ATARI HOME COMPUTER SYSTEM

SERIAL INPUT/OUTPUT INTERFACE

USER'S HANDBOOK

PART I

TABLE OF CONTENTS

APPENDICES

SIO INTERFACE USER'S HANDBOOK

1.0 INTRODUCTION

1.1 This document is intended primarily as a condensed explanation of the Atari Home Computer family's serial I/O interface (SIO). It is written for the reader wanting to understand the interface from the peripheral's end. For instance, it should help the non-Atari OEM quoting costs for engineering and production of a particular standard SIO peripheral such as a printer or diskette.

1.2 Scope

I/O activity between an Atari Home Computer application and the peripheral may be analyzed by viewing the passage of information through a series of layers, with the application at the central, innermost layer, and the peripheral at the outer layer. Each layer may be thought of as consisting of the set of knowledge and data required to characterize that layer, in order to insure that layer is implemented correctly and works.

Diagram 1.1 shows these layers.

```
innermost layer ______________________________
                                             |
                                        APPLICATION
                                             |
                                            CIO
                                             |
                                          HANDLER
                                             |
                                            SIO
                                             |
                                       INTERFACE BUS
                                             |
                                        PERIPHERAL
 outermost layer_____________________________|
```

DIAGRAM 1.1

The scope of this document is confined to detailing the second-to outermost layer, the interface bus. For purposes of this document, the SIO and interface bus layers are considered to be a common pipeline between the handler and the peripheral; all device specificity is contained in the handler and peripheral levels, while SIO and the interface bus are transparent to device specific characteristics. This is true so long as we are considering peripheral devices making use of the interface according to the standards presented here, that is, no attempt is made to describe or list non-standard uses of the serial interface or any of its signals.

An exception to our device non-specificity rule is in the area of predictable device-specific messages over the serial bus. These messages are the ones originating from Operating System resident handlers that use the SIO layer. The O/S resident handlers are characterized in the first two appendices of the document in order to foster future designs around standardized requirements in the common peripherals, the printer and diskette.

Diagram 1.2 shows a more detailed view of the interface bus layer and its surrounding layers.

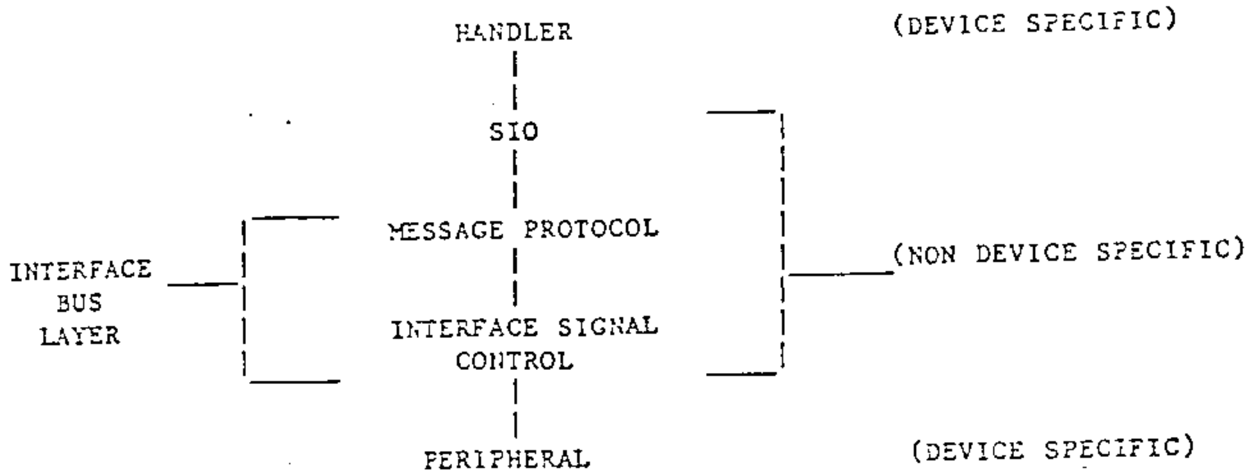


DIAGRAM 1.2

1.3 Goals

The goals of this document are to present in a clear yet terse format the interface bus layer of serial I/O activity. The primary goal is to present all the information necessary to guide design of a new but standardized peripheral (standardized to the bus, not necessarily to all applications).

1.4 Non-Goals

1.4.1 This document does not explain use of the interface bus driving mechanism, SIO, represented by the SIO layer in Diagrams 1.1 and 1.2. SIO's use may become a requirement for the peripheral designer once he has gotten to the point of writing test software for the Home Computer System's exercising of his peripheral across the interface bus. For information on the SIO layer the reader is referred to the Operating System's User's Manual.

1.4.2 This document does not explain the installation of new handlers into the system. Use of the handler layer, and its interfaces to adjacent layers, is explained in the Operating System User's Manual.

1.4.3 Another device-specific function not treated in this document is uploading of a peripheral's handler into the CPU memory directly from the peripheral itself through the serial bus. Provisions exist for uploading and memory management (relocation) of handlers directly from the peripheral's internal ROM storage, obviating the need for additional media devices in the system. A standard protocol for doing this at system initialization time is not described in this document, but will be covered in a follow-up, Part II of this document.

1.4.4 This document is not a Request for Price Quote. It is not a specification for desired features in any peripheral device.

2.0 OVERVIEW

The SIO Interface User's Handbook is organized into three main sections. The first two present the two parts of the interface layer as they are shown in diagram 1.2. The third part describes the intrusion of device specificity into the peripheral designer's task.

In the first section the mechanics, signal descriptions, and electrical specifications of the SIO bus are given. This information will be of primary interest to the interface hardware designer. In the second section, bus protocol is explained. The structures of commands, handshakes/ acknowledgements, data, and transaction complete messages are described along with flowcharts and timing diagrams illustrating their sequence and timing tolerances. This information will be of primary interest to the peripheral firmware designer.

In the third section, device specific information is defined and references are given for finding data about it.

