DIGITAL SYSTEMS <FDC3>

NOTE:

The S100 interface cards versions 1.1,1.2,1.3,and 1.4 induce some complements not present in the normal FDC-3 interface.

Please keep this fact in mind when referring to the following manual.

Address port	S100 address port (after complement)
80H = 84	7FH 1st COMM (Status)
81H = 85	7EH Address HI
82H = 86	7DH Address LOW
83H = 87	7CH 2nd COMM

Also, the accumulator should be complemented before outputting over the command or address ports

i.e.

```
MVI     A,40H    ;NORMAL READ COMMAND
CMA              ;FOR S100
OUT     7FH      ;FOR S100
```

Note 1: (Pre-Changes) Since (A15-A8) = (A7-A0) assumed on IO Instructions, and Since A10 is not decoded by S100 interface card, IO ports (80-83H) = IO ports (84-87
thus 8 Ports are used by S100/FDC disk interface (for S100, they are (78-7F)H)

MS Byte LS Byte

Note 2: (Pre-Changes) Since FDC card does decoding of INPUT from Ports 81H(or 85H) to Strobe its J1-44; thus, on S100 interface card this causes $\overline{\text{pRESET}}$ to strobe
(In other words if an input is done from PORT 81H (or 85H) a processor reset is do
[also true for 82H (+86H) - ?]

1.0.0 Operation of the Disk Controller and the Disks

The FDC-3 disk controller board performs the following functions:

1) Electrical interface with the disk drive.
2) Parallel to serial and serial to parallel data conversion.
3) Generation and check of CRC error codes.
4) Identification of requested tracks and sectors.
5) DMA data transfers of the 128 byte sectors.

A TTL implemented micro-controller on the controller board sequences the on-board logic to perform the above functions. All sectors, single and double density, are 128 bytes long. This length is determined by a variable in the micro-controller's microcode program ROM. Custom microcode ROMs for special applications are possible.

A disk drive must be in the READY state before any data transfer operations can take place. To be READY, a drive must:

1) Have a diskette in it.
2) Have the door closed.
3) Have been selected.
4) Have the head loaded.

Additionally, mechanical action of the drive (track to track movement, head loading, etc) all require delays after the READY indicator is turned on (bit 7 of the drive's status port) before reading from or writing to the disk can be done. Each track stepped requires an 8 millisecond delay,(3 ms for Shugart 850) plus an additional 10 millisecond settling time after the last track stepped. (Stepping Delay) = ((8 × #TRK) + 10) in milliseconds

The FDC-3 can handle up to 8 drives although it connects to the control lines of one disk at a time. The disk drive to which the controller is connected is said to be the selected disk.

1.1.0 Bootstrapping

The FDC-3 disk controller board starts the host computer's operating system automatically. This process is called Bootstrapping. The bootstrapping operation is done by logic built into the controller and occurs whenever the system power switch is turned to ON or when the system is RESET . The controller transfers the contents of track 0 sector 1 of drive 0 to memory locations 0-7FH. The processor is allowed to begin execution at location 0 when the transfer is complete.
(See Note 2 for additional RESET generation)
During the bootstrap, the processor is held in a RESET state. Consequently, memory refresh cycles do not occur and the content of memory is indeterminate.

The FDC-3 controller always selects drive 0 for the bootstrap operation. Should drive 0 be unuseable due to mechanical problems, you can boot from drive 1 by changing the drive select jumper described in section 1.4.1

The data that is loaded from track 0 sector 1 of a CP/M disk is a loader program that proceeds to load in the rest of the CP/M operating system from the disk and then execute CP/M when the loading process is complete. CP/M, while executing, uses the memory locations occupied by the loader program for various tables and system variables and overlays the program with this data. Pushing the RESET button causes the loader program to be written into these locations, which destroys CP/M's record of what was going on at the time the RESET button was pushed.

Should you desire to write your own loader program, or to modify the loader to load some program other than CP/M, all that you have to do is write your own 128 byte program and store it on track 0 sector 1 of the diskette you intend to use as a system disk. (see section 1.3.4 for instructions on how to address a given track and sector directly or make use of the CP/M transient program SYSGEN.)

1.2.0 DMA

Data transfers between the disk controller and memory are performed by a special kind of transfer process called Direct Memory Access (DMA). This means that the controller addresses memory directly, competing with the CPU for access to memory as it reads or writes each byte.

If the diskette is formatted in double density, the transfer will happen every 16 microseconds. A byte is transferred every 32 microseconds for a single density diskette.

<u>Byte Xfer rate</u>
SD = one per 32µS
DD = one per 16µS

1.2.1 Timing considerations for DMA transfers.

In single density mode tranfer of a disk sector (128 bytes) takes 4.096 milliseconds, but in double density mode the same transfer takes only 2.048 milliseconds. Data is recorded on the disk in such a way that there are gaps of about 16 bytes between sectors on double density disks and 32 bytes on single density disks.

This gap between sectors (called the 'inter-record gap') amounts to only 250 microseconds on Double Densisty format disks, which means that programs that try to read physically adjacent sectors on the disk are operating under very strict time constraints.

