CAUTION: See "CAUTION" on page C-3.

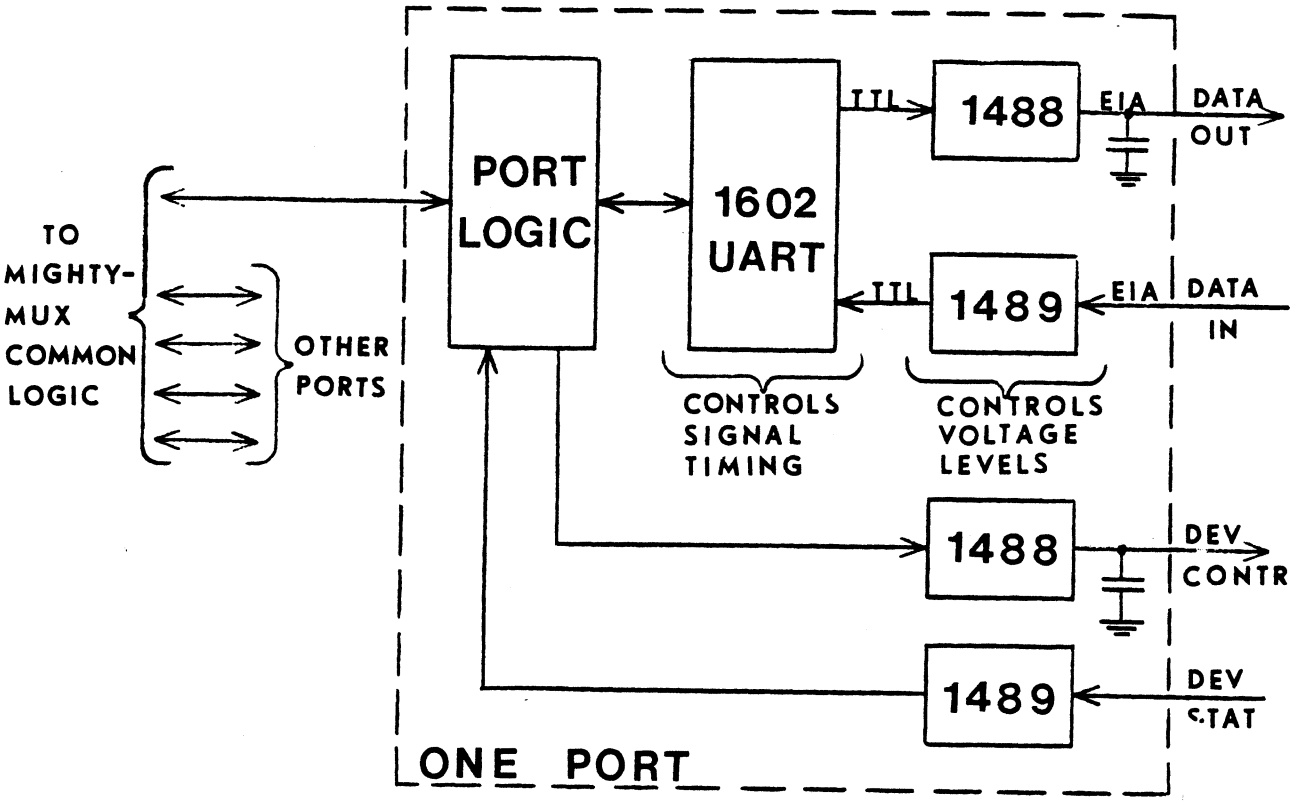



Figure C-2. Block Diagram of EIA Interface

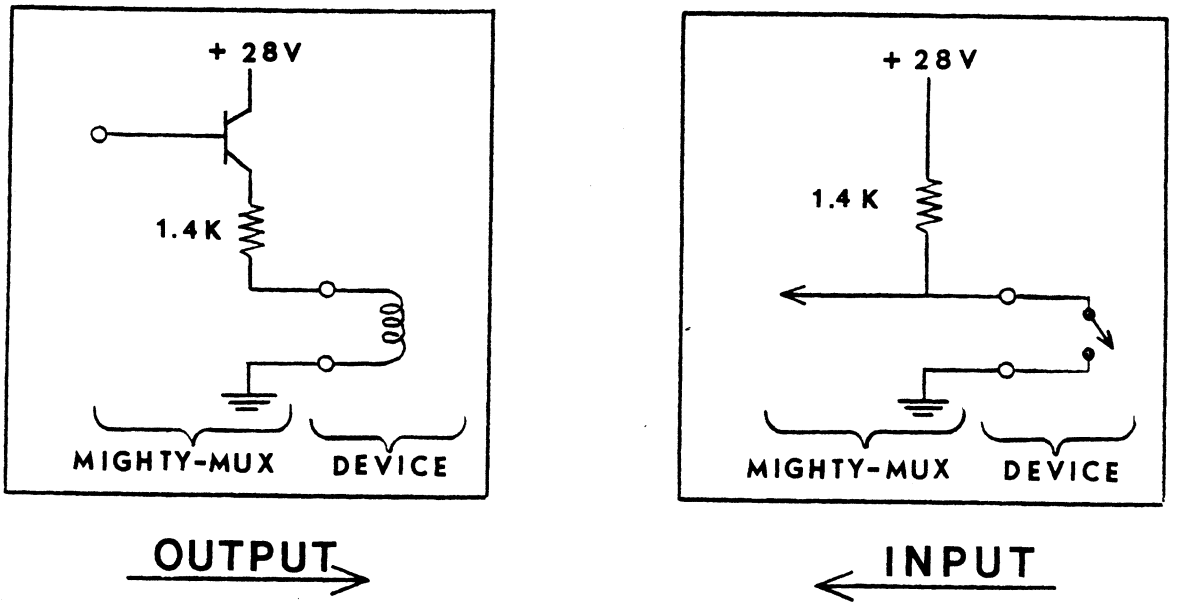



Figure C-3. Current Loop Circuitry

## C.5 Overall Mux Timing

When no input or output is taking place, the Mux inspects each OCW about once every 50 msec. Therefore, when the software starts an output by setting OCW to an active output mode, there may be a delay of up to 50 msec before the Mux begins transmission. (This delay can be avoided by the use of a DOA-, MUX instruction which may be used to "prod" the Mux into immediate action without waiting for the expiraton of the 50ms "Outime" delay). Once transmission has begun on a port, the Mux will reinspect that OCW for the next output command as soon as the output buffer register of that port is empty and the port counter has scanned around to the appropriate port (typically 100 micro-seconds maximum).

If the Device Status changes while there is no output on that port, there is again a delay of up to 50 msec before the Status Changed bit is posted and an interrupt is produced. When output is in process, however, a Device Status change produces the interrupt as soon as transmission of the current character is completed - i.e., at the time the Mux re-examines OCW to obtain the next character to be output.

When an incoming character is received, the Mux will store it away and produce an interrupt, if appropriate, within at most 1 msec after the center of the first Stop Bit. If multiple interrupts occur from various ports, too quickly for the CPU interrupt service routine to handle, then the Mux Status Word for each port (including port identity) is stored by the Mux in a FIFO stack (up to 40 maximum) until the CPU interrupt service routine is able to service each of them in turn; the Mux presenting the interrupts to the CPU in the order in which they were detected.

The relationship between number of ports and maximum data rate per port is shown in Figure C-4. This figure assumes that all ports operate simultaneously at the same data rate. If some of the ports have a lower data rate, a correspondingly larger number of ports may operate simultaneously. Thus, for example, 80 ports at 9600 Baud impose about the same load as 60 ports at 9600 Baud plus 40 ports at 4800 Baud. (This is not an exact relationship, however, and it is wise to allow a reasonable safety margin when making such a conversion.)