3.0 THE ATARI SERIAL I/O INTERFACE BUS ELECTRICAL AND MECHANICAL

ATARI SERIAL I/O INTERFACE

3.1 Principles of Operation

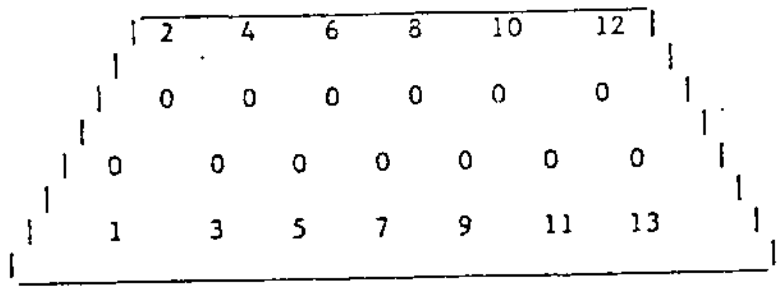
The ATARI personal computers communicate with peripheral devices over a 19,200 Baud asynchronous serial bus. The serial port consists of a serial DATA OUT (transmission) line, a serial DATA IN (receiver) line and other miscellaneous control lines described in the serial I/O Bus Protocol section. Data is transmitted and received as 8 bits of serial data (LSB sent first) preceeded by a logic false start bit and succeeded by a logic true stop bit. DATA OUT is transmitted as positive logic (+4v = TRUE, 0v = FALSE).

This section details the mechanics of serial port communications. For information about peripheral device operations on a system level, consult the Atari Personal Computer System Operating System User's Manual. Part II of this document will describe O/S supported ways in which non-resident, new device handlers can be added or up-loaded into the system.

For information about serial communications that don't conform to the 19,200 baud serial protocol (i.e. the cassette interface) consult the Atari Personal Computer System Hardware Manual.

3.2 SIO Connectors

The Atari A400 and A800

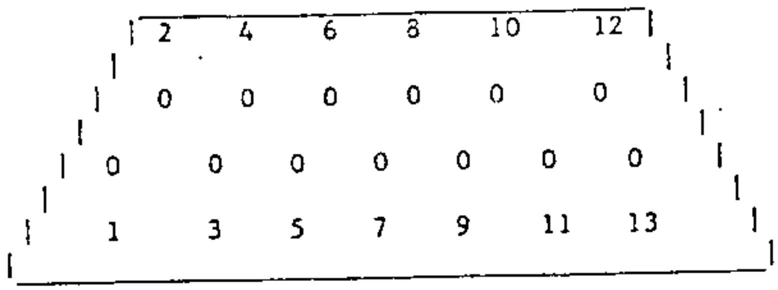


Connector as seen from side of computer (or peripheral).
Cable is mirror image.

1. Clock In
2. Clock Out
3. Data In to Computer
4. GND
5. Data Out of Computer
6. GND
7. $\overline{\text{Command}}$
8. Motor Control
9. $\overline{\text{Proceed}}$
10. +5 / Ready
11. Audio In
12. +12
13. $\overline{\text{Interrupt}}$

3.2 SIO Connectors

The Atari A400 and A800

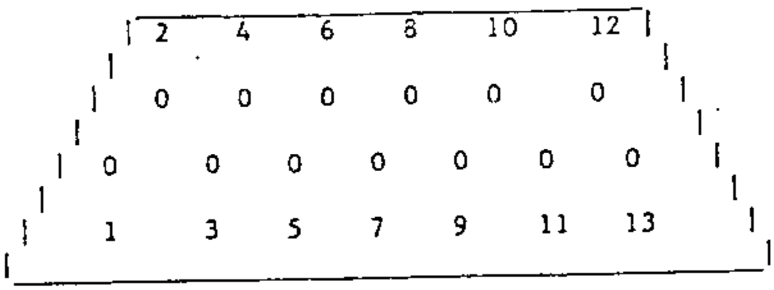


Connector as seen from side of computer (or peripheral).
Cable is mirror image.

1. Clock In
2. Clock Out
3. Data In to Computer
4. GND
5. Data Out of Computer
6. GND
7. Command
8. Motor Control
9. Proceed
10. +5 / Ready
11. Audio In
12. +12
13. Interrupt

3.2 SIO Connectors

The Atari A400 and A800



Connector as seen from side of computer (or peripheral).
Cable is mirror image.

1. Clock In
2. Clock Out
3. Data In to Computer
4. GND
5. Data Out of Computer
6. GND
7. Command
8. Motor Control
9. Proceed
10. +5 / Ready
11. Audio In
12. +12
13. Interrupt

3.3 Brief Description of Serial Port Signals - A800/A400

1. CLOCK IN - Not required in present serial port protocol. Can be used in future synchronous communication schemes.

2. CLOCK OUT- Serial port clock. CLOCK OUT always goes high at the start of each DATA OUT data bit, then CLOCK OUT returns to zero in the center of the DATA OUT bit time. This clock is present only during data transmission.

3. DATA IN - 19200 Baud data line to the computer.

4. GND - Ground (Digital Common.)

5. DATA OUT - 19200 baud data line from the computer.

6. GND - Ground (Digital Common.)

7 $\overline{\text{COMMAND}}$ - Goes to zero when a command frame is being sent.

8. MOTOR CONTROL - Motor Control line for the cassette tape recorder.

9. $\overline{\text{PROCEED}}$ - Not implemented in present serial port protocol. This line may generate (see note 2 below) an IRQ interrupt when it goes to zero.

10. +5/READY - Indicates that the computer is turned on and ready. Restricted use as a +5 volt supply.

11. AUDIO IN - Audio signal from the cassette to the R.F. modulator.

12. +12 VOLT - 300 mA regulated.

13. $\overline{\text{INTERRUPT}}$ - Not implemented in present serial port protocol. This line may generate (see note 2 below) an IRQ interrupt when it goes to zero.

NOTE 1

Lines 1, 2, 9, 12, and 13 have not been utilized in standard peripherals. Line 12 is regulated +12 V dc at 100 mA. Note that this voltage rail is present only on the A800/400, but omitted on the Z800.

NOTE 2

PACTL (Port A control, $D302) is related to $\overline{\text{PROCEED}}$ as follows: Bit 0 is interrupt enable, 1= Enable. It is reset at Power-on. Bit 7 is set when a $\overline{\text{PROCEED}}$ interrupt occurs, and is reset when PACTL is read. PBCTL (port B Control, $D303) is related to $\overline{\text{INTERRUPT}}$ in the same manner as PACTL is related to $\overline{\text{PROCEED}}$.