1.2.2 Interleaving

To avoid this difficulty, software does not try to read adjacent disk sectors. Instead, when the disk is first formatted, the format program writes out sectors with sector numbers in such a way that numerically consecutive sectors are not physically adjacent, but in a given track run as follows: 1,30,2,31,3,33,.... This allows plenty of time (2.3 milliseconds) to do the calculations required to be ready to handle the next sector. The controller itself is fast enough to read or write adjacent sectors, but in general programs are not able to keep up.

To use CP/M efficiently, the data sector numbers are interleaved so that logically consecutive sectors in a file are physically separated by several sectors on the disk. This allocation scheme is designed to give CP/M enough time to process a buffer of data and make the mapping between the next record in the file and its actual physical sector number. This mapping is reflected by the table DOBTAB in the disk format program.

For optimum loading, tracks 0,1 are interleaved every other sector. This is because the bootstrap load programs do very little processing between sectors. The CP/M tracks 2-76 are interleaved every 9 to let CP/M loaded programs execute faster. (for Single density Trks 2-26 are interle
every 6

If you wish to experiment with renumbering sectors by modifying the FORMAT program tables, feel free to do so, but keep the following conditions in mind:

1) Make sure that you have 26 sectors numbered 1-26 for single densisty disks, and 58 sectors numbered 1-58 for DD format disks. If you try to do too few or too many, errors will result.

2) If your program takes more time to set up for the next sector than the inter-record gap time (which will vary according to the number of sectors that are skipped in your interleaving system) it will have to wait one full revolution of the disk to read or write the next sector. This means that if your program is too slow by 1 microsecond, it could take up to 57 disk revolutions to`read a whole track. (that's 9.7 seconds)

CP/M uses the following interleaving: (defined in BIOS)

Logical Sector 1 = physical Sector 1
2 = 7
3 = 13
4 = 19
5 = 25
6 = 5

max = 26
Single Density
or 58 if
double Density

1.3.0 Program communications with the disk controller.

Up to this point we have restricted our discussion of the disk controller to DMA transfers between the disk controller and memory without mentioning communications between the CPU and the disk controller. But no DMA operations other than bootstrap loading will take place unless the controller is commanded to perform them by the CPU. The CPU has to be able to communicate with the disk controller, and this communication takes place through the disk controller address, command and status ports.

1.3.1 The status port (80H) <S100= 7Fh>

The disk controller's status port indicates the current status of the selected disk. The contents of this port can be read into the 'A' register and examined. The table below gives the meanings of the various bits that may be set in the status register.

bit number		meaning
(LSB) 0	DSKWE = 1	0 if diskette is write protected 1 if diskette is not write protected
1	DSKSAME = 1	0 if drive door has been opened since last selected
2	NOTHOME = 1	0 if disk is at track 0
3	IOC = 1	1 if I/O is complete.
4	ERRTRK = 1	1 if a track error has occurred.
5	ERRSCRC = 1	1 if a CRC error occurred in a sector's ID field
6	ERRDCRC = 1	1 if a CRC error occurred in a data field
(MSB) 7	ERRNRDY = 1	0 if selected drive is ready (head loaded) 1 if selected drive is not ready (or in error) (bits 0,1,2 invalid if this bit not 0)

The disk controller status port is numbered 080H. Bits 3-6 indicate errors or states that may have happened at some time in the relatively remote past, and can be reset by giving an output command to this port. To initiate disk activity an OUT 80H should be executed to reset and then an IN 80H should be executed to determine the current status of the drive.

1.3.2 The Command Ports

The disk controller has two ports that are used to pass instructions to it from the CPU. To pass an instruction to the controller, the 'A' register is set to the value of the instruction and an OUT instruction is given with the device address specified as being one of the two command port addresses. (note that reading in information from these ports via the IN instruction is invalid and sets the 'A' register contents to all 1's.)

The first command port is numbered 80H and is used to command the selected disk to move to a certain track. This port is also used to tell the controller which operation is going to be performed at that track -- read, write, or format. See the table below for the details of the meanings of the indiviual bit settings in the byte.

FIRST COMMAND PORT (080H) <S100 = 7Fh>

	BIT NUMBER	MEANING
(LSB)	0	not used
	1	1 Step command (move the head one track) 0 Do Not Step head
	2	1 Head step direction towards track 76 0 Head step direction towards track 0
	3	not used
	4	1 Inhibit disk write precompensation (used when writing on tracks 0-39D) 0 Allow disk write precompenstion
	5	1 format the track 0 don't format the track
	6	1 read requested of track 0 no read requested
(MSB)	7	1 write requested to track 0 no write requested.

The second command port is used to select the disk that is to be read from or written to, and to enable this disk to interrupt the processor the completion of its data tranfer. the Bit assignments are as follows:

SECOND COMMAND PORT (083H) <S100 = 7Ch >

	BIT NUMBER	MEANING
(LSB)	0	disk select bit 0
	1	disk select bit 1
	2	disk select bit 2
	3	if 1 Override the jumper selected density and select the other density.
	4	if 1, interrupt from selected disk is enabled.
	5	not used.
	6	if 1, Format enable
(MSB)	7	not used.

Reset and power-on clear these bits to 0 so that drive 0 s selected for the bootstrap process.

1.3.3 The Address Ports

The controller has two address ports. These ports are set to the address in memory that is the source or destination of the data to be transferred in the DMA transfer.