When the Mux operates at the maximum total data rate according to Figure C-4, it produces about 80% overhead, since it always takes 4 or 5 successive data channel cycles (see Figure 3 - Flow chart) and then gives up the data channel for at least one cycle. If the total data rate is less than the maximum, the overhead ratio goes down in direct proportion. Thus, for example, 10 ports at 9600 plus 80 ports at 1200 Baud give about 25% overhead on a Nova 1200 (because that is equivalent to 20 ports at 9600 Baud which is about one third the maximum rate of 32 KBaud for 20 ports). Similarly, 8 ports operating simultaneously at 1200 Baud would produce about 2% overhead on a D-116 (because for 8 ports the maximum rate is about 50 KBaud and 1200 Baud is one fortieth of that).

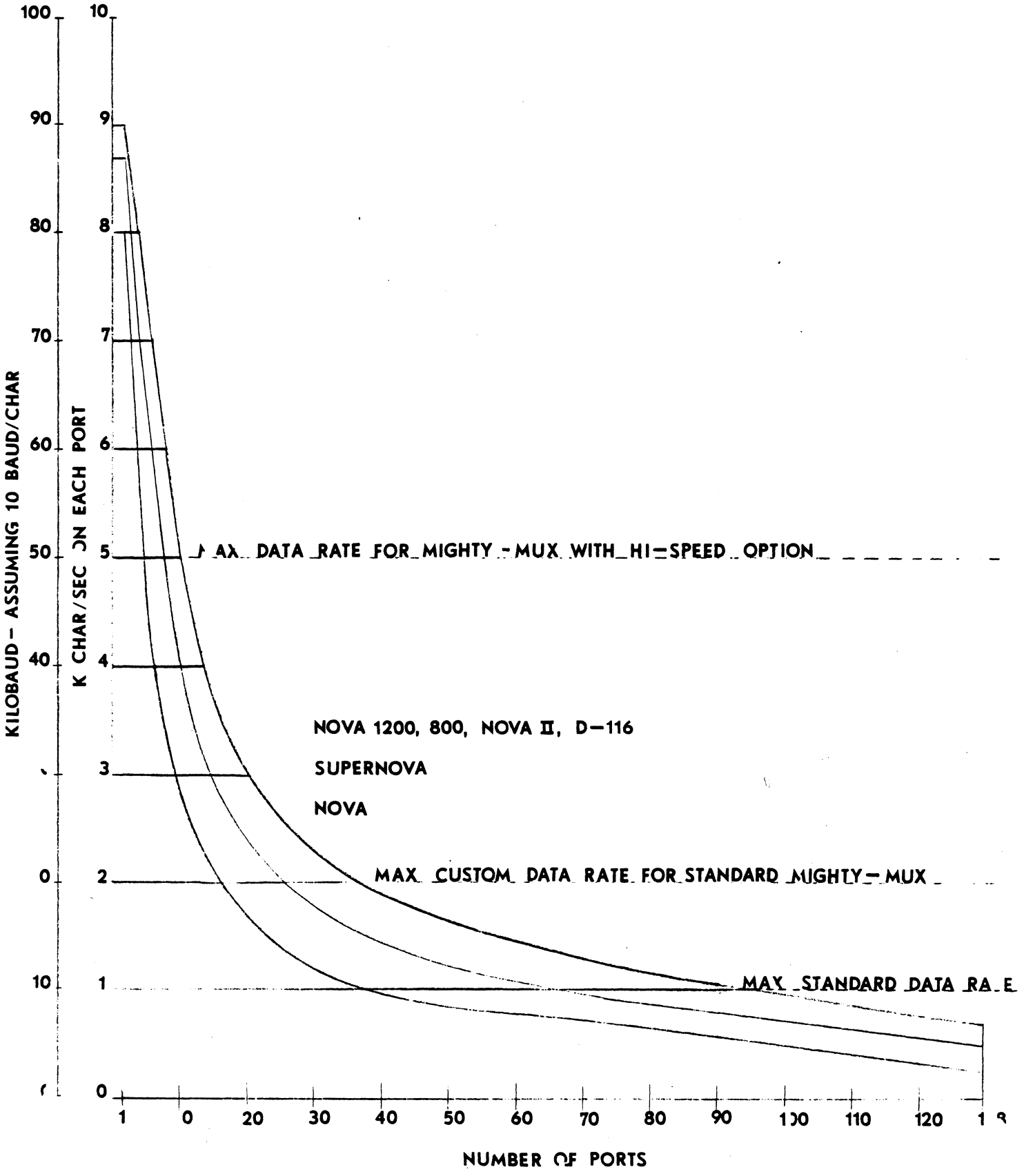



Figure A-5. Relationship Between Number of Ports and Maximum Data Rate per Port

# APPENDIX D

## Installation and Trouble Shooting

1. Installation

a) If any 301 expansion boards are to be used, set the Location of Port Control Blocks within the Control Block Area using the DIP switches on the 301 expansion boards to the desired Control Block locations as explained in Appendix B, Section B.6.

b) With computer power off, insert the 310 board (and 301's, if any) into any slot(s) in the computer chassis.

c) If there are any blank slots in the computer below the 310 board, the Interrupt Priority and Data Channel Priority signals must be jumpered up to the 310 board on the computer's back plane.

d) If any 301 boards are used, they must be connected (on the computer's back plane) to the 310 by connecting the following 41 points on the slot containing the 310 to the corresponding points on any slots containing 301's. This may be done by installing jumper wires (solder or wire-wrap) or by using the EDS-324 Back Plane Connector Cable.

| | |
|---|---|
| A47 | B6 |
| A49 | B13 |
| A57 | B15 |
| A59 | B19 |
| A61 | B23 |
