SMC-200 O.E.M. REFERENCE MANUAL

R-SMC-201-X

July 30, 1982

SMC-200 O.E.M Reference Manual

Section	Title	Page
1	Introduction	3
2	Hardware Theory of Operation	
3	Software Theory of Operation	7
4	Installation	29
5	Maintenance	35
6	Sample Driver Routines	41

This page left intentionally blank.

Introduction

Section 1

Introduction

Section	Title	Page
1.1	Scope	5
1.2	Content	5
1.3	Product Description	5
1.4	Specifications	6
1.5	References	6

1.1 Scope:

The intent of this manual is to provide the user of the KONAN SMC-200 Controller all the necessary data to integrate the controller into his system and, if the user so elects, to maintain the controller. No attempt is made to cover disk drives in this manual.

1.2 Content:

The contents of this manual have been divided into seven major sections. The seven sections are as follows:

Section 1, Introduction

Describes the scope and the content of the manual, provides a generalized product description and references related publications.

Section 2, Hardware Theory of Operation

This section contains information on commands, timing, and schematics. It is provided only after completion of a non-disclosure agreement.

Section 3, Software Theory of Operation

Contains the programming information required to use the KONAN SMC-200 Controller.

Section 4, Installation

This section contains information concerning the installation and cabling of the KONAN SMC-200 Controller.

Section 5, Maintenance

This section contains the maintenance information required to repair the KONAN SMC-200 Controller.

Section 6, Sample Driver

1.3 Product Description:

The KONAN SMC-200 Controller interfaces disk drives with a flat cable storage module interface to computers utilizing an S-100 Bus. Each controller is capable of handling two drives, providing a maximum storage capacity of more than 1.2 billion bytes of storage per controller. The SMC-200 is format compatible with it's predicessor the SMC-100.

1.4 Specifications:

Power requirements:

	Typical	Maximum
-16V	325mamp	400mamp
+8V	1.5amp	2amp

NOTE: The -16V load is reduced by 125 mamp when power save is on.

1.5 References:

The following publications contain information pertinent to KONAN's SMC-200 interface:

1. Magnetic Peripherals Inc. (CDC)
   Flat cable interface specification for the SMD, MMD, and CMD families.
   Specification number 64712400.

2. Fujitsu America Inc.
   M2201 Disk cartridge drive OEM reference manual

3. Fujitsu America Inc.
   M225X Fixed disk unit (FDU) OEM reference manual

4. Memorex Corporation
   601 Disk storage drive, Product specifications
   601.80-02

5. Konan's SMC-200 CP/M Support Package #R-SMC-202-A.

Software Theory of Operation

Section 3

Software Theory of Operation

Section	Title	Page
3.1	General Software Theory	9
3.2	Specific Commands	10
3.3	Select Operation	18
3.4	Seek Operation	19
3.5	Formatting	20
3.6	Write Operation	22
3.7	Read Operation	23
3.8	Data Recovery Procedures	24
3.9	Disk Address Changes	26
3.10	Mapping	28

3.1 General Software Theory:

The KONAN SMC-200 Controller appears to the system as six I.O. locations. The six I.O. locations provide the following functions:

I.O. LOCATION		FUNCTION	
0 and 1	(write Only)	Bus Drivers	Controls the bus and select lines to the disk drivers.
2	(write only)	Command Register	Used to initiate the following: DMA and disk transfers, disk selection, head selection, seeks, and recovery operations.
3	(write only)	Sector Address	Selects the sector to be used for the next read or write operation.
4	(read only)	Sector Buffer	Allows software to read and write the sector buffer.
5 and 6	(not used)		
7	(read, write)	Status	Allows done, ready and error status to be read. Writing to this I.O. port resets the sector buffer address.

Each of these I.O. locations are accessed by use of the IN and OUT instructions with the appropriate address. The most significant five bits are set by the customer (see Section 4, Installation). These five bits are represented by the letter "X" in the following discussions. The least significant three bits are the I.O. locations referenced above.

3.2 Specific Commands:

This subsection will describe in detail each of the software commands.

3.2.1 Bus and Select Lines --

Out instructions to X0 and X1 will place data on the bus and select lines. No function will be initiated.

Figure 3.1, Bus and Select Commands, illustrates the function of each bit in X0 and X1.

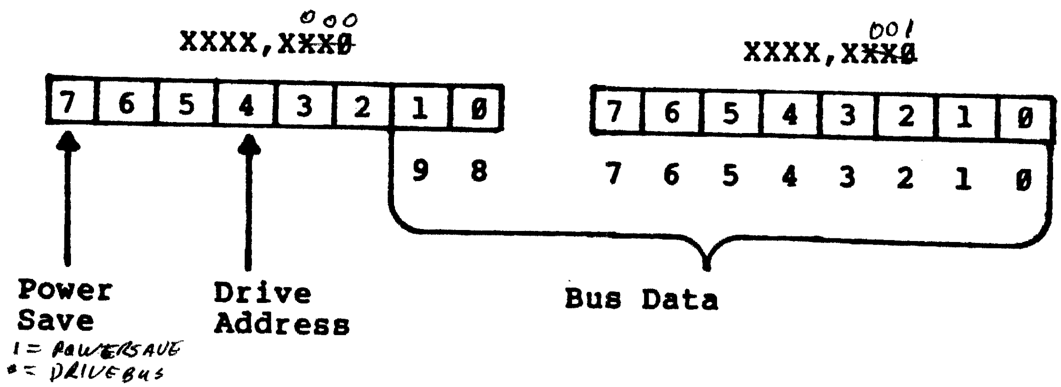


Figure 3.1, Bus and Select Commands

Definition of the bus data depends upon the command following the X0 and X1 OUT instructions. Figure 3.2 defines the bus data for each of the three possible bus line commands.

Bit seven (7) of X0 is a power save option. When this bit is set true, to a one (1), the bus bits 0 thru 9 will not be driven. This decreases the -16V draw and allows the controller to run cooler. To use this feature, set Power Save false, (0), when entering your driver routines and true, (1), when exiting. When not using power fail, simply set bit 7 of X0 to a 0 at all times.

Note: the Power Fail Option does not deselect the drive. The sector counter and other status remains valid, preventing and performance loss.

	BUS BIT	CYLINDER ADDRESS	HEAD SELECT	FAULT RECOVERY
MSB	9	2^9	X	Release
	8	2^8	X	Strobe Late
	7	2^7	X	Strobe Early
	6	2^6	X	Return to zero
	5	2^5	X	X
	4	2^4	2^4	Fault Clear
	3	2^3	2^3	Servo Offset Minus
	2	2^2	2^2	Servo Offset Plus
	1	2^1	2^1	X
LSB	Ø	$2^Ø$	$2^Ø$	X

X indicates don't care

Figure 3.2, Bus Definitions

3.2.2 Command Register --

All disk operations are initiated by OUT instructions to the command register, X2. Figure 3.3 defines the command register.

XXXX,XØ1Ø

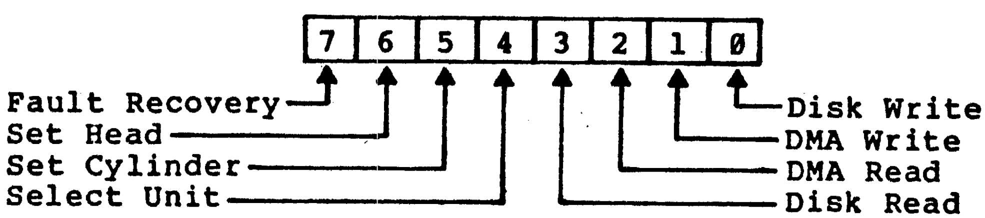


Figure 3.3, Command Register

The following is a list of all legal commands and the setup requirements of each.

ØØH Resets DONE and INTERRUPT, no data set-up required.

Ø1H Writes the contents of the sector buffer to the disk. The data is written on the currently selected drive, head and track at the sector selected by the sector address register (OUT X3).

Ø8H Reads a sector from the disk into the sector buffer. The sector read is defined by the sector register and is taken from the currently selected drive, head, and track.

1ØH This command causes the select tag line to be strobed. Prior to issuing this command the address of the drive to be selected is set into the XØ register.

2ØH Causes the selected drive to seek to the cylinder selected by Bus Ø - 9 lines (set by OUT XØ and X1).

4ØH Selects the disk head identified by the bus lines (set by OUT X1).

8ØH Performs the fault clear function identified by the bus lines.

88H Reads a sector from the disk (same as Ø8H) with error recovery functions selected by OUT XØ and X1.

3.2.3 Sector Address --

The sector address register is loaded with an output instruction to X3. The next read or write disk operation will be at the sector address loaded. The least significant bit of the sector address register is bit 0.

XXXX,X011

7	6	5	4	3	2	1	0
MSB							LSB

Figure 3.4, Sector Address Register

3.2.4 Sector Buffer --

The sector buffer is a 1K RAM on KONAN's SMC-200 Controller that is logically placed between the disk and the user's memory. The purpose of the RAM is to overcome an inherent transfer rate problem and to free the user from memory timing restraints. Generally only a portion of the sector buffer is used. The size of this segment is equal to the number of bytes of user data plus the number of bytes of header. Most commonly, 256 bytes of data and 4 bytes of header are used. This means that 260 sector buffer locations would be used (0 to 259). The sector buffer size, in this case 260 bytes, must be strapped on the controller. (See Section 5, Installation.)

