DEMON/II

DISK EXECUTIVE MONITOR

FOR THE NICOLET 294 DISK SYSTEM

Instructions for Use

and

Programming Manual

Nicolet Instrument Corporation
5225 Verona Road
Madison, Wisconsin 53711

October 1973

TABLE OF CONTENTS

I. INTRODUCTION TO THE DISK

Description of Hardware

The Nicolet 294 disk system is a high speed random access device for the storage of programs and spectral data. It can call in programs and swap them out again about as fast as the commands can be typed on the Teletype. Thus it is ideal for building up libraries of spectral data and various application programs. It can also be used to store multiple spectra in experiments where the response changes as a function of time.

The disk itself is a smooth iron oxide coated platter containing magnetic information which is detected by heads riding over both surfaces of the disk. This information is deposited in concentric circles called tracks. Each track consists of 1536 20-bit words and checkword information. The information in these tracks is deposited as a string of bits which is assembled into words by the NIC-294 controller. When the controller finishes reading each track, the sum of these 20-bit words is compared with one additional word called the checkword. If these two do not agree a read error has occurred and a flag is set by the controller indicating this fact.

There are 406_{10} or 626_8 tracks available on the disk for data storage. This means that there are a total of 623,616 words of storage available on one single disk cartridge. Furthermore, since the cartridges themselves are relatively inexpensive, it is not unreasonable that several of them might be obtained and loaded into the disk controller interchangably, thus increasing the available storage area even further.

Installation

The disk operation for the storage and loading of programs is discussed in detail in the following chapters. These sections should be studied before attempting to use the disk system in conjunction with any serious experimental work. However, this section on installation is intended as a "cookbook" for installing the disk and determining that it is working properly. Instructions are the same for installation and use of the disk with a 1080, NMR-80, LAB-80, MED-80 or NIC-80.

1. Unpack the disk cartons. There should be the following units:

   a. A Diablo 31 disk drive,
   b. a Diablo power supply,
   c. a NIC-294 disk controller and power cord,
   d. a disk software package,
   e. a Model 2315 disk cartridge.

2. The following cables should be supplied with the disk controller and drive:

a. A cable with 5 terminal lugs at one end and a small 14-pin connector at the other.
b. One single belt wire terminating in male and female 42-pin screw-on connectors for each disk unit.
c. A triple belted 1080 I/O connector with flat covered plugs on both ends.
d. An AC power cord.
e. A flat 42-pin terminator connector.

3. If the 1080 is running, press both STOP buttons and turn off the power key-switch.

4. Remove the top cover of the disk drive and remove the brass bracket marked "Shipping Restraint." DO NOT attempt to load a disk cartridge or to apply power before removing this restraint.

5. Connect the cable containing the 5 terminal lugs to the power supply. The two red wires should be connected to the +15 volt terminals and the two blue connectors to the -15 volt terminals. The braided wire can be connected to either terminal marked Ground.

6. Connect the other end of the above cable to the 14-pin connector at the rear of the disk drive.

7. Connect the female end of the 42-pin belted wire to the mating connector at the rear of the disk drive. Connect the male end to the rear of the 294 controller box. Connect each subsequent disk unit with the 42-pin belted wire and terminate the last disk unit with the 42-pin terminator connector.

8. Connect one end of the triple belted 1080 I/O connector to the plug marked I/O at the rear of the 290 or NIC-80. Connect the other end to the mating plug at the rear of the 294 controller. If a small I/O terminator board is in the only available slot at the rear of the 1080, remove it and place it in the other 80-pin slot at the rear of the 294. If a high speed reader is to be used with the disk controller, it should be plugged into this other 80-pin slot and the I/O terminator plug discarded.

9. Connect the AC power cord to the rear of the 294 controller and plug it into the appropriate AC supply.

10. Be sure the LOAD/RUN rocker switch is in the LOAD position, and then plug in the disk drive power supply. Wait for the drive to warm up. When it is warmed up the Lock flag will go down and the Load light will come on. Then open the front door and insert a disk cartridge. All Low Density Model 2315 disk cartridges are interchangable from one manufacturer to another, and any can be used.