3.5 Brief Description of Serial Port Signals - Z800

1. 28 VAC - A.C. power provided for Z800 family peripherals.
2. +5/READY - Indicates that the computer is turned on and ready. Restricted use as a +5 volt supply.
3. GND - Ground (Digital Common)
4. CLOCK OUT - Serial port clock. CLOCK OUT always goes high at the start of each DATA OUT data bit, then CLOCK OUT returns to zero in the center of the DATA OUT bit time. This clock is present only during data transmission.
5. DATA IN - 19200 Baud data line to the computer.
6. $\overline{\text{PROCEED}}$ - Not implemented in present serial port protocol. This line may generate an IRQ interrupt when it goes to zero. See note, Sec. 3.3
7. $\overline{\text{PWRON}}$ - Output from Z800 to peripherals. It causes any device so designed to respond to power-up and initialize.
8. AUDIO - AUDIO Signal from the casette.
9. 28 VAC RETURN - Return pin complementing pin 1.
10. GND - Digital common.
11. CLOCK IN - Not required in present serial port protocol. Can be used in future synchronous communication schemes.
12. DATA OUT - 19200 Baud line from the computer.
13. $\overline{\text{INTERRUPT}}$ - Not implemented in present serial port protocol. This line may generate an IRQ interrupt when it goes to zero. See note 2, Sec. 3.3
14. $\overline{\text{COMMAND}}$ - Goes to zero when a command frame is being sent.
15. MOTOR CONTROL - Motor Control line for the cassette tape recorder.

3.6 SIO ELECTRICAL SPECIFICATIONS FOR PERIPHERALS

DC CHARACTERISTICS

		MIN.	NOMINAL	MAX.	UNITS
Peripheral Inputs Signal Names: Data Out (Pin 5) Command (Pin 7) Clock Out (Pin 2)					
V_{IH}	Input High Voltage	2.0		V_{CC}	VOLTS
I_{IH}	Input High Current @ 2 V_{IN}			+40	μA
V_{IL}	Input Low Voltage	0		0.8	VOLTS
I_{IL}	Input Low Current @ .8 V_{IN}			-50	μA
Peripheral Outputs Signal Names: Data In (Pin 3) Proceed (Pin 9) Interrupt (Pin 13)					
V_{OH}	Output High Voltage	3.0		V_{CC}	VOLTS
I_{OH}	Output High Current @ 3 V_{OUT}			+30	μA
V_{OL}	Output Low Voltage	0		0.4	VOLTS
I_{OL}	Output Low Current @.4 V_{OUT}	+1.1		+1.2	mA

NOTES: 1. Negative current indicates current flow out of the specified terminal, (sourcing). Positive current indicates current sinking.

2. All specifications of maximum current based on a maximum of 12 peripherals on the bus with the exception of the Motor Control line which allows for use by a maximum of four program recorders.

	MIN.	NOMINAL	MAX.	UNITS
BI-DIRECTIONAL "CLOCK IN" (Pin 1)				
Peripheral Output to "Clock In"				
V_{OH} Output High Voltage	2.6		V_{CC}	VOLTS
I_{OH} Output High Current			+0.8	mA
V_{OL} Output Low Voltage	0		0.4	VOLTS
I_{OL} Output Low Current			+1.0	mA
Peripheral Input from "Clock In"				
V_{IL} Input Low Voltage	0		0.8	VOLTS
I_{IL} Input Low Current @ .8 V_{IN}			-50	A
V_{IH} Input High Voltage	2.0		V_{CC}	VOLTS
I_{IH} Input High Current			+40	A
+5V/Ready Input (Pin 10)				
V_{IH} Input High Voltage	2.0		V_{CC}	VOLTS
I_{IH} Input High Current @5.2 V_{IN}			+1.0	mA
V_{IL} Input Low Voltage	0		0.4	VOLTS

DC CHARACTERISTICS (CONTINUED)

	MIN.	NOMINAL	MAX.	UNITS
Motor Control Input (Pin 8)				
V_{IH} Input "ON" Voltage	4.55		4.95	VOLTS
I_{IH} Input "ON" Current @ 4.95 V_{IN} (See note 2)			+15	mA
V_{IL} Input Voltage Low			0.3	VOLTS
I_{IL} Input Current Low			+0.9	mA
+12 Volts (Pin 12)				
Voltage	11.5	12	12.5	VOLTS
Current available to each peripheral			+40	mA

AC CHARACTERISTICS

	MIN.	NOMINAL	MAX.	UNITS
Digital Signal Lines: Peripheral Input- "Data Out" Peripheral Output- "Data In"				
r_{dt} Data Transfer Rate		19.2		K Baud
t_p Pulse Width		52		S
t_r Rise Time	13	16	19	S
Clock Lines: Clock In Clock Out				
t_p Pulse Width		20		S
t_r Rise Time	13	16	19	S
Peripheral Output Signal Name: Audio In (Pin 11)				
Amplitude - Into 6K ohm Load	0.5		1.0	V_{PP}
Z_{IN} Computer Input Impedence		6K		OHMS