The sector buffer can be accessed in four different ways. These are:

1. I.O. OUT instruction (X4)
2. I.O. IN instruction (X4)
4. Disk Write (01 to X2)
5. Disk Read (08 to X2)

In the first two instances, the transfer is between the users processor and the sector buffer. The last two transfer data between the disk and the sector buffer. The sector buffer has its own address counter. Any sector buffer access will cause the address counter to increment once for each byte written or read. The sector buffer address counter cannot be directly accessed by software, except to set it to zero. The format of the sector buffer I.O. commands is shown in Figure 3.5, Sector Buffer Command.

XXXX,X100

7	6	5	4	3	2	1	0	
7	6	5	4	3	2	1	0	DATA

Figure 3.5, Sector Buffer Command

Figure 3.6, Typical Sector Buffer Format, depicts a buffer of 260 characters. The first four characters are used for header and the following 256 for user data. To load the buffer, the user would first insure that the sector buffer address was zero. This is done with an OUT to X7. (Accumulator data is ignored.) Header data may now be entered. This is accomplished by placing the appropriate header data in the accumulator and then performing an OUT to X4, the sector buffer. The first byte would be written into address 000 and then the address would automatically be incremented to 001. The next three header bytes are written in the same manner, leaving the sector buffer address set to 004. The user continues this method of transfer for the data.

```
Address

000  HEADER BYTE 1
001  HEADER BYTE 2
002  HEADER BYTE 3
003  HEADER BYTE 4
004  DATA BYTE 1
005  DATA BYTE 2
 .
 .
 .
 .
 .
 .
 .
259 DATA BYTE 256, END OF SECTOR
 .
 .
 .
 .
 .
 .
1023 END OF SECTOR BUFFER
```

Figure 3.6, Typical Sector Buffer Format

3.2.5 Status --

The status register serves two purposes. Primarily, it is accessed by an IN instruction to provide disk and controller status information. It is also used with an OUT instruction to set the sector buffer address to zero. In this case, the contents of the accumulator are ignored. Figure 3.7 illustrates the status register bit definition when used with the IN instruction.

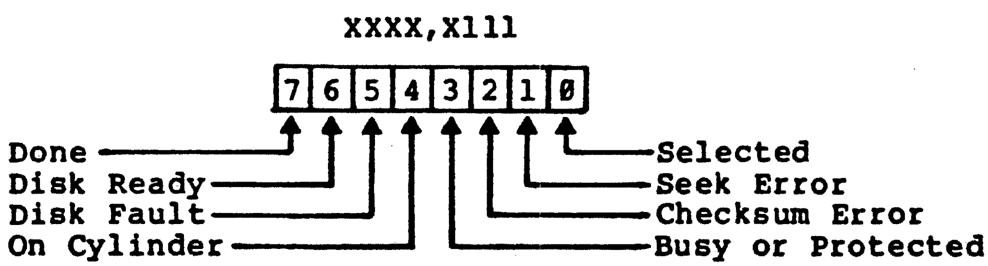


Figure 3.7, Status Register Bit Definition

The following defines the status bits:

DONE
Is set true at the completion if a seek operation (successful or unsuccesful) and upon the completion of a disk transfer. Selecting a disk may also cause DONE. If interrupts are enabled, one will be generated each time DONE goes true.

DISK READY
Indicates that the selected disk is up to speed, the heads are positioned over the recording tracks,and no fault condition exists in the drive.

DISK FAULT
Indicates a fault condition exists in the selected drive.

ON CYLINDER
Indicates the heads are positioned over a track.

BUSY
Used only on dual channel disks. Indicates that the selected drive is currently being accessed by, or is reserved by, another controller.

CHECKSUM ERROR
Indicates a read error has occurred. Will be cleared by the next successful read.

SEEK ERROR
Indicates a seek error has occurred on the selected drive. The error may only be cleared by performing a RTZ (Return to Zero).

SELECTED
Indicates that the last select operation was successful.

WRITE PROTECT
Indicates that current selected head is Write Protected.

NOTE: Bit 3 is selected via a jumper to be Write Protect or Busy.

3.3 Select Operation:

The select operation requires two output instructions. The first output places the address of the drive to be selected on the address lines. Then the select command is issued. The format to specify the address is shown in Figure 3.8.

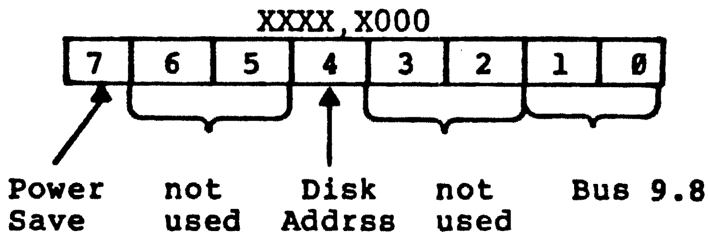


Figure 3.8, Select Address Register

The select command is a Hex 10 to register X2. Selection of the first drive is accomplished with a 00 out to port X0 followed by a 10 Hex out to X2. The state of Power Save and Bus 8.9 do not matter when doing a select. The Select bits and TAG lines, (command strobes), are not power down by Power Save.

```
MVI     A,10H       SET ADDRESS FOR DRIVE 1
                    IN ACCUMULATOR
OUT     90H         LOAD SELECT LINES
MVI     A,10H       SELECT COMMAND TO ACCUMULATOR
OUT     92H         ISSUE COMMAND
IN      97H         GET STATUS
ANI     01H
JZ      ERROR       IF SELECTED FAILED JUMP TO ERROR
XRA     A
OUT     92H         CLEAR ANY FALSE INTERRUPT
```

The select sequence is instantaneous. A false DONE-INTERRUPT may be generated during the select sequence. This can be cleared by an OUT X2 (92 in our example) to clear DONE. The selected drive will remain selected until a new select sequence is performed or the drive is powered off.

3.4 Seek Operation:

The heads may be positioned over any cylinder by a seek operation. The bus lines are set to the desired track address and then a seek command starts the seek. Bus lines are set as follows:

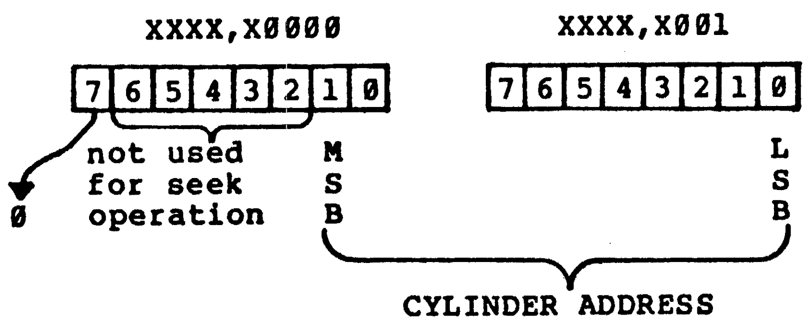


Figure 3.9, Cylinder Address

The cylinder command is issued with an output of 20H to the X2 register. The following is an example of a seek subroutine. It is assumed that the drive was previously selected.

```
* CALL THE FOLLOWING WITH THE DESIRED CYLINDER
* ADDRESS IN REGISTERS D & E
SEEK MOV A,D
     OUT 90H       PLACE MSB OF CYLINDER ADDRESS ON BUS
     MOV A,E
     OUT 91H       PLACE LSB OF CYLINDER ADDRESS ON BUS
     MVI A,20H     LOAD ACCUMULATOR WITH SEEK COMMAND
     OUT 92H       DO THE SEEK
```

Following a seek it is necessary to wait until the drive has completed the seek before any further disk operation is done. This wait can be accomplished via interrupts allowing the processor to be free for other tasks, or by waiting for a done status.

```
* WAIT FOR SEEK DONE OR ERROR
WSK IN 97H             GET STATUS
    ANI 80H            CHECK FOR DONE
    JZ WSK
    ANI 02H
    RZ                 RETURN IF ERROR FALSE
    JMP ERROR          SEEK FAILED
```

3.5 Formatting:

Due to the extremely high density on today's state of the art disk drives, it is desirable to provide assurances that data is written and read from the appropriate locations on the disk. Prior to using a new disk surface, it should be formatted; that is, address data is written at each sector on the disk. Subsequent reading of the disk will involve verification of the address data prior to placing the data into user memory. The format data (called header), is at the beginning of each sector and physically contiguous with it. This technique conserves disk space but requires the header data to be rewritten with each write.

Figure 3.10, Header Format, shows the suggested header format. The following points should be noted:

1. A new header must be rewritten with each write.

2. Header should be verified for each sector on a read and one sector should be read prior to writing if a seek has been performed.