ADDRESS PORT 81H <S100=7Eh>

High order byte of the DMA address (also pReset "Port" strobe)

ADDRESS PORT 82H <S100= 7D h>

Low order byte of the DMA address

1.3.4 Programmed DMA Transfers

The three bytes in memory immediately before the area that is to be read or written should be filled in as follows:

DMA ADDR points here (for Port 81&82) →
address-3 = track number
address-2 = sector number
address-1= address mark (OB= double density)
(FB=IBM single density)

[Data area follows:
(128d bytes long)

(*) The address of the first of these three bytes is the actual address passed to the disk controller over the two address ports as follows:

```
        LXI     DATA-3
        MOV     A,L             ;MOVE THE LOW BYTE UP TO THE 'A' REG.
        OUT     82H             ; LOW BYTE ADRESS PORT
        MOV     A,H             ; HIGH BYTE OF ADRESS
        OUT     81H             ; HIGH BYTE ADRESS PORT
```

After the address has been given to the controller, the actual read or write command is sent over the first command port:

```
;FOR READ
        MVI     A,40H           ; THESE ARE THE BITS TO SET ON FOR READ
        OUT     80H             ; FIRST COMMAND PORT

;FOR WRITE
        MVI     A,80H
        OUT     80 H
```

If interrupts are enabled on this operation, the program can now go do something else while the DMA transfer takes place. The interrupt routine will check the status port to determine if an error occured during the transfer this way:

```
        IN       80H
        ANI     OFOH    ;ALL THESE BITS WILL BE ZERO IF NO ERROR
        JNZ     ERROR
```

If interrupts are not enabled, the program can loop, waiting for completion by testing the status port during the loop:

```
WAIT    IN      80H
        ANI     08H     ; TEST FINISHED BIT
        JZ      WAIT
```

1.3.5 Direct Sector I/O

For an example of how to use the CBIOS routines to do direct sector I/O, here's how to read sector 3 of track 6 into location 80:

```
; FIRST, SAVE ALL REGISTERS, AS CBIOS ROUTINES WILL NOT RESTORE THEM
; SELECT DRIVE  (ONLY NEEDS TO BE DONE IF YOU CHANGE SELECTED DISKS)
        MVI     C,1
        CALL    SELDSK
;
; SET THE DRIVE TO TRACK 0 SINCE ITS PRESENT TRACK NUMBER IS NOT KNOWN.
;  ONLY NEEDS TO BE DONE THE FIRST TIME DRIVE IS USED
        CALL    HOME     ;THE HOME FUNCTION MOVES THE SELECTED DRIVE
                         ;TO TRACK 0 AND INITIALIZES TRACK LOCATION
                         ;IN SCRATCHPAD MEMORY FOR THE SELECTED DRIVE
;   STEP DRIVE TO TRACK 6
        MVI     C,6
        CALL    SETTRK   ;THIS FUNCTION MOVES THE HEADS & UPDATES SCRATCHPAD
;
; REQUEST SECTOR 3
        MVI     C,3
        CALL    SETSEC   ;SETS UP SCRATCHPAD VARIABLE
;
; SET UP DMA ADDRESS OF 80H
        LXI     B,80H
        CALL    SETDMA   NOTE THAT THIS IS ACTUAL DATA ADDRESS, NOT DATA
                         ADDRESS-3 AS IN PREVIOUS EXAMPLE
;
; DO THE ACTUAL READ OPERATION
        CALL    READ     ;READ USING THE TRACK, SECTOR, AND DMA
                         ;ADDRESSES IN THE SCRATCHPAD
                         ;READ or WRITE SAVES AND RESTORES THE 3 BYTES
                         ;BELOW THE DATA AREA, FILLS IN THE TRACK,SECTOR,AND
                         ;ADDRESS MARK AND SENDS COMMANDS TO THE CONTROLLER.
;
;  TEST FOR A PERMANENT ERROR DETECTED BY READ
        CPI     1        ; READ RETURNS WITH 'A' = 1 IF ERROR IS DETECTED
        JNZ     rest of program
        JMP     error handler
;
; TABLE OF ACCESSES INTO CBIOS
;
SELDSK  LXI     D,1BH
        JMP     CBIOS
HOME:   LXI     D,18H
        JMP     CBIOS
SETTRK: LXI     D,1EH
        JMP     CBIOS
SETSEC: LXI     D,21H
        JMP     CBIOS
SETDMA: LXI     D,24H
        JMP     CBIOS
READ:   LXI     D,2AH
        JMP     CBIOS
CBIOS:  LHLD    1        ;GET POINTER TO CBIOS JUMPTABLE
        MVI     L,0
        DAD     D
        PCHL             ;GO TO CBIOS
```

1.4.0 Physical Description

1.4.1 Connectors

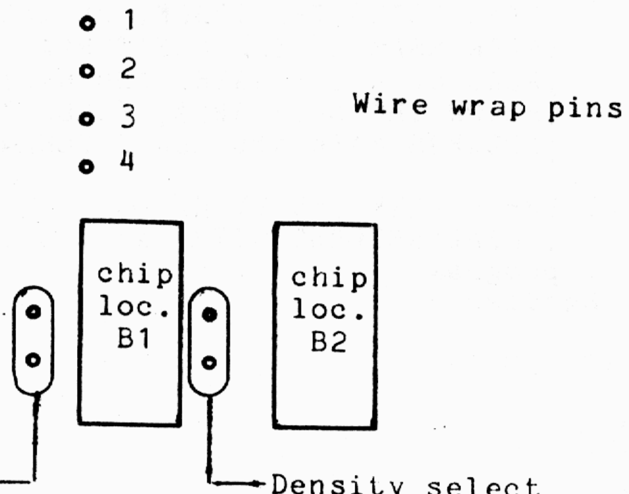
J1	Signal Connector to computer interface
J3	Signal connector to disk drives
J5	DC power connector

1.4.2 Jumper options

4 wire wrap pins near corner of board opposite the crystal.