11. Turn the LOAD/RUN switch to the RUN position and wait for the disk to come up to speed. The yellow Ready light will then come on.

12. Meanwhile, turn on the 1080 power and turn the Teletype power switch to LINE.

13. Be sure that the resident Binary Loader is that loaded using the Nico-Loadeon Bootstrap program and if it is not, load Nico-Loadeon as described under the Loading Programs section of the 1080 manual.

14. Place the DEMON/II Operating System tape in the tape reader. If a high speed reader is available, it may be used. Otherwise, place the tape in the Teletype low speed reader, printed side up, and turn the reader to START.

15. Set the Switch Register to 7777_8 (00 000 000 111 111 111 111), where 1 means "up" and 0 means "down."

16. Depress Load PC and press Execute.

17. Depress Continue and press Execute.

18. The tape should start reading in.

19. When the tape has read in to the last row of holes, the program will stop. To initialize the disk, press Start. The computer will stop immediately. Then depress Continue and press Execute. DEMON/II will write the appropriate segments onto the disk, type a carriage return, a line feed and a *. It is then ready for commands.

20. Type DIRECTORY followed by a carriage return.

    a. If "garbage" is printed on the Teletype, stop the program and restart it at 7600. This is accomplished by SR=7600, Load PC, Execute, Continue and Execute. Then type KILL and answer Y to ARE YOU SURE? This zeros the directory on a virgin disk.

    b. If a list of files is produced, wait for the list to finish. or interrupt it by typing Q. These are files currently stored on that disk.

21. Type STORE SYSGEN 0-7577;0:P and follow this with a Return. This saves the disk system generating program on disk for future use in initializing new disk cartridges.

22. The DEMON (Disk Executive MONitor) is now ready for use.

23. Place the FT-Nmr tape in the reader and type BINLDR and follow it with a Return. The tape should read in and halt when read in.

24. Place the FT-Nmr Disk Patch in the reader, press Continue and press Execute. This tape should read in and halt at the Rubout in the trailer.

25. Set the Switch Register to 7600_8, depress Load PC and Execute, then depress Continue and press Execute.

26. The monitor will start and type a *.

27. Type STORE FTNMR 0-7577;0:P
This stores the FT-Nmr program on the disk. To start the FT program type GO followed by a Return. To call the monitor from the FT program, the command is MO.

The storage of additional programs and spectra is discussed in Chapters II and VII.

II. INTRODUCTION TO DEMONOLOGY

The Disk Executive Monitor System (DEMON/II) is a series of modular routines which reside on the Nicolet 294 Disk System and are called from 1080 programs by a few simple commands. The standard Binary Loader is used to load these programs initially, and, in the process of loading, the Binary Loader is replaced by a new series of subroutines called the monitor "head." The DEMON/II head resides in locations 7600 - 7777 and always calls the entire monitor and disk directory into core when the computer is started at 7600. Each time the monitor is called, the head first swaps out the contents of memory and reads in the monitor and the directory. After the various programs are stored or loaded, the command GO starts the program and causes the monitor to vanish. Thus, to the user, the DEMON is invisible, appearing and disappearing only when invoked by command.

A. Loading DEMON/II

DEMON/II is loaded using the standard Binary Loader, which is loaded using Nico-Loadeon, NIC-80/S-7115. If you have any doubts that this loader is present, you must load it as described in the Loading Programs sections of the 1080 Manual or the Programming Manual.

Once the Binary Loader is resident, load the DEMON tape by placing it in the Teletype reader or the high speed reader, and starting the Binary Loader at 7777_8. The tape should read in and stop at the end while the program loads the disk. If the tape stops at any place in the middle and the Teletype bell rings, a checksum error has been detected. If this occurs, the Binary Loader must be reloaded, and the entire process started over. If checksum errors persist, the tape may be torn or wrinkled, or the Teletype may need adjustment.