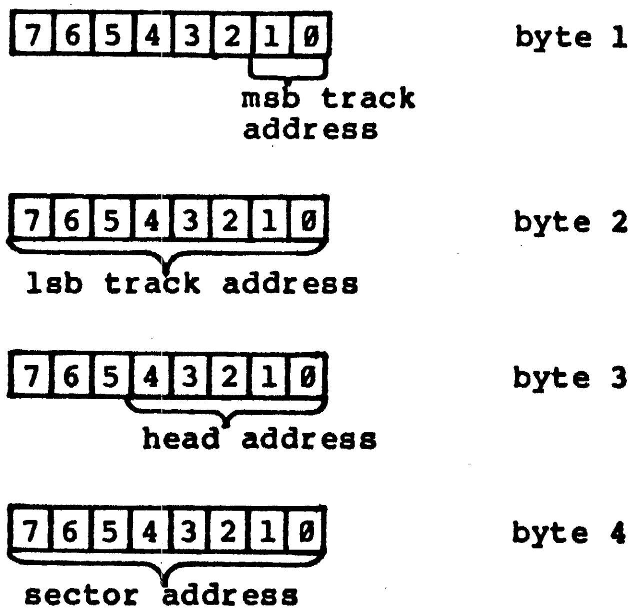
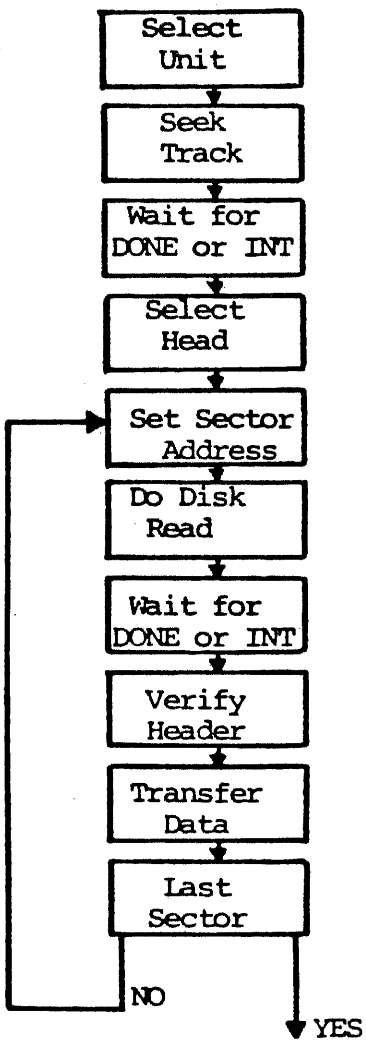


Figure 3.10, Header Format

Additional information on formatting is contained in the Installation Section.

3.6 Write Operation:

The write requires two separate data transfers. Data is first transferred to the sector buffer from the users RAM and then the data is transferred to the disk. Figure 3.11, Typical Write, is an example of a write operation.

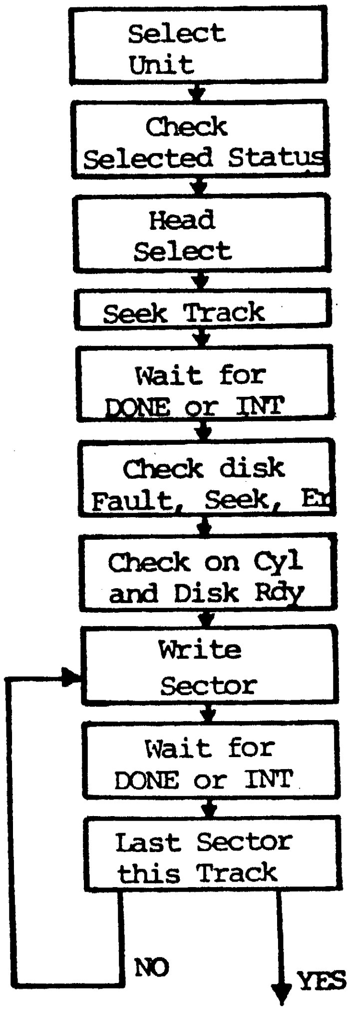



```
  OUT 97H   ZERO SECTOR BUFFER ADDRESS
* PLACE THE TWO MOST SIGNIFICANT BITS
* OF TRACK IN ACCUMULATOR
  OUT 94H   HEADER WORD 1
* PLACE LSB OF TRACK IN THE ACCUMULATOR
  OUT 94H   HEADER WORD 2
* PLACE HEAD IN THE ACCUMULATOR
  OUT 94H   HEADER WORD 3
* PLACE SECTOR IN THE ACCUMULATOR
  OUT 94H   HEADER WORD 4
  OUT 93H   SET SECTOR
  PLACE BYTE OF DATA IN ACCUMULATOR
  OUT 94H
  REPEAT ABOVE TWO LINES SECTOR SIZE TIMES
  OUT 97H
  XRA A  CLEARS THE BUS
  OUT 90H
  OUT 91H
  MVI A,01H DISK WRITE COMMAND
  OUT 92H DO DISK WRITE
```

Figure 3.11, Typical Write

3.7 Read Operation:

The disk is read by first positioning the heads, selecting the appropriate head, and then performing the read. This leaves the read data in the sector buffer. The sector buffer data is then transferred to the user RAM. The read operation will be explained by an example. Figure 3.12, Typical Read.




```
* PLACE SECTOR # INTO ACCUMULATOR
  OUT 93H        LOAD SECTOR ADDRESS
  XRA A          CLEAR THR BUS
  OUT 9ØH
  OUT 91H
  OUT 97H        ZERO SECTOR ADDRESS
  MVI Ø8H
  OUT 92H        DO THE DISK READ
* WAIT FOR INTERRUPT OR DONE
  OUT 97H        ZERO SECTORBUFFER ADDRESS
  IN 94H         GET FIRST HEADER WORD
* CHECK ACCUMULATOR FOR PROPER MSB OF TRACK
  ON 94H         GET SECOND HEADER WORD
* CHECK ACCUMULATOR FOR PROPER LSB OF TRACK
  IN 94H         GET THIRD WORD OF HEADER
* CHECK ACCUMULATOR FOR PROPER HEAD
  IN 94H         GET LAST WORD OF HEADER
* CHECK FOR CORRECT SECTOR
  IN 94H         READ DATA BYTE
  MOVE INTO USER MEMORY
  REPEAT ABOVE TWO LINES SECTOR SIZE TIMES
```

Figure 3.12, Typical Read

3.8 Data Recovery Procedures:

Typically, storage module drives have recoverable error rates of approximately one in every 10^{10} bits. Non-recoverable errors occur at a rate of one in every 10^{13} bits. Proper recovery procedures can, therefore, improve the users data integrity 1000 times over systems not using recovery procedures. Some SMD drives provide a head offset and others do not. Proper error recovery includes 3 attempts to read the record at zero offset and nominal strobe, and 2 attempts to read at each offset position (if so equipped) with early, nominal, and late strobes (19 reads).

Figure 3.13, Data Recovery Procedures, shows one approach to recovery on a drive with offset.

Care should be taken to insure that track offset is turned off prior to attempting to seek or write. When track offset is set or changed, a delay is required prior to performing the next disk function. Fortunately, a full disk revolution must take place between attempts to read the same sector. This will generally provide adequate time to change the offset. Following a successful recovery, it would be possible, however, to turn off the offset and select a following sector too soon. Software must, therefore, provide a delay if a track offset condition was true when exiting error recovery routines. This delay varies between various disk manufacturers. Reference appropriate drive manufacturers specifications for details.

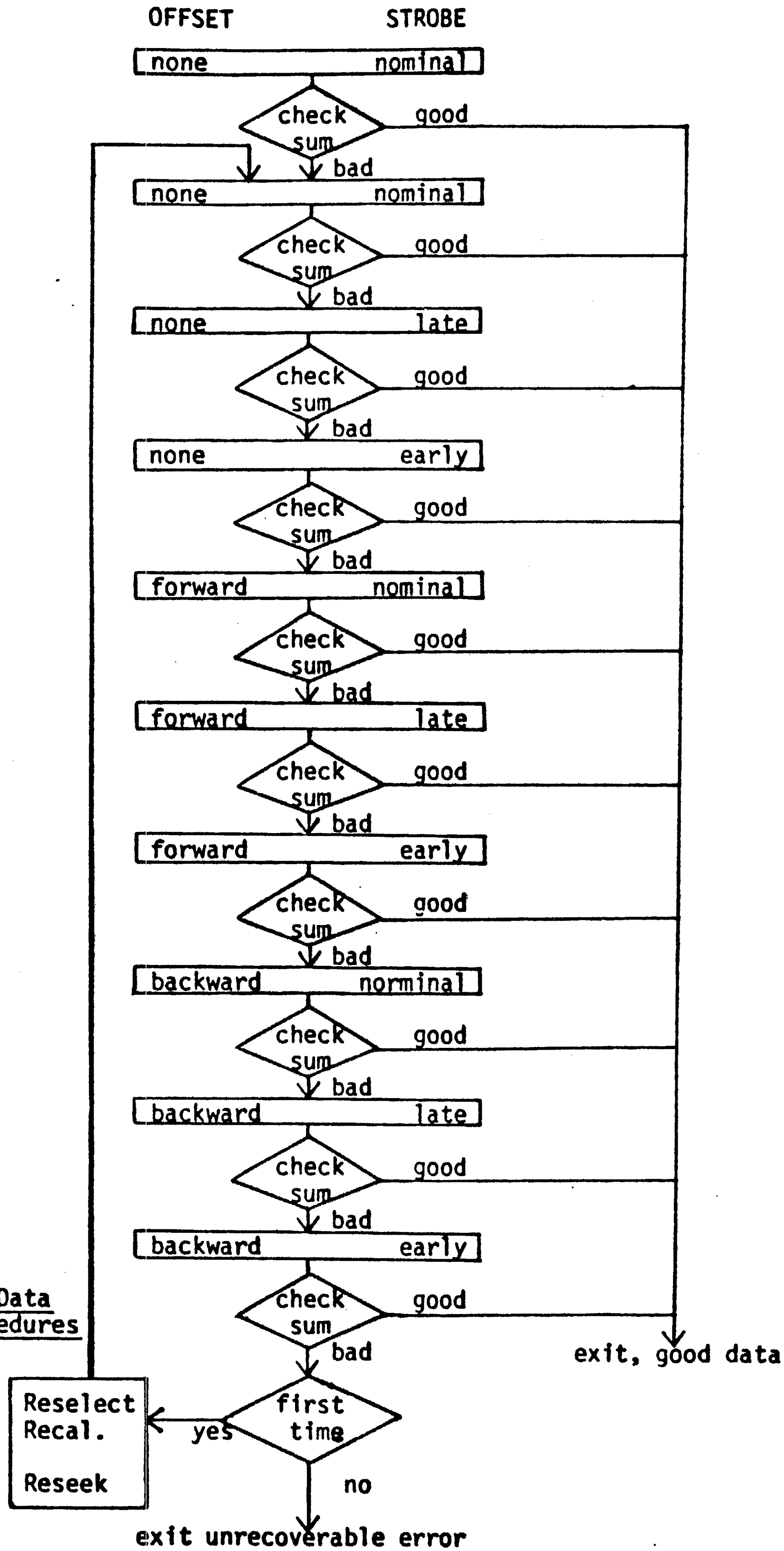


Figure 3.13, Data Recovery Procedures

3.9 Address Changes

When the disk address changes, ie. new head, track, unit, or sector, the new address is selected by software. Some of these selections are immediate, (head, unit, sector) and one, track, requires some time to perform. The following points are of interest:

1. Unit select causes the sector counter to become invalid.

2. Head select is usually immediate with no adverse effects on the sector counter. CMD DRIVES ARE AN EXCEPTION! When the volumn bit changes in a CMD drive it must be followed by a seek operation.