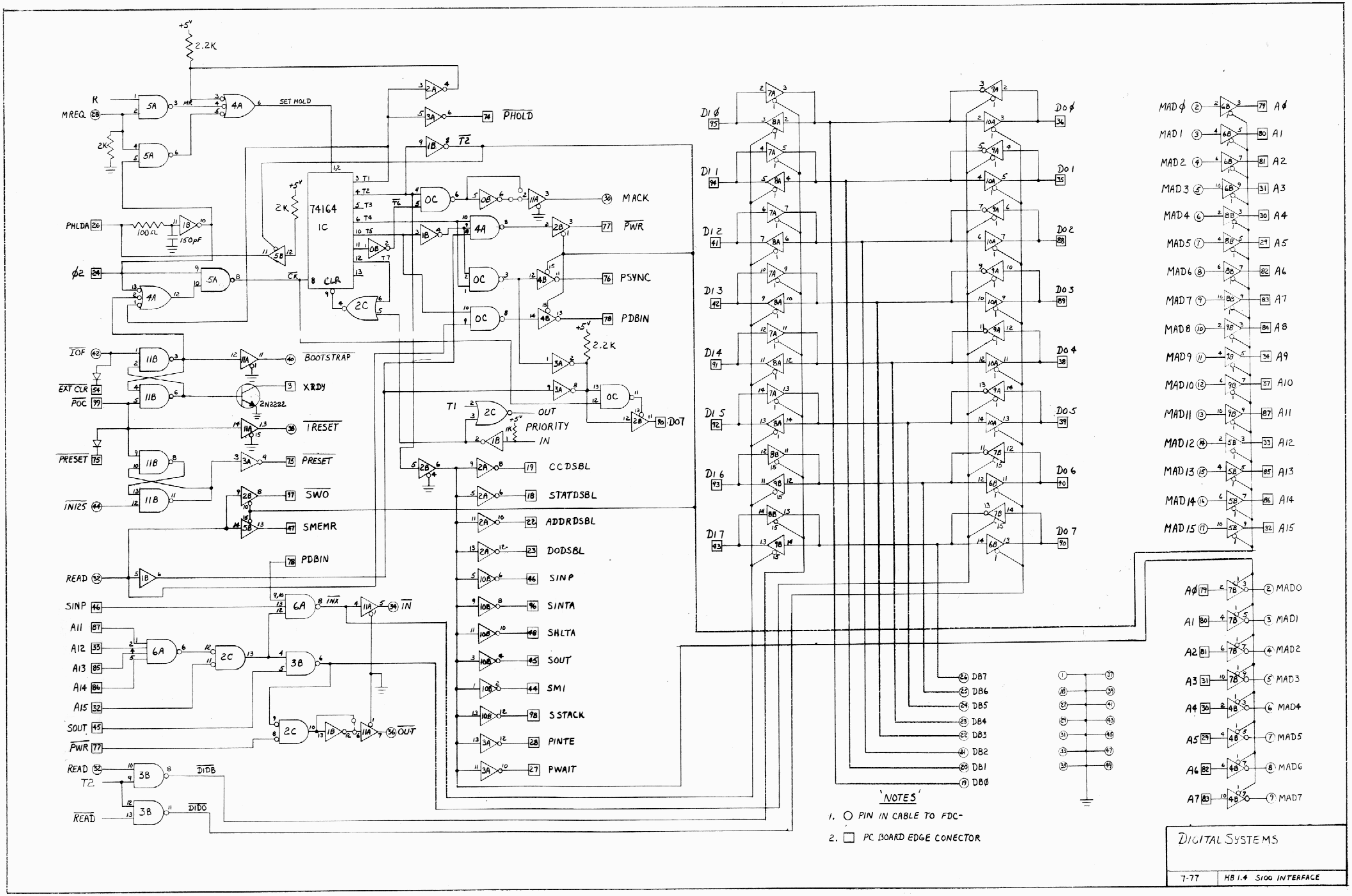
pin 1	ground	(pin closest to corner)
pin 2	write protect	ground to protect normally shipped open
pin 3	density select	ground to select double density normally shipped open
pin 4	drive swap	ground to select right drive as A normally shipped grounded

For convenience, pins 3 and 4 can also be grounded as illustrated below using a shorting plug:



Drive swap
Normally shipped with plug in.
(Presently goes to front panel switch)

Density select
Normally shipped with plug removed
(Presently goes to front panel switch)