When the tape has read in successfully, the program just loaded will stop. To start the disk initialization program press Stored Program Start on 1080's and Data Processor Start on all other machines. The program will stop immediately. Then depress Continue and press Execute. DEMON/II will then start the disk monitor program and await Teletype commands. If the program "hangs up" the computer does not recognize the disk. Usually this means that the disk switch is not in the RUN position.

When the monitor starts after loading, or whenever it is started at location 7600, it will type a carriage return and line feed and then a * to indicate it is ready to accept commands. The display will also be active, allowing the user to view whatever section of memory is selected by the Readout Starting and Size buttons. The first command after loading a disk system from scratch should always be STORE SYSGEN 0-7577;0:P.

A. Basic Commands

DEMON/II's Keyboard Monitor will accept a command string which has the following general format.

COMMAND NAME-device arguments: options

Typing a Rubout deletes one character to the left with each occurrence with the deleted characters being enclosed in left slashes (\). If Line Feed is typed, the Keyboard Monitor will print the command string which has been typed with all the deleted characters removed and await further characters. If CTRL/O is typed, the entire line is cancelled. Typing a Return causes the command string to be processed by the Keyboard Monitor.

If no device is appended to the file name, unit 1 is assumed. For those commands which do not have a file name included in their command string, again unit 1 is assumed if no device is specified. Options which are preceeded by a colon (:) should be the last argument in the command string. A space must appear between the COMMAND and NAME and it is a good practice to include them between the NAME and arguments. A dash (-) and no space must be between the NAME and device whenever it is specified. Only the first three letters in the COMMAND and only the first six in the NAME are significant. The remaining letters are ignored and may be omitted.

DEMON will accept the following commands:

DIRectory	List the disk directory
KILl	Kill the entire disk directory
STOre	Store the specified file
LOAd	Load the specified file
DELete	Delete the specified file
BINldr	Call the Binary Loader
GO	Start the loaded program at the specified address.
RUN	Load the specified file and start at program starting address
EXPunge	Expunge all non-protected files for directory.

Each of them has its own additional modifiers and conditions which are discussed below. Arguments in small letters are optional.

1. KILl device

The command to KILL the disk directory or specified device or unit 1 is appropriate only when the user desires to wipe out all information on the disk. This might be true if a new disk cartridge has been loaded, if the system has just been installed, or if the experimenter wishes to delete the results of a large number of experiments at once. Since the result of KILL is so disastrous, the typing of a Return produces the query:

ARE YOU SURE?

If the user ansers this question with a Y, the disk directory is zeroed, and an answer with any other character aborts the KILL command. Thus KILL will zero the directory on disk 1, while KILL-D2 will zero disk 2, and so forth.

2. DIRectory device :F

This command causes a listing of all files stored on the disk specified or unit 1, followed by the number of disk tracks which they occupy. These files may be either programs or spectra from the data area of memory. They are all stored in the same binary format. A directory listing looks like this:

```
*DIR

FTNMR .P        3
NICBUG          1
SPCTRM          2
HINB2           1

    576 TRACKS LEFT
```

The names on the left are the actual file names, which are 1 to 6 character names containing any alphabetic or numeric information. The right hand column contains the number of disk tracks or blocks which each file occupies. Each track contains 1536_{10} or 3000_8 words. The message at the bottom tells how much space remains on the disk for additional storage. The ".P" following a file name indicates that this file is protected from being expunged.

If the user types the DIRECTORY command followed by the modifier :F, a full directory is typed out in the following format:

```
*DIR:F

FTNMR .P        0   7600;       0      20       3
NICBUG       4000   2000;    4700      23       1
[EMPTY]    114000   2000;  114700      24       1
SPCTRM     100000   4000;    7600      25       2
HINB2      114000   2000;  114700      27       1
[EMPTY]    171773  16001;1740000      30     575

    576 TRACKS LEFT
```

where the columns are file name, first core address, number of words stored, program starting address, disk track address, and number of tracks used.