4.0 BUS PROTOCOL

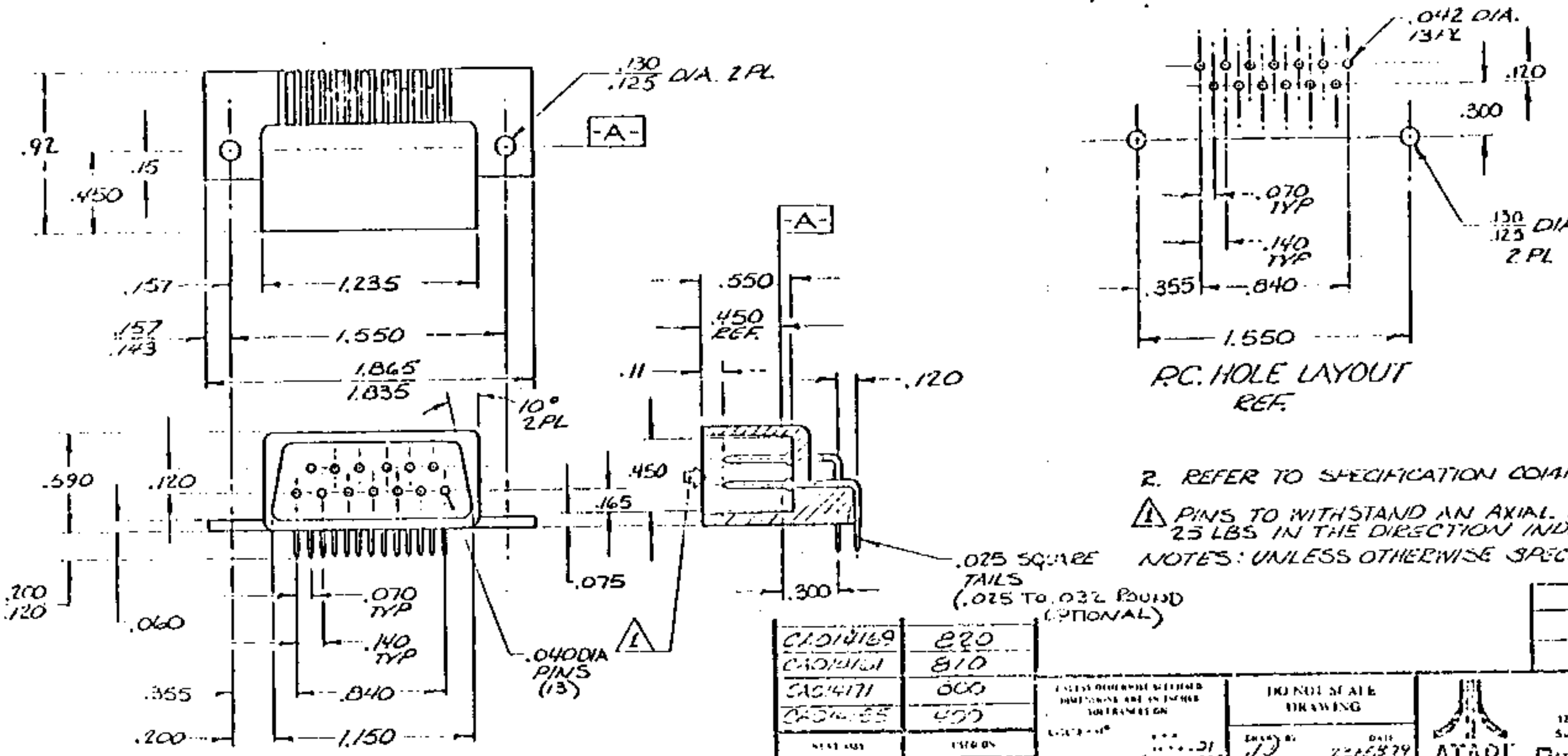
4.1 Bus Operation Overview

4.2 Bus Commands

4.3 Bus Frame

4.4 Bus Timing

LTR	REVISION DESCRIPTION	DATE	APPROVED
4	REDRAWN & REVISED		
A	PRODUCTION RELEASE		
B	REVISED PER ECN # 3.346		
C	REVISED PER ECN # A0138		
D	REV PER ECN # A0651		



NOTES: UNLESS OTHERWISE SPECIFIED

1. (△) PINS TO WITHSTAND AN AXIAL FORCE OF 25 LBS IN THE DIRECTION INDICATED
2. REFER TO SPECIFICATION C014132.

NEXT ASSY	USED ON
CA014169	820
CA014161	810
CA014171	800
CA014155	400
APPLICATION	

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ON: .XX ±.01 .XXX ±.005

MATERIAL: SPACER - NYLON (BLACK)

PINS - BRASS

FINISH: PINS - 150 μIN TIN OVER 30 μIN NICKEL.

DO NOT SCALE DRAWING

DRAWN BY: [illegible] DATE: 22 FEB 79

CHECKED: [illegible]

ENGINEER:

PRODUCT ENGINEER:

MFG. ENGINEER:

OUTSTANDING ECNs

ATARI Inc. 1265 Borregas Avenue Sunnyvale, Calif. 94086

TITLE: CONNECTOR 13 PIN - RIGHT ANGLE

SIZE: C DRAWING NO.: C012995 REV: D

SCALE: [illegible] SHEET 1 OF 1

NOTICE TO ALL PERSONS RECEIVING THIS DRAWING

CONFIDENTIAL: [illegible]

4.0 BUS PROTOCOL

4.1 Bus Operation Overview

4.2 Bus Commands

4.3 Bus Frame

4.4 Bus Timing

4.1 Bus Operations Overview

The bus protocol specifies that all commands must originate from the computer, and that peripherals will present data on the bus only when commanded to. Every bus operation will go to completion before another bus operation is initiated (no overlap). An error detected at any point in the command sequence will abort the entire sequence. The serial bus protocol provides for three types of commands: 1) data send, 2) data receive and 3) immediate (no data -- command only).

A bus operation consists of the following elements:

Command frame from the computer.
Acknowledgement (ACK) from the peripheral.
Optional data frame to or from the computer.
Operation complete (COMPLETE) from the peripheral.

Note that in the data receive type of operation, operation complete (COMPLETE) is sent from the peripheral before the data frame is sent from the peripheral to the computer.

4.2 Command Frame

There is a common element in all three operation types, a command frame consisting of five bytes of information sent from the computer while the $\overline{\text{COMMAND}}$ line is held low. The format of the command frame is shown below:

device I.D
command
auxilliary #1
auxilliary #2
checksum

The device I.D. specifies which of the serial bus devices is being addressed (see Appendix C for a list of device I.D.s).

The command byte contains a device dependent command (see Appendix C for a list of device commands).

The auxilliary bytes contain more device dependent information.

The checksum byte contains the arithmetic sum of the first four bytes (with the carry added back after every addition).

COMMAND FRAME ACKNOWLEDGE

The peripheral being addressed would normally respond to a command frame by sending an ACK byte ($41) to the computer; if there is a checksum problem with the command frame, the peripheral should not respond.

The following page shows a flowchart for the peripheral's command frame processing. All operation types conform to the command frame structure indicated. What happens after the command frame is successfully processed depends on what type operation is being specified by the command. This is indicated by the A/B/C designation at the end of the flowchart. Flow-charts for each of the three types of operations ("A", "B", "C") are given in the following paragraphs describing the data frame structure. All three types of operation are also shown after their flowcharts by means of timing diagrams, which show the entire operation, including the command frame part of the operation. Actually timings for the tx parameters are given in section 4.4 on Bus Timing.