3. Seeking, (track select), requires time for the heads to move. To insure that the sector counter did not get "glitched" during the seek, reads and writes are inhibited until index. This is transparent to the user. It insures proper sector selection but can cause a slight performance decrease.

4. Unit select must always be followed by a seek, even if the cylinder number has not changed.

Figure 3.14 on the following page will insure proper address selection without unnecessary delays.

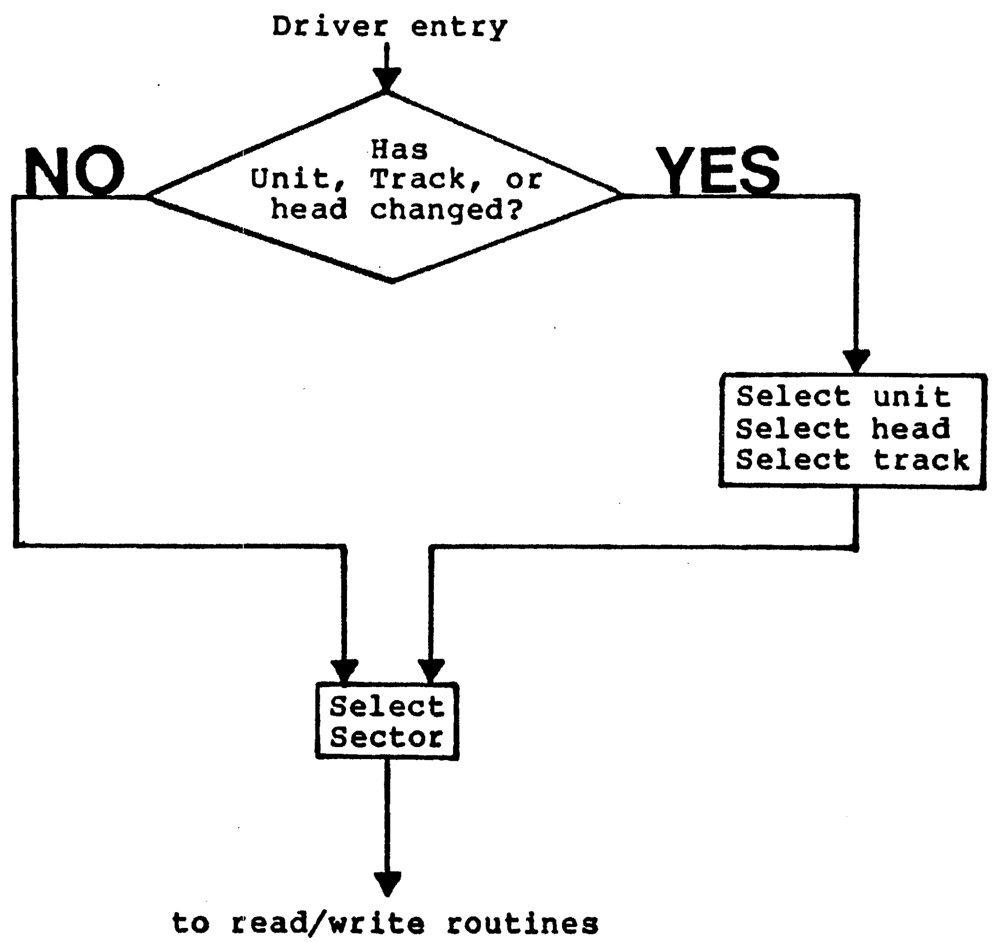


Figure 3.14, Address Selection

If unit, head, or track is changed, all of them are selected. If a CMD will not be used, the Head change test can be deleted, with head selection done with Select sector.

3.10 Mapping

It is advised that all systems have some method of re-allocating disk space in order to avoid media flaws. Generally, this can be handled by the operating system through allocation maps or special directory entries. This task can also be performed in the disk driver code, making it transparent to the operating system. An excellent example of track mapping can be found in Konan's CP/M support package.

Installation

Section 4

Installation

Section	Title	Page
4.1	Strapping	31
4.2	Installing Controller	32
4.3	Disk Format	33

4.1 Strapping:

4.1.1 Sector Size

Straps A --> E set the sector size. Options are:

A	B	C	D	E	
in	out	out	in	out	256 byte sector
in	out	out	out	in	256 byte sector + four bytes header
in	out	out	out	out	512 byte sector
out	out	out	in	in	512 byte sector + four bytes header

4.1.2 Preamble length

Straps F --> N select the preamble length as follows:

F	1
H	2
J	4
K	8
L	16
M	32
N	64

The preamble may be set to any value from 1 to 127, STRAPS ARE OUT TO ENABLE A COUNT AND IN TO DISABLE. Standard preamble is 31, straps F, H, J, K and L out, M and N in.

4.1.3 Board Address

Straps P --> W select the board address as follows:

W	80 hex
V	40 hex
P	20 hex
S	10 hex
T	8 hex

Installing a strap sets the corresponding address to zero. Removing strap sets it to a 1. Standard base is 90 hex, 10010XXX.
W, S out
V, P, and T in

4.1.4 Busy 1 Protected Option

Strap X-Y selects the definition of bit 3 of the status register.

STRAP Y selects Write Protect, this is the normal setting.
STRAP X selects Busy. Busy is used when two SMC-200's are accessing a shared disk drive. The drive must have the dual port option.

4.2 Installing Controller:

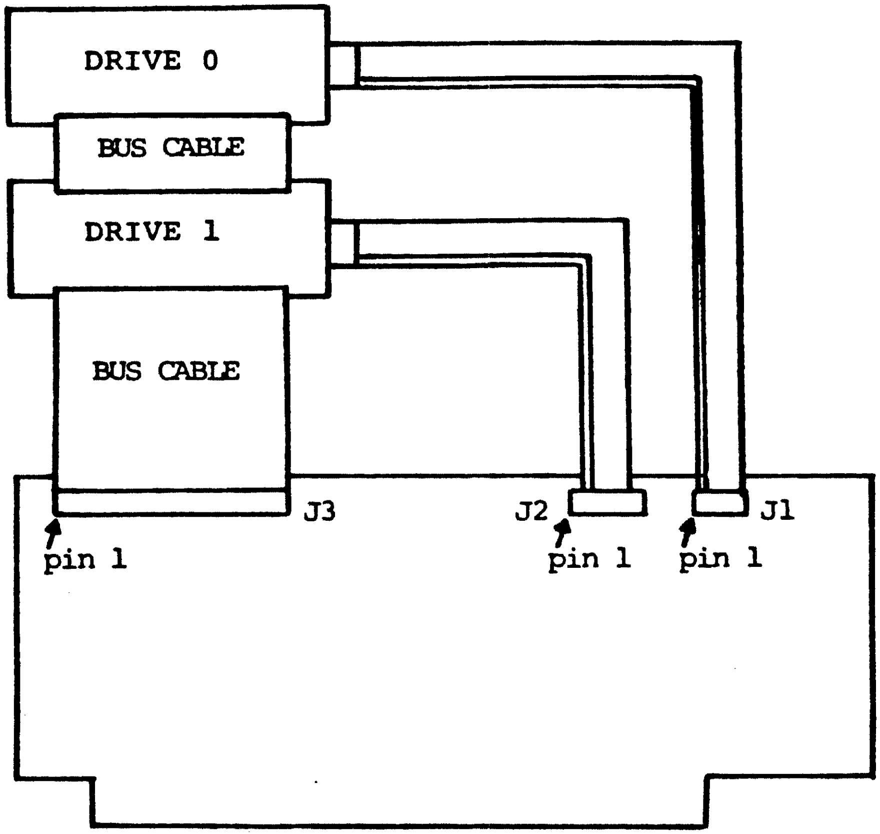
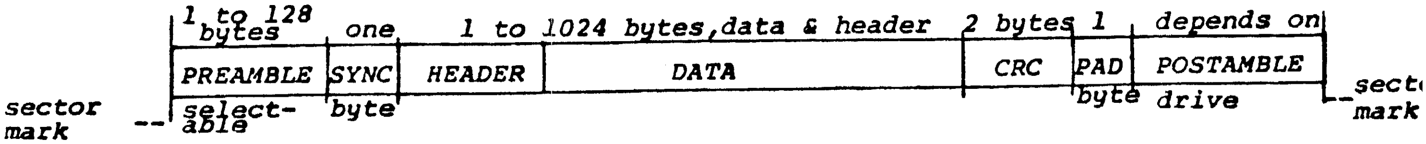


Figure 4.1 Cable Routing

Figure 4.1 shows proper cable routing. The bus cable (the 60 pin flat cable), connects the controller to one drive only. The remaining drives are "daisy-chained" together with bus cables. The bus cables may be connected to the drives in any order indpendent of the drives unit numbers. The final drive must have a terminator. Maximum length of all bus cables combined may not exceed 100 feet. Radial cables connect to each drive. These cables must be routed to the appropriate drive. J1 connects to drive 0, J2 to drive 1.