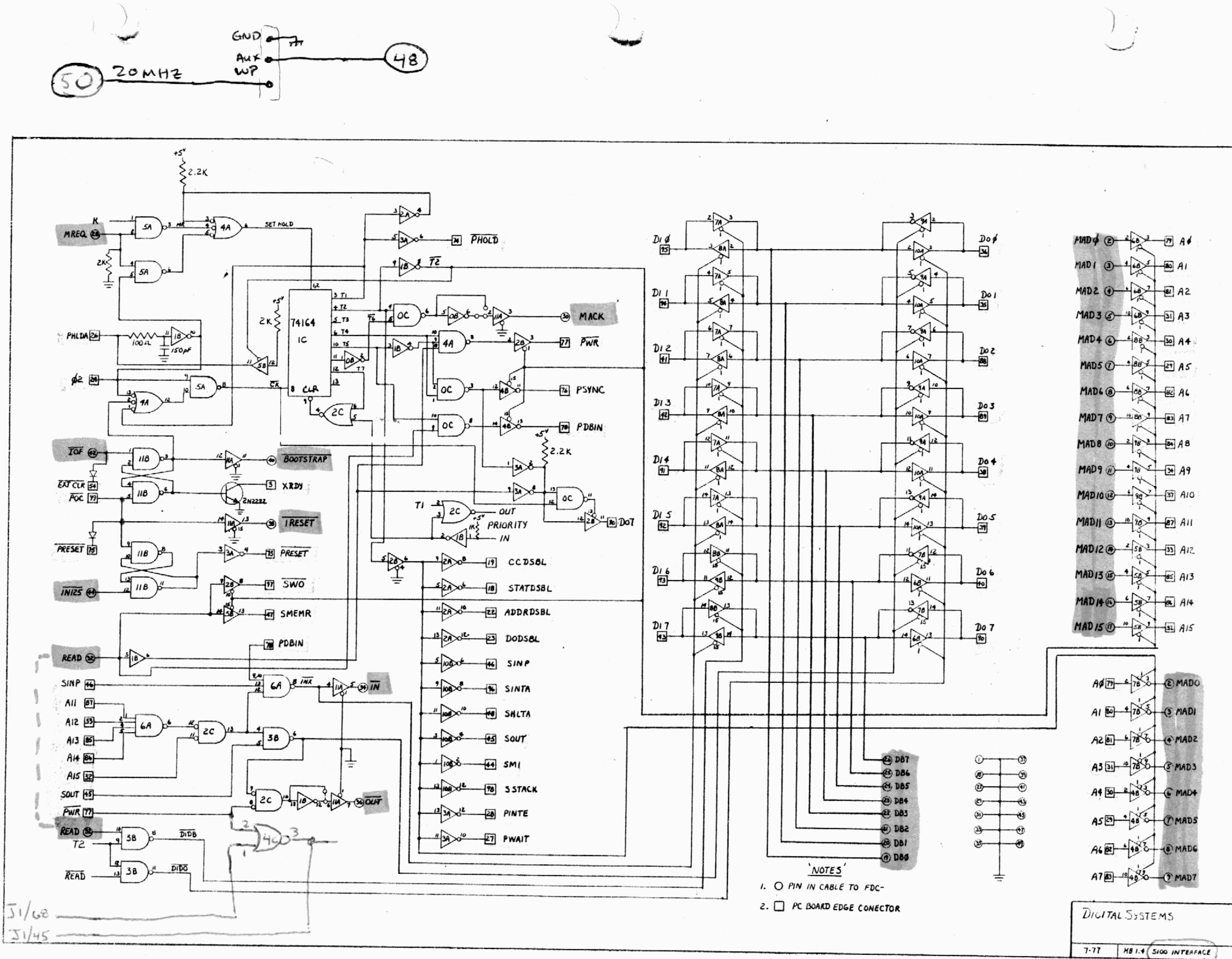
'NOTES'