PERIPHERAL'S COMMAND FRAME PROCESSING

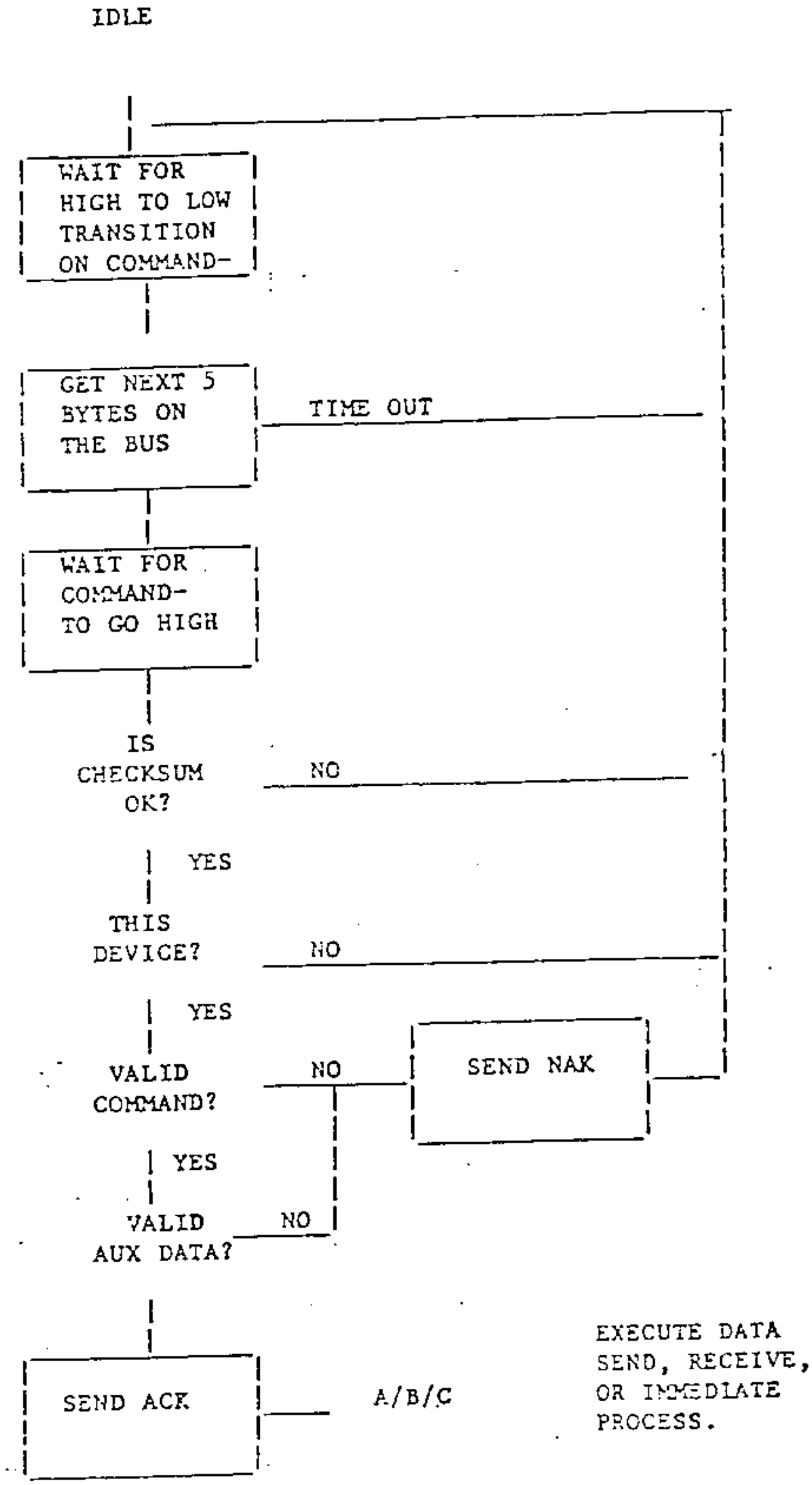
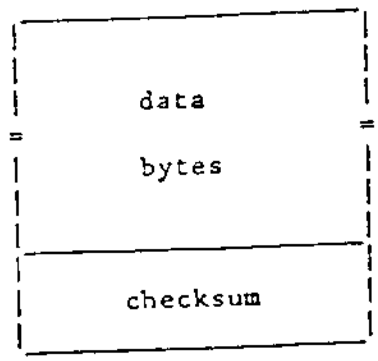


Diagram 4.1

4.3 Data Frame

Following the command frame (and ACK) may be an optional data frame which is formatted as shown below:



This data frame may originate at the computer or at the device controller, depending upon the command.

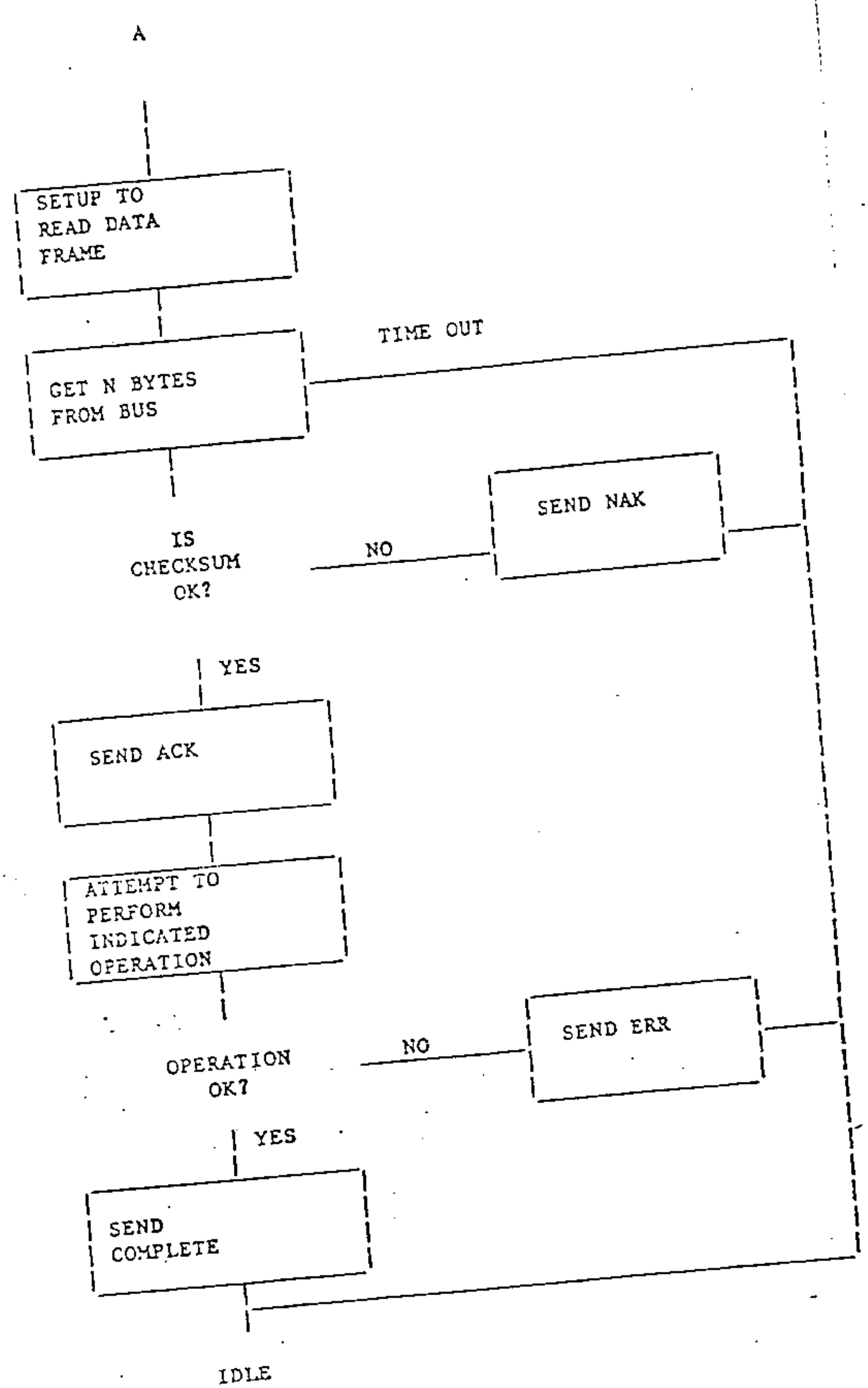
The checksum value in the data frame is the arithmetic sum of all of the frame data preceding the checksum, with the carry from each addition being added back (the same as for the command frame).