4.3 Disk Format

The Sector format used by the SMC-200 is as follows:



Preamble can be from 1 to 128 bytes. This must be set equal to or greater than the drive manufacturers minimum. A sync byte follows the preamble.

The header is not required, but due to the high density of SMD drives it is highly recommended that the header be used to identify the head, sector and track address to prevent positioning and head selection errors. All the following explanations and figures will assume a header of four bytes (the header is physically contiguous with data to conserve disk space). The data field (including header) can be any length up to 1024 bytes. This is always followed by three CRC bytes and one pad byte. The postamble will extend to the next sector mark. The length of the sector is equal to the length of the:

preamble + sync + header + data + CRC + pad + postamble

To determine the maximum number of sectors, the total number of data bytes per track must be divided by the number of bytes per sector. Often the division leaves a remainder which will be present on the disk surface as a fragment of a sector which cannot be used. In many cases this remainder can be divided between the sectors to extend the postamble. This may or may not improve reliability.

This page left blank intentionally.

Maintenance

Section 5

Maintenance

Section	Title	Page
5.1	Maintenance Philosophy	37
5.2	Cables	37
5.3	Bringing Up The System	38

5.1 Maintenance Philosophy:

The SMC-200 requires no preventative maintenance. The suggested method of repair is by replacement. When a board fails it should be replaced with a good board and the faulty board returned to KONAN for repair.

5.2 Cables:

When installing the cables on the CMD* Drive pin 1 is the top most pin on the radial and the bus cables.

The radial cable is shielded; therefore, the black stripe must correspond to pin 1.

The bus cable is multiple twisted pairs and the designation for pin 1 is not critical as long as the color code is consistent.

Multiple insertions of these cables have a tendency to weaken the connections and the mass termination to the actual cable. Therefore, it is recommended that as few cable changes as possible be done, using care not to pull on the actual cable itself.

Some of the symptoms of the improper cabling are:

the drive faults (bus cable)
the system hangs (radial cable)

*Due to the popularity of the CDC CMD Drive, it is used here as an example; however, the SMC-200 is compatible with any Storage Module Drive.

5.3 Bringing Up A System:

The recommended procedure for bringing up a system that is provided with a CP/M Support Package is:

1. Run IORAM. This diagnostic checks the capability of the SMC-200 onboard buffer to be read and written via I/O Commands.

2. Run LACEDIAG. A scratch disk must be installed in the drive and it must be up to speed and ready with no faults indicated.

This test serves two purposes:

a. It test the disk subsystem independently of the BIOS or Boot ROM
b. It aids in the location of Media flaws.

IMPORTANT! THIS TEST DESTROYS THE DATA AND MEDIA MAP ON THE DRIVES(S) BEING TESTED.

<u>After running the Diagnostic/Media test the drive must be reformatted (and mapped if desired).</u>

Disk Diagnostic/Media Test Abstract

Upon entering the Konan Media/Diagnostic test the following message is printed:

KONAN MEDIA TEST REVISION N DD-DD-DD where N is the revision and D is the last date updated.
DMA DISABLED BASE OF I.O. PORTS IS 90 SECTOR SIZE = 512
The operator is then prompted as follows:

<u>UNIT</u>? Enter a one (1) digit number corresponding to the unit to test. (Normally 0)

<u>DRIVE TYPE</u>? (C=CMD M=MMD)? Enter C for CMD (ie. Phoenix, M for all others.

<u>CRCTEST</u>? (N=NO Y=YES)? If yes, CRC is force to be bad. If following read fails to detect bad CRC and error is printed. Normally enter N.

<u>BYTE FOR BYTE COMPARE?</u>? Verifies each byte of data in addition to the CRC in header tests. Greatly slows the test. Normally enter N.

TEST READ DATA AT ALL OFFSETS? Causes the strobe early/late and offset forward/backward to be used on each read pass. Aids in finding marginal data. Greatly slows the test. Normally enter N.

FIRST TRACK? Decimal address of the first track to be tested.

LAST TRACK? Decimal address of the last track to be tested.

FIRST HEAD? Address of first head to test with CMD. Zero is for removable, 1 for first fixed, 2 for next, etc.. For MMD, zero = zero, 1 = 1, etc..

LAST HEAD? Address of last head to test.

INTERLACE PATTERN? Interlace pattern. Allows speed of diagnostic to be matched to system/disk.

LAST SECTOR? Last sector to test. Enter Ø to test only one sector per track, 35 to test all 36 sectors.

The operator may interrupt the test at any time by pressing a Control C, this will print the following message:

BREAK TYPE D FOR DOS, R TO RESTART AND CR TO PRINT STATUS AND CONTINUE

Typing D will cause a return to CP/M via a JMP Ø. R will restart the test, and carriage return will print the status as follows:

FULL PASSES=Ø9 R,W PASSES-ØØ37 ERRORS-ØØØØØØ
SECTORS READ=ØØØØE8Ø8 ØØØ2 Ø ØØ

Where FULL PASSES equal the number of times the drive has written and read all selected sectors with the following patterns: All zeroes, All ones, Ripple up, Ripple down, floating zero and floating one. One each read/write pass the base for the ripple and floating patterns changes.

R,W PASSES is a hex number equal to the number of read write passes through the selected portion of the disk.

ERRORS is a total of all errors occurring since the test was started, in hex.

SECTORS READ is the hex number of sectors read.

The last line is the decimal address where the break occured. The first four digits are track, then sector and finally head.

DATA ERRORS PRINT AS FOLLOWS:

```
Decimal         Hex
Track           Track,Head,Sector
    Head        in the form Good/Bad
       Sector
0005  1 33      0005/FFFF 01/FF 21/FF L,BD
0005  1 33      0005/FFFF 01/FF 21/FF L,BD
0005  1 02      0005/FFFF 01/00 02/05 L,GD
0005  1 02      0005/0005 01/01 02/ID L,GD,R
PASS  0002      1 14 0002/FFFF 01/FF 0E/FF N,BD
0002  1 14      0002/FFFF 01/FF 0E/FF N,BD
0002  1 18      0002/FFFF 01/FF 12/FF N,BD
TRCK  H SC        TRACK  HEAD  SCTR  TYPE  WORD,GOOD/BAD
0002  1 18      0002/FFFF 01/FF 12/FF N,BD
0002  1 22      0002/FFFF 01/FF 16/FF N,BD
0002  1 22      0002/FFFF 01/FF 16/FF N,BD
0002  1 26      0002/FFFF 01/FF 1A/FF N,BD
0002  1 26      0002/0002 01/01 1A/0E N,GD,R <RETRY, if retry is
                                              always bad,error is
                                              usually a write error
PASS  0005      1 00 FFFA/FFFA FE/FE FF/FF E,BD
0005  1 00      FFFA/FFFA FE/FE FF/FF E,BD
0005  1 04      FFFA/FFFA FE/FE FB/FB E,BD
0005  1 04      FFFA/FFFA FE/FE FB/FB E,BD
0005  1 08      FFFA/FFFA FE/FE F7/F7 E,BD
TRCK  H SC        TRACK  HEAD  SCTR  TYPE  WORD,GOOD/BAD
0005  1 08      FFFA/FFFA FE/FE F7/F7 E,BD  013,00/08 <word 13 is bad
0005  1 12      FFFA/FFFA FE/FE F3/F3 E,BD  013,00/08  it should have
                                                     been 00,was 08
```

CRC
BD=BAD
GD=GOOD

If bad data,the first four words print in this area. If this is blank then data was OK.

NON-DATA ERRORS

Non-data errors are also displayed. After printing them the status also prints. There errors are:

SELECT ERROR: (could not select unit)
TIMEOUT ERROR: (done loop timed out)
WRITE FAULT ERROR
SEEK ERROR
HEAD SELECT ERROR

SAMPLE DRIVER ROUTINES

Section 6

Sample Driver Routines

Section	Title	Page
6.1	Write Routine	43
6.2	Read Routine	44
6.3	Read Recovery	46
6.4	Disk Address Selection	47
6.5	Seek Routine	50
6.6	Mapper Routine	52

The following sample routines are taken from KONAN'S CP/M Support Package. They interface to CP/M's blocking and deblocking algorithms, using 512 byte sectors.