3. STOre FILNAMe-device :BP or :B
m-n;o :P

Files are stored from memory on the disk by using the STORE command. The memory block can be either program or data memory and this block can be specified

either by octal addresses or by the Readout pushbuttons. The STORE command assumes that the previous program occupies all of program memory and that the monitor is not present. Thus, the monitor is invisible, seeming to occupy no memory at all below 7600. For instance, if DEMON was called from within the FT-Nmr program, the command to store locations 0-7577 would store the FT-Nmr program rather than the monitor.

a. Storing Data

The STORE command gets the addresses of the memory block to be stored on disk from readout memory pushbuttons if the suffix :B is added to the store command. Since the display is active while DEMON/II is running, :B says, store the region currently displayed. Each file to be stored on disk must be given a unique 1 to 6 character file name preceded and followed by a space.

To store a file named SPEC1, occupying a region selected by the pushbuttons, type:

STORE SPEC1 :B

When a Return is typed following the B, the monitor will copy the region of core memory selected by the Readout Starting and Size pushbuttons onto disk and indicate that it is ready for new commands by typing a *.

b. Storing Programs

When programs are stored, the region of memory is selected by entering the actual starting and terminating octal addresses. Remember that the monitor appears to occupy only locations 7600-7777.

To store the FT-Nmr program which, it is assumed, has just been loaded by the BINldr command from paper tape, enter the command

STORE FTNMR 0-7577;0

This command says write locations $0 - 7577_8$ onto disk and enter the file name FTNMR in the disk directory. There must be one or more spaces between the end of the file name and the first digit of the octal starting address. Spaces between the two addresses are optional, but there must be a dash between them. DEMON/II will not allow you to store address 7600-7777, since reloading them would wipe out the monitor. The 0 following the semi-colon specifies the program starting address as 0. If not included, 7600 would have been specified by default.

c. Protecting Files

A file is said to be protected from being expunged if the notation .P appears after its name in the disk directory.

A protected file is formed by adding the suffix :P to the STORE command string. For instance, to store and protect PROG1, which occupies locations 0-550, we type:

STORE PROG1 0-550 :P

A file which is stored from the pushbutton selected area can also be protected by typing :BP. Starting address information must preceed the colon.

d. Duplicate Files

The request to store a file having the same name as a file already in the disk directory will cause the question

DELETE:

to be typed. If the user answers Y, the specified file will replace the old one. If any other character is typed, that STORE command is ignored.

4. DELete FILNAM.extension-device

A file may be deleted from the disk by simply deleting the directory entry. This is accomplished by the DELETE command. The command has the format

DELETE FILNAMe-device

where FILNAME is any desired file in the directory of the device specified or of unit 1. If the file is found, the question

DELETE:

will be asked. This must be answered Y to delete the file. If any other character is typed, the command is ignored. If there is not any file in the directory having the specified name the message

FILE NOT FOUND

will be given. Files having extensions .A, .B or .C may also be deleted but the extension must be typed by the user. Otherwise the file having no extension is deleted.

5. LOAd FILNAMe-device :B
nnnn

The LOAD command loads the specified file from the device specified or from unit 1 into memory in the same region from which the data was originally taken.

Thus, if the command STORE NMRCAL 0-7577 was at sometime given, the command

LOAD NMRCAL

will cause that file to be loaded into locations 0-7577.

A file can be loaded into any portion of data memory regardless of where it came from if this is desired. Suppose that the first 2K of data memory are specified when the store command is given for SPEC1

STORE SPEC1 :B

The information that the file SPEC1 was originally stored in locations 100000-103777 is stored in the directory along with the file name. Thus, the command

LOAD SPEC1

will load the contents of this file into memory locations 100000-103777. However, this file can also be loaded into other regions of data memory of equal size which are selected by the Readout pushbuttons. This is accomplished by giving the command

LOAD SPEC1 :B

This command says to ignore the original file address information and load the file SPEC1 into the region specified by the Readout pushbuttons. It is important, however, that this new region be of the same size as the original region. If the size specified by the readout pushbuttons differs from that of the original file, the message

WRONG SIZE?

will be printed.