In the case of the computer sending a data frame to a peripheral, the peripheral is expected to send an ACK if the data frame is acceptable, and send a NAK ($4E) or do nothing if the data frame is unacceptable.

OPERATION COMPLETE

A peripheral is also expected to send an operation COMPLETE byte ($43) at the time the commanded operation is complete. The location of this byte in the command sequence for each command type is shown in the timing diagrams which follow. If the operation cannot go to normal, error-free completion, the peripheral should respond with an ERROR byte ($45) instead of COMPLETE.

DATA FRAME TO PERIPHERAL

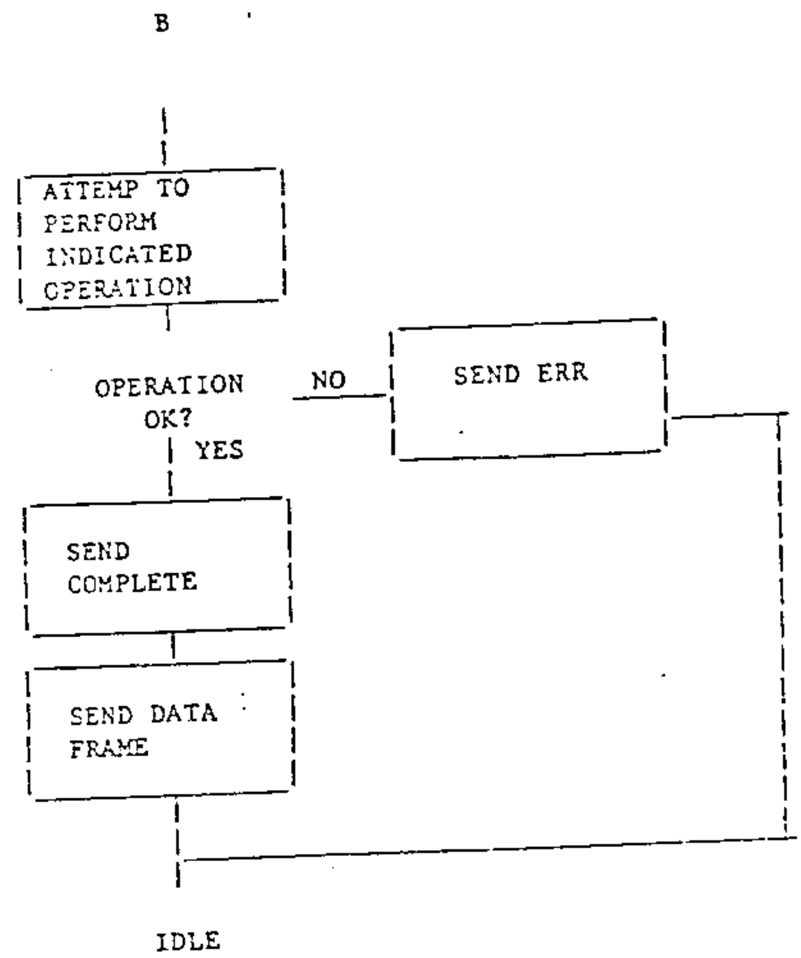


DATA FRAME TO PERIPHERAL
(DATA SEND)

```
           ---+             +------//-----------------------------------
              |             |
COMMAND-      |             |
              + ----------- +


                +--------+             +---//---+
              . |        |             |        |
DATA OUT        | CMND   |             |  data  |
           -----+ Frame  +--------//---+  frame +-----------------------


                              +-+                         +-+      +-+
                              | |                         | |      | |
                              | |                         | |      | |
DATA IN    -------------------+ +--//---------------------+ +-//--+ +-
                              ACK                         ACK     CMPL

              | |        | | | |       |         |        | |      |
              | |        | | | |       |         |        | |      |

               t0        t1  t2   t3               t4          t5
```

Timing parameters are given in section 4.4, bus timing section

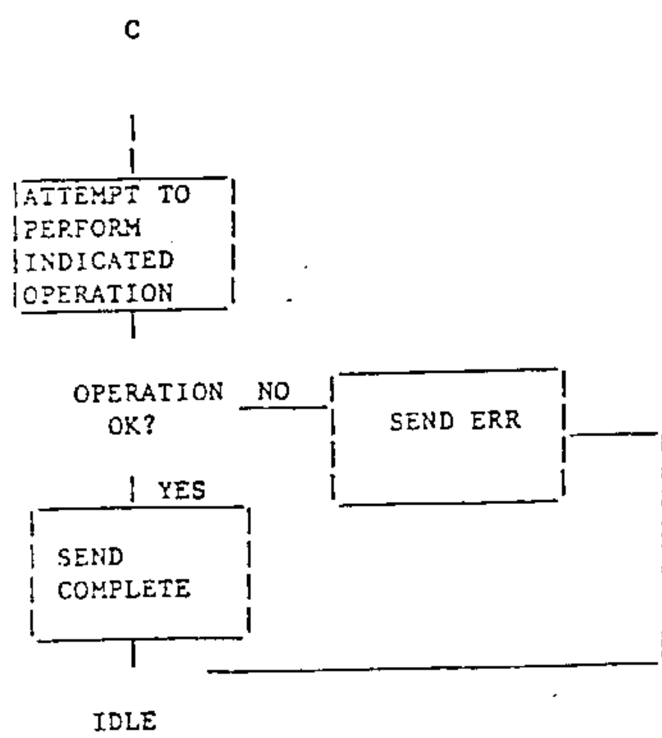
DATA FRAME TO COMPUTER
(Data Receive)