| A63 | B25 |
| A65 | B31 |
| A67 | B34 |
| A69 | B36 |
| A71 | B38 |
| A73 | B40 |
| A75 | B48 |
| A76 | B49 |
| A77 | B51 |
| A78 | B52 |
| A79 | B53 |
| A81 | B54 |
| A83 | B67 |
| A85 | B69 |
| A87 | |
| A89 | |
| A91 | |

e) Mount the Junction Panel(s) (ribbon cable on top) and Power Supply in a convenient place. Connect the 5-conductor cable(s) from the power supply chassis to the Molex connector on each printed circuit board on the junction panel(s). If in-board power option is used, connect -15v to backplane pin B93 (or if Eclipse, modify 310 as per special instructions, see Appendix B, Section B.9).

Attach the free end(s) of the 50 conductor ribbon cable(s) to the connector on the 310 board (and those on the 301's, if any). The ribbon cable may be connected with the cable extending either downward or upward. The only effect of reversing the cable is to reverse the numbering of the eight connectors on the junction panel. Normal left-to-right numbering corresponds to the cable extending upward; this is done so that the cable may be draped over the computer and then to the rear, so that the computer may still be slid in and out of a rack.

f) Plug in the power supply line cord(s), preferably into the rear of the computer so that when the computer is off, no voltage comes to the Mighty-Mux through the ribbon cables.

The Mighty-Mux is now ready for use.

It is recommended that the Mighty-Mux Diagnostic Program furnished with the Mux (MUXDP0428) be run before the Mux is used on a system.

## 2. Trouble Shooting

If there are any indications that the MIGHTY-MUX is not working correctly, it is recommended that the MIGHTY-MUX Diagnostic Program (MUXDP0428) be run. This program tests the Mux in all its operational modes and types out appropriate fault messages if it finds any errors.

If the MUXDP0428 is not available, or if a simple "quick and dirty" test is desired, the following elementary test may be used.