6.1 Write Routine

The disk address is selected by the KCOMMON routine, covered later. Note that there is no error recovery. The philosophy here is that all write errors are fatal and to retry is risking damage to the data base. (Something must be "broke" in order to get a write error).

```
WRITEHST:                ;ENTRY INTO THE WRITE PROGRAM
        CALL    KCOMMON ;SETS SECTOR,DMA,HEAD,TRACK,UNIT
        RC              ;ERROR RETURN
        XRA     A       ;CLEAR BUS
        OUT     BUSL
        OUT     BUSU    ;BUS IS NOW CLEARED
        OUT     STATU   ;SECTOR BUFFER NOW ZEROED
;       WRITE THE HEADER NOW (TRACK MSB,TRACK LSB,HEAD,SECTOR)
        LXI     H,PTRK  ;POINT TO THE TRACK
        MVI     B,4     ;HEADER BYTE COUNT
HEADWT  MOV     A,M     ;GET BYTE IN ACUMULATOR
        OUT     SCTBU   ;SEND IT
        INX     H
        DCR     B       ;DECREMENT COUNT
        JNZ     HEADWT  ;IF NOT DONE JMP TO HEADER WRITE
        IF      DMA     ;****************************
        MVI     A,Ø2    ;DMA WRITE COMMAND
        OUT     CMND    ;DO THE WRITE
        ENDIF           ;****************************
        IF      NOT DMA ;****************************
        LXI     H,HSTBUF;IO WRITE LOOP
        LXI     D,HSTSIZ
NXIOW   MOV     A,M
        OUT     SCTBU
        INX     H
        DCR     E
        JNZ     NXIOW
        DCR     D
        JNZ     NXIOW
        ENDIF           ;***************************
        OUT     STATU
        MVI     A,1     ;DISK WRITE COMMAND
        OUT     CMND    ;DO THE WRITE
        CALL    WT      ;WAIT FOR IT TO FINISH
        MVI     A,'A'   ;ERROR A IS A TIMEOUT
        JC      ERREPORT        ;IF TIMEOUT, PRINT A
        IN      STATU   ;GET STATU
        ANI     2ØH     ;IS FAULT ON?
        MVI     A,'B'
        JNZ     ERREPORT        ;PRINT B IF WRITE FAULT
;REPORT END STATUS
        RET             ;DONE WITH WRITE
```

6.2 Read Routine

The read routine has been divided into two sections - READHST and READINT. READINT reads the data into the sector buffer. READHST calls READINT and transfers the data to the host. This separation is done to allow the Mapping code to call readint allowing it to read data into the buffer but avoiding a host transfer. Errors are handled by the next section, Read Recovery.

```
READHST:                ;READ SUBROUTINE.  READS VIA READINT THEN DOES DMA
         CALL   READINT ;PLACE DATA INTO THE SECTOR BUFFER
         IF     DMA     ;******************************
         MVI    A,04H   ;DMA READ COMMAND
         OUT    CMND    ;DO THE READ
         ENDIF
         IF     NOT DMA ;******************************
         LXI    H,HSTBUF;IO READ LOOP
         LXI    D,HSTSIZ
NXIOR    IN     SCTBU
         MOV    M,A
         INX    H
         DCR    E
         JNZ    NXIOR
         DCR    D
         JNZ    NXIOR
         ENDIF          ;******************************
         RET            ;RETURN
READINT:                ;READ TO SECTOR BUFFER
         CALL   KCOMMON ;SET SECTOR,DMA,TRACK,HEAD,AND UNIT
         RC             ;ERROR RETURN
         XRA    A       ;GENERATE A ZERO
         OUT    BUSL    ;CLEAN UP LOWER BUS
         OUT    BUSU    ;CLEAN UP UPPER BUS
         STA    CERRC   ;ZERO THE ERROR COUNT
RDKON    OUT    STATU   ;SET BUFFER TO ZERO
         MVI    A,88H   ;DISK READ COMMAND
         OUT    CMND    ;DO THE READ
         CALL   WT      ;WAIT FOR DONE
         MVI    A,'C'
         CC     ERREPORT        ;PRINT C IF TIMEOUT ERROR
         JC     RCOVRE  ;GO ATTEMPT TO RECOVER THE TIMEOUT
         IN     STATU   ;GET ENDING STATUS
         ANI    4H      ;IS CHECKSUM ON?
         JNZ    CHKRECOV        ;GO REPORT/RECOVER
;               WE WILL NOW TEST THE HEADER
         LXI    H,PTRK  ;POINT TO THE TRACK
         MVI    B,4     ;NUMBER OF HEADER BYTES
         OUT    STATU   ;ZERO BUFFER ADDRESS
HDRTST   IN     SCTBU   ;GET HEADER READ
         CMP    M
         JNZ    FRMTRCV ;RECOVER IF DIFFERENT
         DCR    B
```

```
        INX     H
        JNZ     HDRTST  ;IF HEADER COUNT IS NON ZERO TEST ONE MORE
;       WHEN WE ARIVE HERE THE READ IS DONE.
        RET
```

6.3 Read Recovery

The data recovery program follows. It follows the flow described in Figure 3.15, Data Recovery Procedures.

```
;
;       READ RECOVERY PROGRAM FOLLOWS
CHKRECOV:
        MVI     A,'D'
        CALL    ERREPORT        ;PRINT A D IF CHECKSUM ERROR
        JMP     RCOVRE
FRMTRCV MVI     A,'E'
        CALL    ERREPORT        ;PRINT A E IF FORMAT ERROR
RCOVRE: LDA     CERRC
        INR     A       ;INCREMENT THE COUNT
        STA     CERRC   ;SAVE THE NEW COUNT
        MOV     C,A     ;SAVE THE ERROR COUNT
        ANI     3       ;GET LAST 2 BITS,STROBE EARLY AND LATE
        CPI     3       ;NOT LEGAL TO HAVE BOTH ON
        JZ      RCOVRE  ;IF ILLEGAL, GO TO NEXT CERRC
        MOV     A,C     ;GET THE ERROR COUNT
        ANI     18H     ;GET THE OFFSET MINUS AND PLUS BITS
        CPI     18H     ;ARE THE BOTH ON?
        JZ      RCOVRE  ;IF SO GO TO THE NEXT COUNT
        MOV     A,C     ;RESTORE THE ERROR COUNT
        CPI     20H     ;IS THIS THE START OF THE SECOND PASS?
        CZ      PANIC   ;IF 20H WE WILL RETRY SEL,SEEK,MAPPER ETC.
        LDA     CERRC   ;RESTORE THE ERROR COUNT
        MOV     C,A     ;PLACE IT IN C FOR FUTURE RECALL
        ADI     0DCH    ;TIME TO GIVE UP?
        JC      HARDRDER        ;IF CARRY WE HAVE A HARD READ ERROR
;       IF WE GET HERE WE ARE READY TO SET RECOVERY BITS AND TRY AGAIN
        MOV     A,C     ;RESTORE ERROR COUNT
        RRC             ;POSITION BITS
        ANI     8CH     ;GET OFFSETS PLUS STROBE EARLY
        OUT     BUSL    ;SET THE BITS
        MOV     A,C     ;RESTORE ERROR COUNT
        RRC             ;POSITION BITS
        ANI     1       ;STROBE LATE BIT
        OUT     BUSU    ;BUS NOW COMPLETE
        JMP     RDKON   ;GO DO THE READ RECOVERY
HARDRDER:               ;HARD DISK READ ERROR
        IN      STATU   ;SET THE READ BUFFER TO ADDRESS 4
        IN      SCTBU
        IN      SCTBU
        IN      SCTBU
        MVI     A,'F'   ;HARD ERROR
        JMP     ERREPORT
```

6.4 Disk Address Selection

The KCOMMON routine is used by both the read and the write to select sector unit, head and track. It also sets the DMA address, useful only for those using DMA and an SMC-100.

The HSTPHY routine transforms CP/M addresses into hard disk addresses. This routine is CP/M unique and probably of no interest to non-CP/M users.