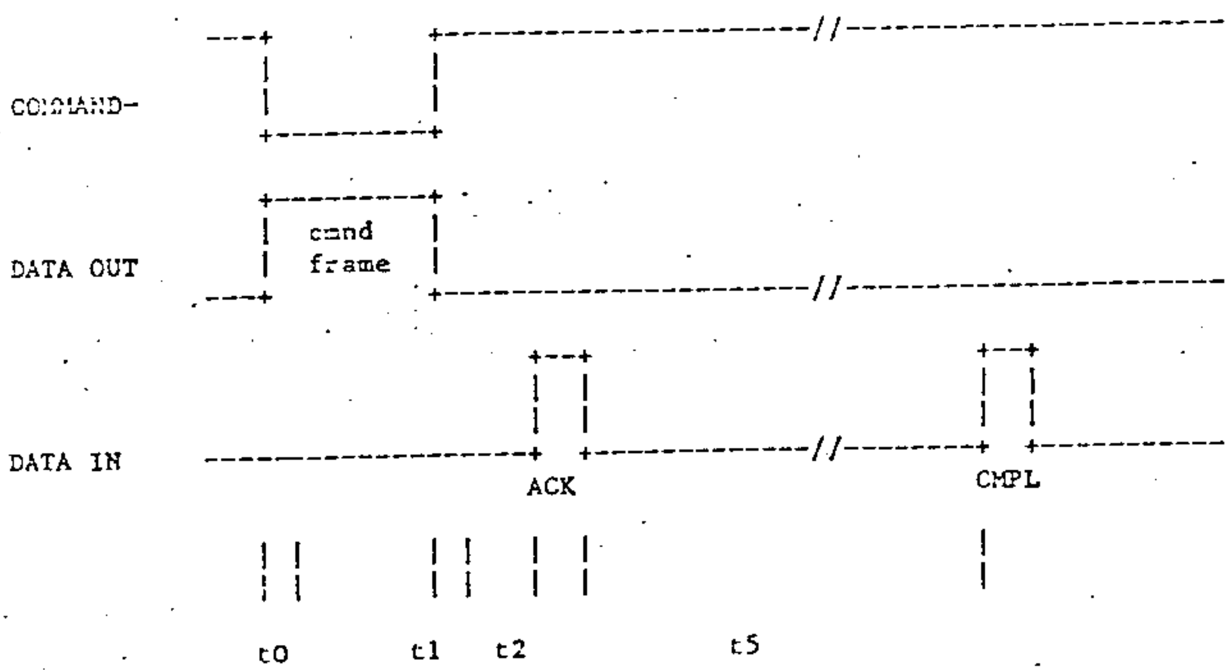
DATA FRAME TO COMPUTER
(Data Receive)

```
                 ---+               +----------//-----------//------------
                    |               |
COMMAND-            |               |
                    +---------------+

                      +-----------+
                      |  cmnd     |
DATA OUT              | frame     |
                 -----+           +-----------//-----------//------------

                                         +-+       +-+  +---//----+
                                         | |       | |  |  data   |
                                         | |       | |  | frame   |
DATA IN          ------------------------+ +--//---+ +--+         +---------
                                         ACK       CMPL

                      ||          ||     | |       |
                      ||          ||     | |       |

                      t0         t1    t2        t5
```

IMMEDIATE



IMMEDIATE sequence:



4.4 BUS TIMING

t0 is the delay between the lowering of $\overline{\text{COMMAND}}$ and the transmission of the first byte of the command frame. The computer generates this delay.

t0 (min) = 750 usec.
t0 (max) = 1600 usec.

t1 is the delay between the transmission of the last bit of the command frame and the raising of the $\overline{\text{COMMAND}}$ line. This delay is generated by the computer.

t1 (min) = 650 usec.
t1 (max) = 950 usec.

t2 is the delay between the raising of $\overline{\text{COMMAND}}$ and the transmission of the ACK byte by the peripheral. The peripheral generates this delay.

t2 (min) = 0 usec
t2 (mac) = 16 msec.

t3 is the delay between the receipt of the last bit of the ACK byte and the transmission of the first bit of the data frame by the computer. The computer generates this delay.

t3 (min) = 1000 usec.
t3 (max) = 1600 usec.

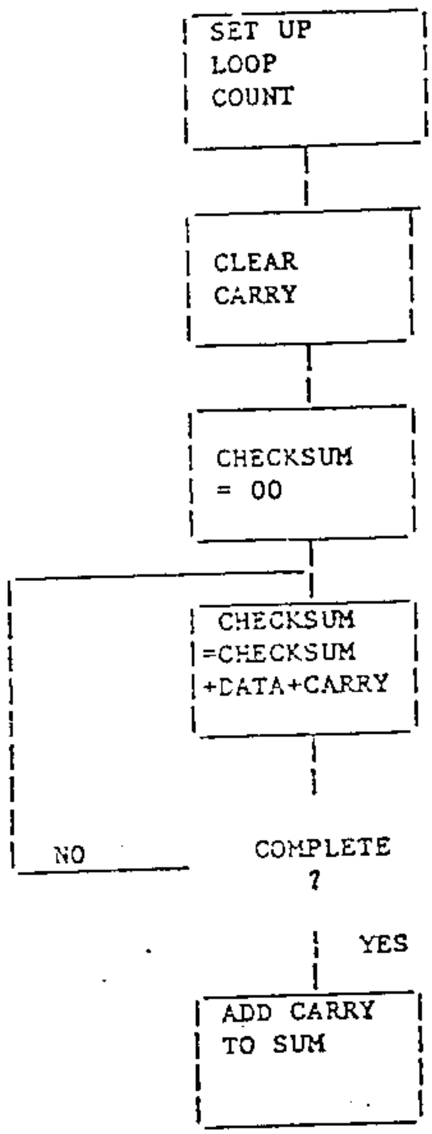
t4 is the delay between the transmission of the last bit of the data frame and the receipt of the first bit of the ACK byte by the computer. The peripheral generates this delay.

t4 (min) = 850 usec.
t4 (max) = 16 msec.

t5 is the delay between the receipt of the last bit of the ACK byte and the first bit of the COMPLETE byte by the computer. The peripheral generates this delay.

t5 (min) = 250 usec
t5 (max) = 255 sec. (handler dependent)

All checksums used as part of the serial bus protocol are simple 8-bit arithmetic sums using the carry bit, as shown below:



For instance, the checksum for two bytes each of value \$80 would be \$01, not \$00.

APPENDIX A

THE PRINTER HANDLER CHARACTERIZATION

A simple O/S resident handler exists for common support of all ATARI hard-copy peripherals. Characterization of this handler follows for the sake of allowing future SIO printers to utilize the same resident handler.

The handler was written to support a line-at-a-time printer (the 820™). For this reason, the handler always buffers data into 820™ line -length data frames. (40, 29, or 20 characters.)