LOAD NICBUG-D2 114000

This command says load NICBUG from disk unit 2 starting at address 114000. The address given here overrides the information given at store time.

It should be noted, however, that very few programs are relocatable, and that attempting to run such a program at the new address will probably result in failure unless the program is very carefully written.

6. GO nnnn

The GO command starts the program just loaded at the specified address. If no address is given, address 0 is assumed. The following two commands load the FT-Nmr program from the disk and start it at location 0.

```
LOAD FTNMR
GO
```

To start a program at an address other than 0 the address must be specified in the GO command. This address must be separated from the GO by one or more spaces. For instance, let us assume that the Assembler is to be loaded and started. This program starts at address 2000. The following commands would be given:

```
LOAD ASMBLR
GO 2000
```

When the Return is typed following the address, DEMON swaps the original program back in and jumps to location 2000, in this case starting the Assembler program. It should be noted that it is not necessary to reload a program to restart it. If DEMON has been called by command from the FT-Nmr program, the command GO will restart the FT at any time.

7. BINldr

The BINLDR command restores the original Binary Loader to locations 7632-7751 so that it can be used to read in programs from either the high speed or low speed reader. This is exactly the same Binary Loader as is loaded using the Nico-Loadeon Bootstrap tape. It automatically starts after it is loaded and selects either the high speed or the low speed reader and reads in tape until it finds a Rubout in the trailer section of the tape. It then halts. Additional paper tapes can be read in as usual by depressing Continue and then pressing Execute or by restarting the loader at 7777_8. When all desired tapes have been loaded, they can be stored on the disk by restarting the monitor at location 7600. Restarting at this point calls in the monitor and allows the use of the STORE command to store whatever programs were just loaded.

8. RUN NAME-device

The RUN command loads the file and then transfers control to the starting address that was specified when the program was stored. If the program did not have a starting address specified when the program was stored, the RUN command performs the same function as the LOAD command except all arguments are ignored.

9. EXPunge-device

The EXPUNGE command will remove all non-protected files from the directory of the device specified. As in the KILL command,

```
ARE YOU SURE
```

is typed to confirm this operation. Typing a Y will allow the operation to proceed. Any other character will abort this command. All consecutive empty tracks on the directory that is expunged will be combined into one empty file.

C. Disk Error Messages

DISK READ ERROR

A disk hardware read error occurred. Proceed at your own risk.

NO ROOM

A directory search revealed that there isn't a free disk segment large enough to contain your file. However, if you list the directory this process will combine two or more consecutive empty files into a single file. At this point you can try to store the file again.

FILE NOT FOUND

No such file exists on the device specified or unit 1 if no device was specified.

WRONG SIZE

The size on the status buttons does not agree with the information provided during the storing of the file.

ILLEGAL DEVICE

A disk unit was specified that does not exist.

SYNTAX ERROR

There is a section in your command string which the Keyboard Monitor could not interpret or would not accept.

ILLEGAL COMMAND

There is no such command.

ILLEGAL START

The address given in the GO command was in the Monitor Head.

MONITOR ERROR

A hardware read error occurred or an "impossible" software error occurred. If a soft error occurred, location 7704 will contain a -1. If a hard error occurred, it will contain a +1. Normally it contains a zero.

D. System Start-Up and Shutdown

When you are finished with the 1080 disk system for a time and wish to turn it off, you should proceed as follows:

1. Turn the disk LOAD/RUN switch to LOAD,
2. Press Stored Program STOP,
3. Turn off the Teletype,
4. Turn off the 1080 power key switch,
5. Unplug the disk if desired.