1. ○ PIN IN CABLE TO FDC-

2. ☐ PC BOARD EDGE CONECTOR

DIGITAL SYSTEMS

7-77 HB 1.4 S100 INTERFACE



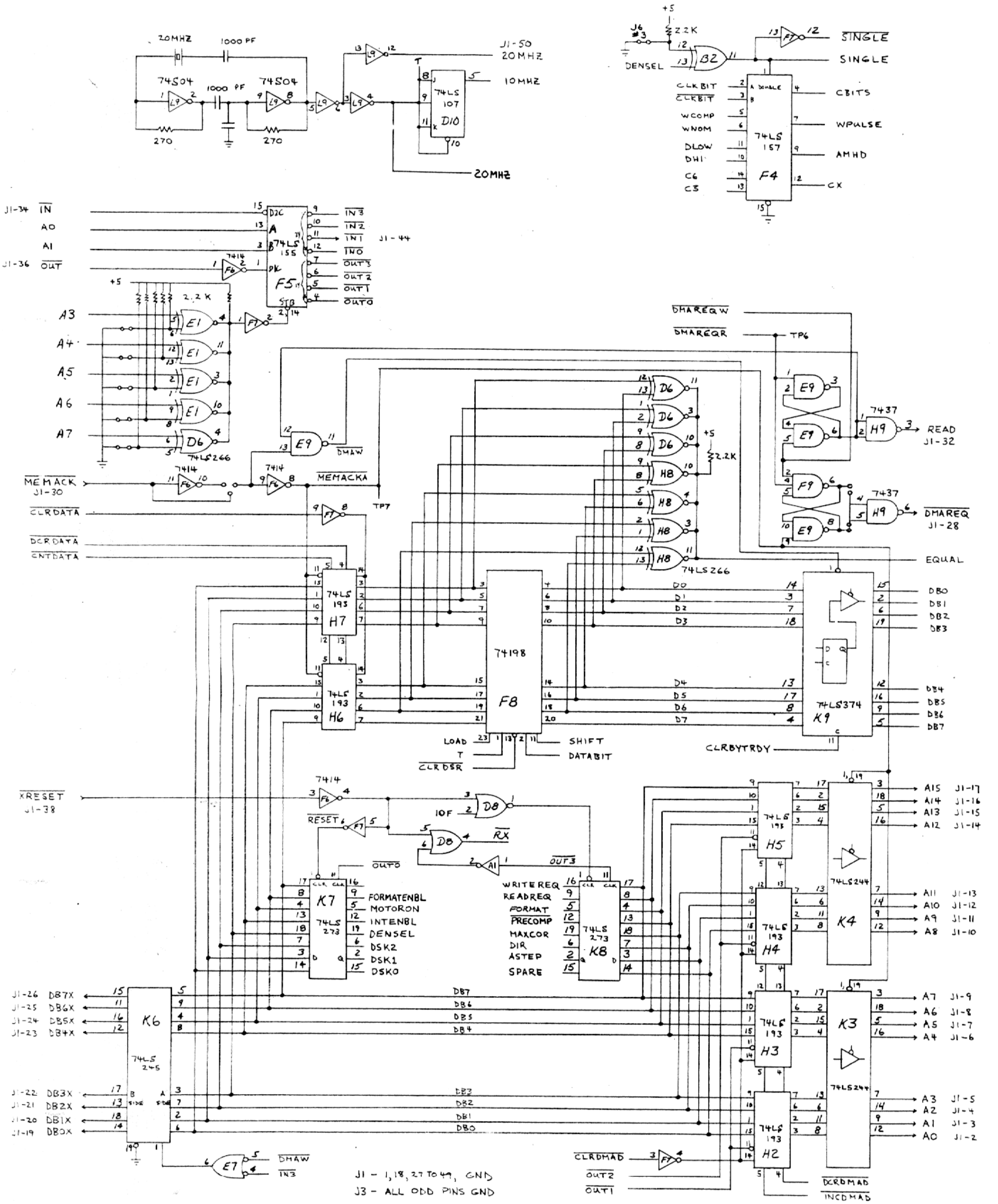
'NOTES'