PRINTER COMMANDS

The printer supports two types of commands, WRITE (print data) and GET STATUS (printer returns status to the CPU).

PRINT LINE

The computer sends a command frame of the format shown below:

```
Device I.D.  = $40
Command byte = $57
Auxilliary 1 = doesn't matter.
Auxilliary 2 = $4E for normal print, $53 for sideways, or $44 for double
               width.
Checksum     = checksum of bytes above.
```

The computer sends a data frame of the format shown below:

```
Leftmost character of line (column 1)
Next character of line (column 2).

Rightmost character of line (column 40, 29, or 21).
Checksum byte
```

Note that the data frame size is variable, either 41, 30, or 21 bytes in length, depending upon the print mode specified in the command frame. Note that, for a printer printing in horizontal mode only, the value of AUX2 is of no significance except to indicate the byte-count per frame that the printer should anticipate.

Special characters are transferred to the printer to invoke special functions. The character to be universally recognized is the EOL (End Of Line) character ($9B). This code tells the printer to print its buffer and advance to the beginning of the next line. Note that when an EOL occurs somewhere in the middle of the data frame, the remainder of the frame is padded with spaces ($20) or nulls, which remaining characters should be ignored (i.e., not added to the next line's print buffer.)

The Printer (cont')

Note also that, for 80 column or 132 column print functions, two or more data frames from the CPU may be received before an EOL is received. (Assuming 40 byte + 1 byte checksum data frame lengths.)

GET STATUS

The computer sends a command frame of the format shown below:

Device I.D. = $40
Command byte = $53
Auxilliary 1 = doesn't matter.
Auxilliary 2 = doesn't matter
Checksum = checksum of bytes above.

The printer controller responds with a data frame of the format shown:

command stat.
AUX2 of prev.
timeout
(unused)
checksum

The command status contains the following status bits:

Bit-0 indicates an invalid command frame was received.
Bit-1 indicates an invalid data frame was received.
Bit-7 indicates an intelligent controller (normally = 0).
where bit-0 is the least-significant-bit.

The next byte contains the AUX2 value from the previous operation.

The timeout byte contains a controller provided maximum timeout value (in seconds).

APPENDIX B

THE DISK DRIVE

SIO INTERFACE

The controller serial bus I.D.s range from $31 (for 'D1') to $34 (for 'D4').

The controller supports the following SIO commands.

GET STATUS

e computers sends a command frame of the format shown below:

```
Device I.D.   = $31-34.
Command byte  = $53.
Auxilliary 1  = doesn't matter.
Auxilliary 2  = doesn't matter.
Checksum      = checksum of bytes above.
```

The controller formats a four bytes status frame and sends it to the serial bus preceded by a COMPLETE byte and followed by a checksum of the frame.

The status frame format is shown below:

```
+-+-+-+-+-+-+-+-+
| command stat. |
+-+-+-+-+-+-+-+-+
| hardware stat.|
+-+-+-+-+-+-+-+-+
| timeout (lsb) |
+-             -+
| timeout (msb) |
+-+-+-+-+-+-+-+-+
```

The command status contains the following status bits:

Bit-0 = 1 indicates an invalid command frame was received.
Bit-1 = 1 indicates and invalid data frame was received.
Bit-2 = 1 indicates that and operations was unsuccessful.
Bit-3 = 1 indicates that the disk is write protected.
Bit-4 = 1 indicates active/standby (motor on/off).

The hardware status byte contains the inverted value of the status register of the FD1771 Floppy Disk Controller, for the last command issued. See Appendix B for information relating to the meaning of each bit in the byte.

The timeout byte contains a controller provided maximum timeout value (in seconds), for theworst case command, to be used by the handler. The worst case operation is a disk format, and the timeout value is $00E0 (244 seconds).

PUT SECTOR

The controller receives a command frame containing the following information:

```
Device I.D. = $31-34.
Command byte = $50.
Auxiliary 1 = sector number (lsb).
Auxiliary 2 = sector number (msb).
Command frame checksum.
```

Put Sector (Cont')

The controller then acknowledges the command frame and waits for a data frame. When the data frame is recieved, it is first acknowledged and the the frame data is written to the specified sector and finally a COMPLETE bytes is sent.

PUT SECTOR (WITH VERIFY)

The computer sends a command frame of the format shown below:

Device I.D. = $31-34.
Command byte = $57.
Auxiliary 1 = low byte of sector number.
Auxiliary 2 = high byte of sector number (1-720).
Checksum = checksum of bytes above.

The computer sends a data frame of the format shown below:

128 data bytes.
Checksum byte.

The disk controller writes the frame data to the specified sector, then sends a COMPLETE byte value which indicates the status of the operation.

GET SECTOR

The computer sends a command frame of the format shown below:

Device I.D = $31-34
Command byte = $52.
Auxilliary 1 = low byte of sector number.
Auxilliary 2 = high byte of sector number (1-720).
Checksum = checksum of bytes above.

The disk controller sends a data frame of the format shown below:
128 data bytes.
Checksum byte.

FORMAT DISK

The computer sends a command frame of the format shown below:

Device I.D. = $31-34.
Command byte = $21.
Auxilliary 1 = doesn't matter.
Auxilliary 2 = doesn't matter.
Checksum = checksum of bytes above.

FORMAT DISK

The disk controller completely formats the disk (generates 40 tracks of 18 soft sectors per track with the data protion of each sector equal to all zeroes) and then reads each sector to verify its integrity. A data frame of 128 bytes plus checksum is returned in which the sector numbers of all bad sector (up to a maximum of 63 sectors) are contained, followed by two consecutive bytes of $FF. If there are no bad sectors on the disk the first two bytes of the data frame will contain $FF.

APPENDIX C

erial Bus I.D. and command summary

Serial bus device I.D.s

Floppy disks	D1-D4	$31-34
Printer	P1	$40
RS-232-C	P2	$4F
	R1-R4	$50-53

Serial bus Control codes

ACK	-	$41 ('A')
NAK	-	$4E ('N')
COMPLETE	-	$43 ('C')
ERR	-	$45 ('E')

Serial bus command codes

READ	- $52 ('R')	Disk
WRITE/VERIFY	- $57 ('W')	Printer/Disk
STATUS	- $53 ('S')	Printer/Disk
WRITE/NO VERIFY	- $50 ('P')	Disk
FORMAT	- $21 ('!')	Disk
READ SPIN	- $51 (' ')	Disk
MOTOR ON	- $55 (U)	Disk
VERIFY SECTOR	- $56 (V)	Disk
POLL	- $3F (?)	
SEND RELOCATOR	- $21 (!)	
SEND HANDLER	- $26 (&)	