To start up the disk system once DEMON has been loaded, it is merely necessary to

1. Turn on the 1080 and Teletype,
2. Plug in the disk and turn the LOAD/RUN switch to RUN,
3. Set the Switch Register to 7600.

When the Ready light comes on,

4. Depress Load PC and press Execute,
5. Depress Continue and press Execute.

DEMON will automatically be called from the disk and will type a *. When DEMON has been called from a cold start like this, it is always a good idea to load a program by a specific LOAD or RUN command rather than just typing GO. This is simply protection against starting some unwanted program. Never start a program using the Data Processor Start button if the monitor may still be resident.

E. DEMON Bootstrap

There are several levels of program failure that should be considered in restarting after a "crash." If a simple experimental program has failed, DEMON can probably be recalled by simply starting at location 7600.

If, on the other hand, such a program has wiped out the monitor head which resides above 7600, it is necessary to recall more information from disk. This can be accomplished by toggling in the following "bootstrap loader" at the Switch Register.

7566	2111566
7567	4546
7570	44541
7571	2036
7572	1570
7573	46526
7574	1573
7575	2405772
7576	2705575
7577	1573

When this bootstrap loader has been toggled in, start it at 7566. The monitor head and DEMON should be reloaded from the disk and started. If this fails, the disk has been wiped out as well and the entire DEMON program must be reloaded as described in Chapter I.

F. Initializing Additional Disk Cartridges

The Nicolet 294 disk system utilizes low density replaceable disk cartridges of the IBM Model 2315 type. These are available from a number of manufacturers and all meet the same standards.

To initialize a new disk, it is necessary to place a copy of the monitor on this new disk. This is done using the SYSGEN program which was saved when the first disk was initialized. If this program was not saved or has been deleted since that time, the new disk can only be initialized by the same procedure as used for the first disk.

However, if the program SYSGEN is still on one disk, it should be loaded by

RUN SYSGEN

The program will halt with the PC = 1.

Then, turn the LOAD/RUN switch to LOAD and wait for the LOAD light to come on. Open the disk drive door and remove the old disk. Replace it with the new one. Turn the switch back to RUN and wait for the ready light.

Continue SYSGEN by depressing Continue on the 1080 and pressing Execute. The disk will initialize itself and the monitor will return and type a *.

If this is indeed a virgin disk, type KILL to enter a zero directory on the disk. Answer Y to the question ARE YOU SURE? Then store SYSGEN on this disk as well by typing

STORE SYSGEN 0-7577;0 :P

The new disk is now initialized.

III. PROGRAMMING THE DISK

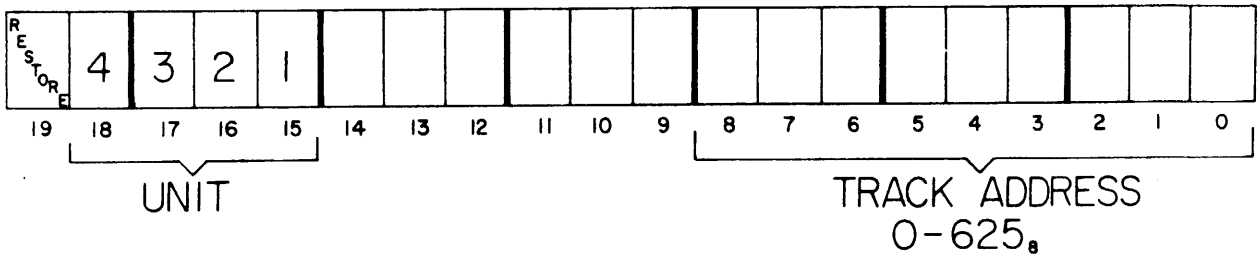
The Nicolet disk system can be programmed quite easily using a few simple commands as described in this chapter. The advantage of doing one's own programming for this disk is that one can write rather complex routines which take maximum advantage of existing programs and call them from disk one at a time as needed.