Mini - Test Using Computer Front Panel

| Set Switches | Press | Comment |
|---|---|---|
| Any ICW | Reset, Examine | |
| 0 | Deposit | Clear ICW |
| 40000 | Deposit Next | OCW = Port Control Word |
| 63025 | Deposit Next | DOC 0,MUX; turns Mux on |
| 400 | Deposit Next | JMP .; allows data channel action |
| ICW+2 | Start | |
| ICW | Reset, Examine | Should still be 0 |
| | Examine Next | OCW should be 140000 or 146400 |

If OCW is 146400, it means that the incoming Device Status line has a positive voltage on it and the Mux is working correctly. If ICW is 106777, it is a very strong indication that the -12 volt supply is not getting to the 310 board (especially if the same occurs in all ICW's). This may be checked with a voltmeter or an oscilloscope at pin 2 of any of the 40-pin UART chips (type 1602).

If ICW and OCW are still 0 and 40000, respectively, it is possible that the Mux is not looking for its control blocks in the right place. The following program may then be tried to search for the Mux's apparent control area.

```
0:   20014  LDA     0,14     ;start here
1:   30015  LDA     2,15
2:   41000  STA     0,0,2    ;store 40000 in all words
3:  151404  INC     2,2,SZR
4:       2  JMP     2
5:   63025  DOC     0,MUX    ;start Mux
6:   30015  LDA     2,15
7:   25000  LDA     1,0,2    ;check all words
10:  106414 SEQ     0,1
11:  63077  HALT             ;word not = 40000
12:  151400 INC     2,2
13:      7  JMP     7
14:  40000  40000            ;Mux control word
15:  100016 100016           ;initial address (msb = 1)
```

This program stores 40000 (PCON output or Auto input) in each word above the program up to the end of core, then starts the Mux and tests each word to see if it has changed. If so, it halts, with the changed word in A1 and its address (with msb = 1) in A2.

Press Reset and start at 0.

The program should halt within a second or so, with A2 containing the address of the first OCW, and A1 containing either 140000 or 146400, according as Device Status is 0 or 1. If Continue is then pressed, it should halt at the next OCW, etc., until finally it will halt at the top of core, i.e. with A2 indicating the amount of core in the system.

If the program halts with an ICW address in A2 it indicates that data (or noise) is coming in, or - if the data is all ones - that the - 12v supply is missing. In these cases it may also halt at the corresponding IBP which may have been incremented to 40001.

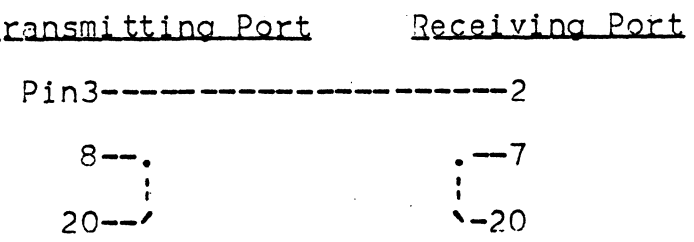
If the program halts with any other value in A2 it may indicate that the address switches on the 301 boards are not set correctly.

If it halts only at the top of core it may indicate that the Mux Control Block Area default value is set to a value greater than the amount of core available, or that the Mux board is not plugged all the way in, or that it is not getting a +5 volt supply from the computer.