COMCON determines if head, track, or unit have changed. If any have changed, all are reselected by the NEWADD: routine. Note that 15 is added to head address for CMD drives.

```
;
;       KONAN DISK READ AND WRITE COMMON ROUTINE
;       SETS THE SECTOR, DMA, UNIT, HEAD, AND TRACK
;       ADDRESSES.  NOTE THAT EVERY HEAD SELECT IS
;       FOLLOWED BY A SEEK TO MAINTAIN COMPATIBILITY
;       WITH CMD DRIVES.
;       RETURNS WITH CARRY SET IF ERROR, RESET IF NOT
;
KCOMMON:
        XRA     A       ;MAKE A ZERO
        STA     DIOST   ;SET ERROR FLAG TO ZERO
        LXI     H,HSTBUF        ;PLACE DMA ADDRESS IN H&L
        MOV     A,L     ;GET LSB OF DMA ADDRESS
        OUT     DMACL   ;SET LSB OF DMA ADDRESS
        MOV     A,H     ;GET MSB OF DMA ADDRESS
        OUT     DMACU   ;SET MSB OF DMA ADDRESS
        XRA     A       ;MAKE A ZERO
        OUT     BUSL    ;CLEAR THE LOWER BUS
        OUT     BUSU    ;CLEAR THE UPPER BUS
        CALL    HSTPHY
        JMP     COMCON
HSTPHY  ;CALCULATE THE PHYSICAL ADDRESS
        LHLD    HSTTRK
        LXI     D,-402  ;402 IS MAX TRACK PER UNIT
        DAD     D       ;IF CARRY TRACK IS ILLEGAL
        MVI     A,'N'   ;ERROR N IS ILLEGAL TRACK
        CC      ERREPORT
        RC
        LXI     D,402   ;OFFSET FOR INNER TRACKS
        LHLD    HSTTRK  ;CPM TRACK
        LDA     HSTDSK  ;0 FOR A, 1 FOR B, ETC
        IF      FLP1ST
        SUI     NUMDSK-NUMHRD
        ENDIF
        RRC             ;PLACE LSB IN CARRY
        JNC     INNER   ;DO NOT ADD 402 IF INNER
        DAD     D       ;ADD 402 IF OUTER
INNER   ANI     0FH     ;REMOVE END AROUND CARRY
        MVI     B,0     ;B=UNIT, INITIALLY 0
NUNIT   CPI     HEADQ   ;IS HEADQ LARGER THEN HSTDSK?
        JC      UNDONE  ;B IS UNIT, A IS HEAD
        SBI     HEADQ   ;DECREMENT BY THE NUMBER OF HEADS PER UNIT
        INR     B       ;INCREMENT THE UNIT
```

```
UNDONE  STA     PHEAD   ;REMAINDER IS THE PHYSICAL HEAD
        MOV     A,B
        STA     PUNIT   ;UNIT IS THE QUOTIENT
        MOV     A,H     ;MSB OF TRACK
        STA     PTRK    ;STORE IT IN PHYSICAL UPPER TRACK
        MOV     A,L     ;LSB OF TRACK;
        STA     PTRK+1  ;STORE IT IN PHYSICAL LOWER TRACK
        LDA     HSTSEC
        OUT     SCTAD
        STA     PSECTOR
        RET
;       PHYSICAL ADDRESS IS NOW SET
;
;WE WILL NOW CHECK TO SEE IF SELECT IS TRUE, AND IF THE TRACK
;, HEAD, AND UNIT ADDRESSES ARE THE SAME AS THE LAST TRANSFER,
;STORED IN OLDUNIT, OLDTRK, AND OLDHEAD.
;
COMCON
        IN      STATU   ;GET THE STATUS
        RRC             ;PLACE SELECTED BIT INTO CARRY
        JNC     NEWADD  ;IF NOT SELECT FORCE NEW ADDRESS
;       TEST NEW VS OLD ADDRESS
        LXI     H,OLDUNIT       ;H,L POINT TO OLD ADDRESS
        LXI     D,PUNIT ;D&E POINT TO NEW ADDRESS
        MVI     B,4     ;NUMBER OF BYTES TO CHECK
ADCMP   LDAX    D       ;GET NEW ADDRESS BYTE
        CMP     M       ;COMPARE IT TO OLD
        JNZ     NEWADD  ;IF DIFERENT GO TO NEW ADDRESS
        DCR     B
        INX     H       ;INCREMENT THE POINTER TO CURRENT ADDRESS
        INX     D       ;INCREMENT THE POINTER TO DESIRED ADDRESS
        JNZ     ADCMP   ;IF LOOP IS NOT DONE COMPARE ONE MORE
;       RETURN WITHOUT ERROR, SAME ADDRESS
        XRA     A       ;CLEAR CARRY
        RET
NEWADD:
;       SET OLD TO PHYSICAL
        LXI     D,OLDUNIT
        LXI     H,PUNIT
        MVI     B,4     ;COUNT
UPDATE  MOV     A,M     ;GET NEW
        STAX    D       ;UPDATE OLD
        INX     D
        INX     H
        DCR     B
        JNZ     UPDATE
        LDA     PUNIT   ;GET THE PHYSICAL UNIT
        RRC             ;UNIT IS UPPER HALF OF BYTE
        RRC
        RRC
        RRC
        OUT     BUSU    ;PLACE IT ON THE BUS
        MVI     A,10H   ;SELECT COMMAND
        OUT     CMND    ;SELECT I
```

```
        IN      STATU   ;GET STATUS
        RRC             ;PLACE SELECT IN CARRY
        CMC             ;CARRY 1 IF NOT SELECTED
        MVI     A,'G'   ;SELECT ERROR
        JC      ERREPORT        ;PRINT IT
        LDA     PHEAD   ;
        IF      CMD
        CPI     Ø
        JZ      HDJUST  ;IS HEAD ZERO LEAVE ALLONE
        ADI     15      ;ADD 15 TO SET LOLUME TAG
        ENDIF
HDJUST  OUT     BUSL    ;SET IT ON THE BUS
        MVI     A,4ØH   ;SET HEAD COMMAND
        OUT     CMND    ;DO THE HEAD SELECT
        IN      STATU   ;GET THE STATUS
        ANI     2ØH     ;TEST DISK FAULT BUT
        MVI     A,'H'
        JNZ     ERREPORT        ;IF FAULT IS ONE, REPORT IT
```

6.5 Seek Routine

The seek example contains a complex mapping facility and retries. Most of the code is conditional and is only used if mapping is enabled.

```
;       SEEK ROUTINE
;       TRACK SUBSTITUTION INCLUDED
;
SEEK:   CALL    GSEEK   ;GO PERFORM THE SEEK
        RNC             ;RETURN IF SEEK WAS GOOD(NO CARRY)
;       IF HERE, FIRST SEEK ATTEMPT FAILED
        CALL    RECAL   ;RECAL DRIVE TO ATTEMMPT TO CLEAR ERROR
        CALL    GSEEK   ;GIVE IT ONE MORE TRY
        RNC             ;IF NO ERROR IT RECOVERED
;       IF HERE, HARD SEEK ERROR OCCURED
        MVI     A,0FFH  ;SET A TO ALL ONES
        STA     OLDUNIT ;THIS WILL CAUSE A SELECT AND SEEK UPON
;                       NEXT ACCESS
        MVI     A,'I'
        JMP     ERREPORT        ;REPORT THE SEEK ERROR
GSEEK:
;       COMPRIZE COMMPOSITE ADDRESS TO SEARCH TRACK TABLE FOR
        LDA     PTRK
        MOV     D,A
        LDA     PTRK+1
        MOV     E,A
        IF      MAPEN   ;MAP THE TRACK IF MAP IS ENABLED
        LDA     PHEAD   ;GET THE HEAD NUMBER
        RLC
        RLC             ;PLACE HEAD IN BITS 5-2
        ANI     1CH     ;STRIP ANY ENTRA JUNK
        ORA     D       ;OR IN MSB OF TRACK
        MOV     D,A
        LDA     PUNIT   ;GET UNIT NUMBER
        RRC
        RRC
        RRC             ;UNIT NOW IN 6-4
        ANI     60H
        ORA     D       ;OR HEAD TO UNIT/TRACK
        MOV     D,A     ;D IS NOW HEAD,UNIT,TRACKMSB
        MVI     C,0     ;SET B,C,TABLE ENTRY PIONTER, TO ZERO
FINDT   LXI     H,SUBTBL        ;H&L NOW POINT TO THE TABLE
FINDN   MOV     A,M     ;GET BYTE FORM THE TABLE, FIRST HALF
        ADI     80H     ;ADD MSB TO SEE IF IT WAS ON
        JC      SUBDONE ;IF CARRY THE SUBSTITUTION IS DONE
        MOV     A,D     ;LSB OF DESIRED
        CMP     M       ;IS IT THE SAME?
        INX     H       ;INCREMENT THE POINTER
        JNZ     NOCMP   ;JMP NO COMPARE IF DIFFERENT
        MOV     A,E     ;MSB OF DESIRED
        CMP     M       ;IS IT THE SAME
```

```
        JZ      SUBT     ;IF ZERO GO DO THE SUBSTITUTION
NOCMP   INX     H        ;INCREMENT THE POINTER
        INR     C        ;INCREMENT THE LOCATION IN THE TABLE
        MOV     A,C      ;GET TABLE POSITION
        CPI     MAXMAP+1         ;CAUSE CARRY IF MAXMAP OR LESS
        CMC              ;INVERT CARRY,
        MVI     A,'M'    ;M IS MAP OVERRUN ERROR
        CC      ERREPORT         ;PRINT M IF CARRY
        JMP     FINDN    ;GO TEST THE NEXT ENTRY
SUBT:   MVI     B,Ø      ;B,C=THE TABLE ENTRY WHERE THE MATCH OCCURED
        LXI     H,MAXTRK+1       ;THE BEGIN OF THE SUBSTITUTION TRACKS
        DAD     B        ;H&L NOW EQUAL THE NEW TRACK
        XCHG             ;D&E NOW EQUAL THE NEW TRACK
        JMP     FINDT    ;GO SEE IF THE SPARE IS MAPPED
        ENDIF
;
;SUBSTITUTION IS DONE, DO THE SEEK TO THE TRACK IN D&E
SUBDONE
        MOV     A,E      ;LSB OF TRACK
        OUT     BUSL
        MOV     A,D      ;MSB OF TRACK
        ANI     3        ;STRIP ALL BUT TRACK
        OUT     BUSU     ;BUS NOW SET TO TRACK
        MVI     A,2ØH    ;SEEK COMMAND
        OUT     CMND     ;DO THE SEEK
        CALL    WT       ;WAIT FOR DONE
        MVI     A,'J'
        JC      ERREPORT         ;PRINT J IF SEEK TIMEOUT
        RET
```