1. ○ PIN IN CABLE TO FDC-

2. ☐ PC BOARD EDGE CONECTOR

DIGITAL SYSTEMS

7-77 | MB 1.4 | S100 INTERFACE



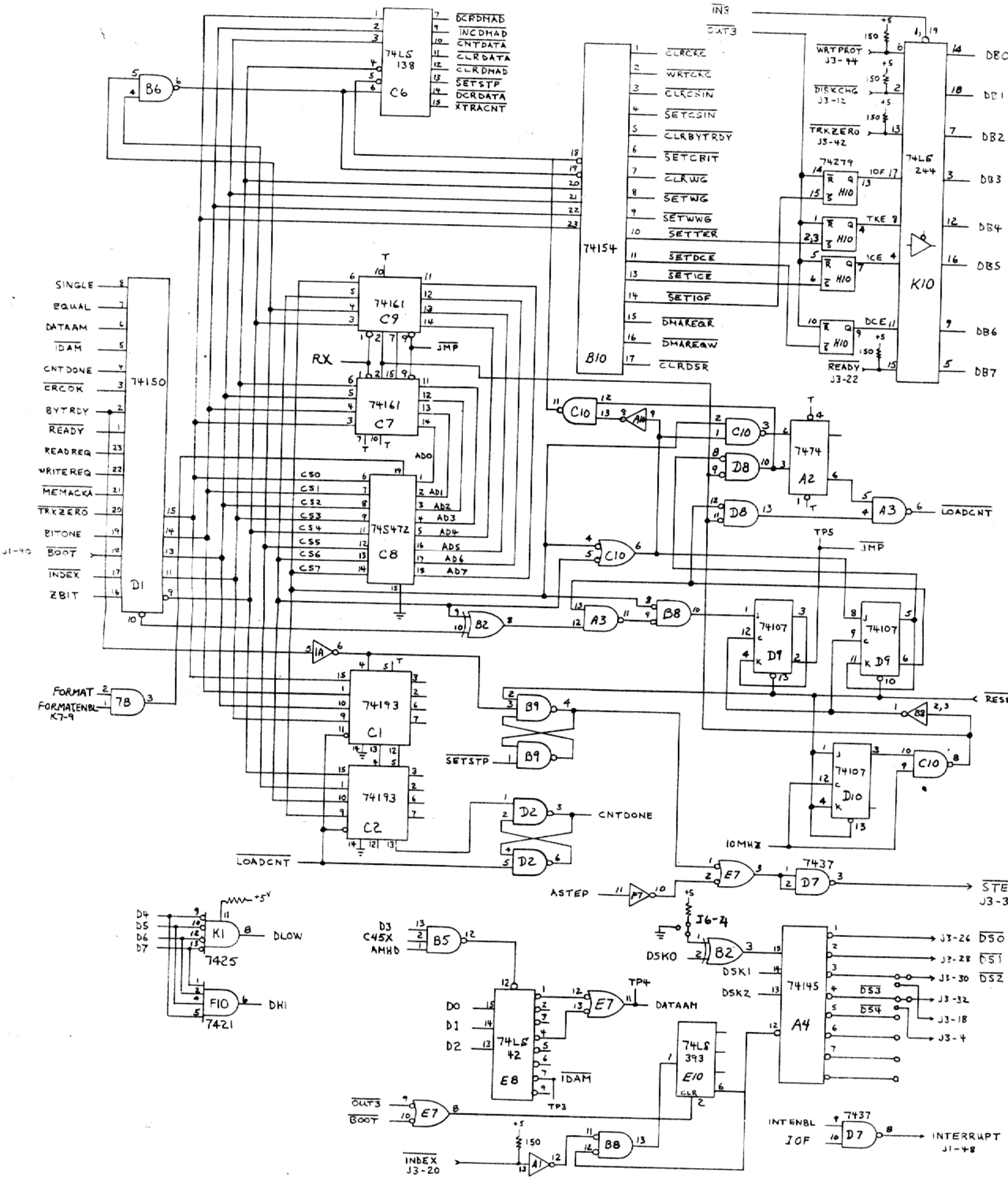
DIGITAL SYSTEMS

FLOPPY DISC

CONTROLLER FDC-3.3

DATE: 1-78

SHEET 1 OF 3

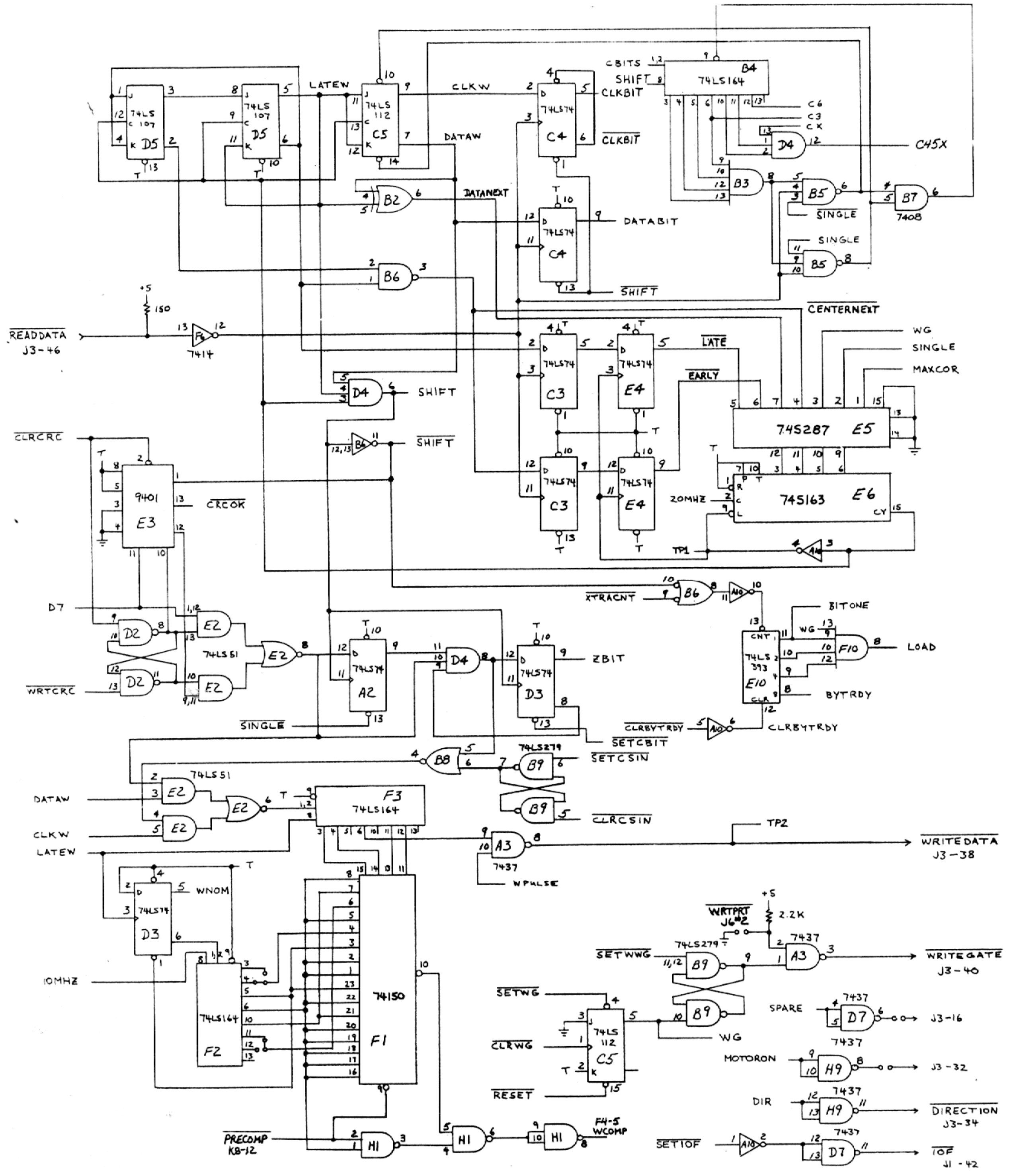


DIGITAL SYSTEMS

FLOPPY DISC
CONTROLLER FDC-3.3

DATE: 1-78

SHEET 2 of 3



DIGITAL SYSTEMS

FLOPPY DISC
CONTROLLER FDC-3.3

DATE: 1-78

SHEET 3 of 3

```
;       BOOTSTRAP LOADER
;BOOT3-S2
;THIS VERSION USES TIGHT CODE THAT GIVES MORE SPACE AT THE END
;FOR UART  INITIALIZATION.
        ORG     00H
;
;
;LOAD THE MONITOR AT 2900
LOADP   EQU     2900H   ;LOAD POINT
BCP     EQU     3E00H   ;MONITOR
PORT    EQU     4AH
LAST    EQU     21      ;LAST SECTOR ON TRK1 TO LOAD
SKIP    EQU     1       ;NUMBER OF SECTORS TO SKIP
;
DMAH    EQU     126     ;HIGH ORDER DMA ADDR        7EH          127 = 7FH
DMAL    EQU     125     ;LOW ORDER DMA ADDR         7DH          126 = 7EH
REBOOT  EQU     125     ;REBOOT SYSTEM ON ERROR     7DH          125 = 7DH
INIT0   EQU     ((SKIP+1) SHL 8) OR 0   ;INITIAL TRACK,SECTOR
END0    EQU     (0 SHL 7) OR 27 ;END OF FIRST TRK
INIT1   EQU     (129 SHL 8) OR 1        ;TRACK,SECT ON TRK 1
END1    EQU     (1 SHL 7) OR LAST+1     ;LAST POSITION ON TRK 1
;
T76     EQU     0FBH    ;TOWARD 76          1111 1011
STT     EQU     0FDH    ;STEP TRACK         1111 1101
DOUT    EQU     7FH     ;DISK OUTPUT PORT
DINP    EQU     7FH     ;DISK INPUT PORT
RDS     EQU     0BFH    ;READ SECTOR
ERR     EQU     0F0H    ;ERROR CONDITIONS
IOF     EQU     08H     ;IO FINISH
;
;
        NOP             ;SOME DYNAMIC MEMORIES PROHIBIT USE OF
                        ;THIS BYTE IN THE BOOTSTRAP
;       INITIALIZE TRK/SEC AND DMA ADDRESS
START:
        LXI     D,INIT0
        LXI     H,LOADP-3
;
READL:  ;READ LOOP
        SPHL            ;SET THE STACK POINTER
;       SEND DMA ADDRESS
        MOV     A,H
        CMA
        OUT     DMAH
        MOV     A,L
        CMA
        OUT     DMAL
;
;       SET UP TRACK AND SECTOR
;