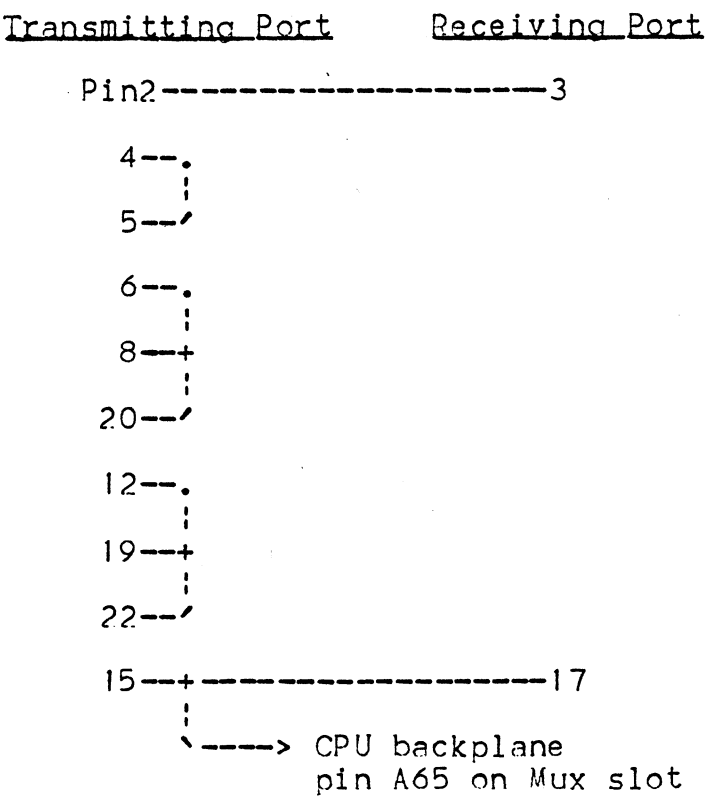
If the program does not halt at all it indicates that the Mux is never releasing the data channel, the most likely cause being that the Data Channel Priority signal is not getting to the 310 board.

D.3 Mux Test Cables (for running diagnostic MUXDP0428)

Asynchronous:




Synchronous:



# GLOSSARY

| | |
|---|---|
| ADCCP | Advanced Data Communication Control Procedures |
| ASC | 7-bit ASCII mode special interrupt request |
| BIN | Byte Indicator |
| C-L | Current Loop |
| CARR | Carrier (Received Line Signal Detector) |
| CPU | Central Processing Unit (Computer) |
| CRC | Cyclic Redundancy Code |
| CTS | Clear To Send |
| DMA | Direct Memory Access (Data Channel) |
| DSC | Device Status Changed |
| DSR | Data Set Ready |
| DTR | Data Terminal Ready |
| DVC | Device Control |
| DVS | Device Status |
| EIA | Electronic Industries Association |
| EOF | End-Of-Frame |
| FDX | Full Duplex |
| FIFO | First-In, First-Out |
| HDLC | High-Level Data Link Control |
| HDX | Half Duplex |
| IBP | Input Byte Pointer (Word 4 of PCB) |
| ICW | Input Control Word (Word 0 of PCB) |
| IDN | Input Done (Bit 0 of ICW) |
| INCHAR | Incoming Character |
| INMO | Input Mode (Bits 1-2 of ICW) |
| LIB | Last Input Byte Pointer (Word 6 of PCB) |
| LOB | Last Output Byte Pointer (Word 7 of PCB) |
| LSB | Least Significant Bit |
| MSB | Most Significant Bit |
| MSO | Mux service Overload |
| MSW | Mux Status Word |
| MUXSTWD | Mux Status Word |
| Mux | MIGHTY-MUX DMA Multiplexer |
| NDB | Number of Data Bits |
| NEC | Not Echoed |
| OBP | Output Byte Pointer (Word 5 of PCB) |
| OCW | Output Control Word (Word 1 of PCB) |
| ODN | Output Done (Bit 0 of OCW) |
| OUTCHAR | Outgoing Character |
| OUTMO | Output Mode (Bits 1-2 of OCW) |
| PCB | Port Control Block |
| PCON | Port Control |
| PDS | Previous Device Status |
| PIN | Parity Inhibit |
| PMO | Parity Mode |
| RECFLGS | Receiver Flags |
| RTS | Request To Send |
| SBS | Stop Bits Select |
| SCAR | Secondary Carrier |
| SDLC | Synchronous Data Link Control |
| SIR | Special Interrupt Request |
| SRTS | Secondary Request To Send |
| UART | Universal Asynchronous Receiver-Transmitter |