6.6 Mapper Routine

The mapper routine builds a list of all bad tracks on drives present. This is used by the Seek program to substitute spare tracks for bad tracks. This routine is usually called at boot time. If this routine was not called, if a new drive is now poWered up, or if a pack has been changed, the 6 Seek program will automatically call this program and rebuild the list.

```
;       MAPPER ROUTINE
;       GETS THE MAPS OFF EACH DISK AND MAKES A SYSTEM WIDE
;       MAP OF ALL BAD TRACKS
MAXMAP  EQU     65      ;20 ENTRIES ALLOWED
MAXUNIT EQU     1       ;NUMBER OF DISK UNITS
        IF      MAPEN   ;ASSEMBLE ONLY IF MAP IS ENABLED
SMAPPER
;       SET PHYSICAL ADDRESS TO FIRST
        XRA     A       ;MAKE A ZERO
        STA     HSTTRK
        STA     HSTTRK+1
        STA     HSTDSK
        LXI     H,SUBTBL
        SHLD    SUBPNT
        MVI     A,HSTSPT-1      ;LAST SECTOR NUMBER
        STA     HSTSEC  ;SET IT FOR KCOMMON
;       CLEAR THE MAP
        MVI     A,0E5H
        MVI     B,2*MAXMAP+1
CLRMAP  MOV     M,A
        INX     H
        DCR     B
        JNZ     CLRMAP
GETMAP          ;SEE IF SURFACE IS PRESENT
        CALL    HSTPHY
        LDA     PUNIT
        RLC
        RLC
        RLC
        RLC
        OUT     BUSU
        MVI     A,10H   ;SELECT UNIT COMMAND
        OUT     CMND
        IN      STATU   ;CHECK UNIT STATUS
        ANI     1
        JZ      NEXTMAP ;IF NOT SELECTED GO TO NEXTMAP
        LDA     PHEAD
        IF      CMD
        CPI     0
        JZ      CARTH   ;IF CARTRIDGE THE HEAD ADDRESS IS GOOD
        ADI     15      ;OTHERWISE SET VOLUMN BIT
        ENDIF
CARTH   OUT     BUSL
```

```
        MVI     A,40H
        OUT     CMND     ;SELECT HEAD
        IN      STATU
        ANI     20H
        JNZ     NEXTMAP
        CALL    READINT  ;READ WITHOUT DMA
        JC      NEXTMAP
RDMAPBUF:
        LHLD    SUBPNT   ;SET H&LTO MAP ADDRESS
NBADTRK IN      SCTBU    ;GET FIRST BYTW
        CPI     0E5H     ;IS IT THE END
        JZ      NEXTMAP  ;IF END DO THE NEXT SURFACE
        MOV     B,A      ;PUT BYTE IN B (TRACK MSB)
        LDA     PUNIT    ;GET THE UNIT NUMBER
        RRC              ;POISITION UNIT BITS
        RRC
        RRC
        ANI     60H      ;GET RID OF ANY OTHER JUNK
        ORA     B
        MOV     B,A      ;PUT TRACK AND UNIT IN B
        LDA     PHEAD    ;GET HEAD
        RLC              ;POISITION BITS
        RLC
        ANI     1CH
        ORA     B        ;OR WITH TRACK AND UNIT
        MOV     M,A      ;PLACE INTO TABLE
        INX     H        ;POINT TO NEXT BYTE IN TABLE
        IN      SCTBU    ;GET SECOND BYTE OF TABLE
        MOV     M,A      ;ENTRY IS COMPLETE
        INX     H        ;POINT TO NEXT
        MVI     M,0E5H   ;SET END OF TABLE
;       TEST FOR FULL
        LXI     D,SUBTBL+2*MAXMAP
        MOV     A,H
        CMP     D
        JNZ     NBADTRK
        MOV     A,L
        CMP     E
        JNZ     NBADTRK
        MVI     A,'M'
        CALL    ERREPORT
        JMP     NBADTRK  ;GO GET ADDRESS OF NEXT BAD TRACK
NEXTMAP:

        SHLD    SUBPNT
        LDA     HSTDSK
        INR     A
        INR     A
        STA     HSTDSK
        CPI     2*HEADQ  ;ARE WE DONE ?
        RZ      ;IF SO, RETURN
```

```
        JMP     GETMAP
        ENDIF
;       IF HERE WE ARE DONE!
;       DO NOT CHANGE THE ORDER OF THE NEXT 9 BYTES
CERRC   DB      Ø       ;ERROR COUNTER
PUNIT   DS      1       ;DESIRED PHYSICAL UNIT
PTRK    DS      2       ;DESIRED REAL TRACK ADDRESS
PHEAD   DS      1       ;DESIRED HEAD
PSECTOR DS      1       ;DESIRED SECTOR,512 BYTE
OLDUNIT DB      ØFFH    ;LAST HEAD SELECTED, FF TO CAUSE INITIAL SEL.
OLDTRK  DW      ØFFFFH  ;LAST TRACK SEEKED
OLDHEAD DB      ØFFH    ;LAST HEAD ACCESSED
```

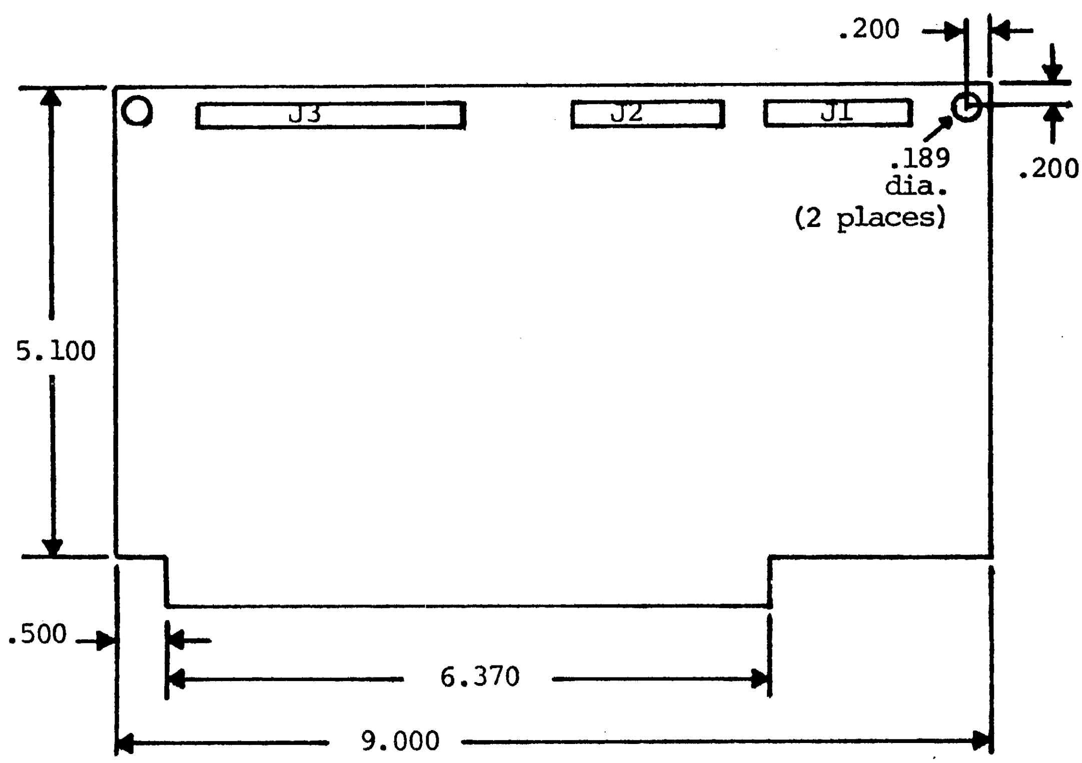
SML-2ØØ INTERFACE SUPPLEMENT

I. Intent

The intent of this manual is to define the interface of the SML-2ØØ in sufficient detail to allow its use as a stand alone controller. This manual must be used in conjunction with the SMC O.E.M. Reference Manual, R-SMC-2Ø1-X.

II. Mechanical Interface

Figure 2.1 shows the physical outline of the SML-2ØØ PCB.



The edge connector is a 1ØØ pin (dual 5Ø) with .125 spacing.

J1 and J2 are dual .1ØØ spacing, .Ø25 square gold posts with 26 pins.

J3 is a 6Ø pin dual .1ØØ spacing, .Ø25 square gold posts

III. Signal Description:

This section is divided into two parts. The first section covers mandatory signals which must be used to interface the SMC-200. Additional signals are covered in the next section, optional signal lines. It is intended that these be used only when it simplifies the users interfacing task to do so.

Note that a separate data out and data in buss are provided. These can and usually are, tied together, forming a bi-directional data buss.

Mandatory Signals

Pin #	Name	Description
1	+5V	regulated +5V
35	DO1	
36	DO0	
38	DO4	
39	DO5	
40	DO6	
41	DI2	
42	DI3	
43	DI7	
50	GND	GND
51	+5V	+5V, regulated
52	-5V	-5V, regulated
53	GND	GND
77	PWR*	Write Strobe
78	PDBIN	Read Strobe
79	A0	Selects one of
80	A1	seven on board
81	A2	registers
88	DO2	
89	DO3	
90	DO7	
91	DI4	
92	DI5	
93	DI6	
94	DI7	
95	DI0	
100	GND	

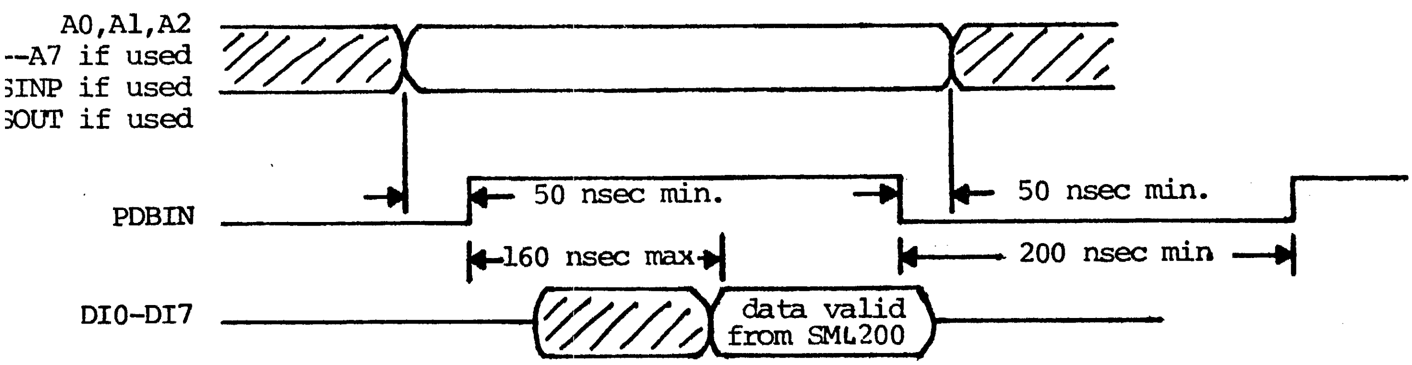
Optional Signals

Pin #	Name	Description
13	PWRFAIL*	disables drive on power down
29	A5	board select address
3Ø	A4	board select address
31	A3	board select address
45	SOUT	can be used to qualify PWR
46	SINP	can be usd to qualify PDBIN
75	RESET*	clears controller
82	A6	board select address
83	A7	board select address

When not in use, GND pins 29,3Ø,31,82,83
, pull up 45,46,13,75

IV. Timing

READ CYCLE



WRITE CYCLE