        MOV     L,E     ;TRACK TO L
        MOV     A,D     ;SECTOR TO A
        ANI     7FH     ;CLEAR HIGH BIT
        MOV     H,A     ;SECTOR TO H
        XTHL            ;SET TRK AND SECTOR
        INX     SP      ;ADJUST STACK POINTER
        POP     B
;       PERFORM THE READ
        MVI     A,RDS
        OUT     DOUT
```

```
;
WRD:    IN      DINP
        ANI     ERR
        JZ      NERROR
;
;       ERROR, REBOOT
        IN      125
        HLT
;
;
NERROR: ;NO ERROR SO FAR
        IN      DINP
        ANI     IOF
        JZ      WRD     ;GO BACK AND WAIT
;
;       I/O FINISH,
;       REPLACE DATA BYTES
        PUSH    B       ;REPLACE 1 BYTE
        INX     SP      ;ADJUST STACK POINTER
        PUSH    H       ;REPLACE 2 BYTES
;
;       ADD 128 TO DMA ADDRESS
        LXI     H,128
        DAD     SP
;       INC TRACK/SECTOR
        INR     D
        MOV     A,D
        CPI     END0    ;END OF TRACK 0?
        JNZ     CMP1
;
;       STEP TRACK TO TRACK 1            ?
        MVI     A,T7128
        DAD     SP
;       INC TRACK/SECTOR
        INR     D
        MOV     A,D
        CPI     END0    ;END OF TRACK 0?
        JNZ     CMP1
;
;       STEP TRACK TO TRACK 1
        MVI     A,T7MVI C,82H   ;ONE MILLISEAOND TIMING FOR OUR IMSAI
                  6     ;CAN BE ADJUSTED FOR DIFFERENT MACHINES
WST2:
        DCR     C
        JNZ     WST2
        DCR     A
        JNZ     WST0
;
;               TRACK STEPPED
        LXI     D,INIT1
        MOV     A,D
CMP1:   CPI     END1    ;END OF TRACK 1?
        JNZ     READL   ;GO BACK FOR MORE
;
;       END OF LOAD
;SET LOC 4 TO CONTAIN ACTIVE DRIVE=0
        LXI     H,PORT
        MOV     M,H     ;H IS 0
        INX     H
        MOV     M,H
        INX     H          ;POINT TO DENSITY SELECT
        MVI     M,0        ;SINGLE DENSITY
;SET LOC 4 TO CONTAIN ACTIVE DRIVE = 0
        MVI     L,4
        MOV     M,H
        JMP     BCP        ;GO TO COMMMND PROCESSOR

        ORG     7FH
        DB      0CH        ;FOR SOME DYNAMIC MEMORIES
        END
```