The following four instructions comprise the entire disk instruction set. They are not recognized by the Assembler and must be defined by an equals sign somewhere in each program that refers to them.

Octal Code	Instruction	Meaning
46526	RDISK	Read one word from the disk and skip
6536	WDISK	Write one word onto the disk and skip
4546	LTRACK	Load track address from AC
44541	DSTAT	Read disk status word into AC

LTRACK

In order to read or write information onto the disk it is first necessary to select the track onto which information is to be written. This is accomplished by the LTRACK command. This command loads the contents of the AC into the disk track and device registers from bits set as follows:



Bits 0 - 8 specify the track to be loaded. This is an octal number between 0 and 625. Loading a higher numbered track will result in the disk not being able to acknowledge successful execution of this command. Only bits 0 - 8 are loaded. Tracks 0 - 17 are reserved for use by DEMON and reading or writing on these tracks will result in its complete exorcism.

Bits 15 - 18 specify the unit number. It is possible for the controller to address four different disks, having unit numbers 1 - 4 set as specified by these bits. If bit 15 is 1, unit 1 is selected; if bit 16 is 1, unit 2 is selected; and so forth. A single disk system always selects unit 1 for that disk and attempts to address nonexistent disks will result in the controller not acknowledging the attempt.

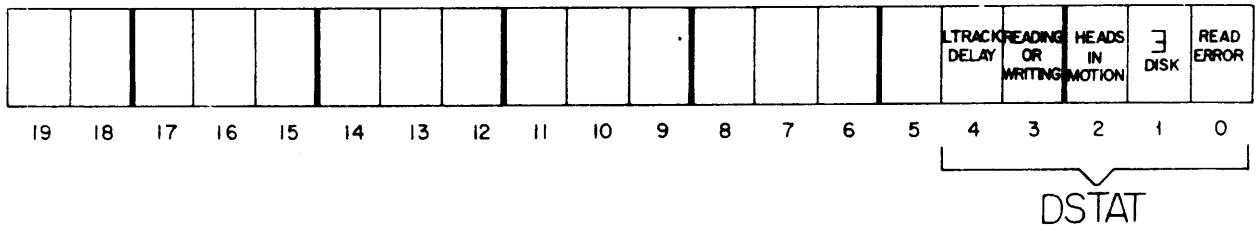
Bit 19 is the Restore bit and has the effect of restoring the heads to track 0 of whichever disk is selected. Both bit 19 and the unit bit must be selected, however, to restore a particular disk head to track 0.

A typical piece of code for selecting a track would be as follows:

```
UNIT,     100000        /UNIT 1
          . . .
START,    MEMA (20      /SELECT TRACK 20
          A+MA UNIT     /ADD ON UNIT NUMBER
          LTRACK        /HAS BEEN DEFINED TO ASSEMBLER BY
                        /LTRACK=4546
```

DSTAT

Once the track register has been loaded, the computer user must then know when the heads are in position and ready for further commands. This is accomplished using the Disk Status Register. This register is read into the AC using the instruction DSTAT. The information in DSTAT is contained in bits 0-4 as shown below.



Bit	Meaning
0	A checkword error occurred during the last read. This bit is not cleared until a successful read takes place.
1	There exists a disk having the unit number specified by the last LTRACK command. If there is no disk, or if it is not READY, this bit will be 1.
2	Heads in motion. The disk heads are still moving toward the track specified by the last LTRACK command. This bit becomes 0 when the heads stop moving.
3	Reading or writing. This bit is one if the disk is still reading or writing data.
4	LTRACK Delay. This is a 100 μsec delay which occurs if the track loaded is on the other side of the disk from the last one. Bit 2 does not go high until after this 100 μsec delay, so an additional delay is needed to keep from reading or writing too soon.

As can be seen from the above, DSTAT should be all zeroes when the disk is ready to perform a new task. In the case that a new task is to be performed after a read error occurred on the last track read, it is necessary to ignore bit 0. This can be accomplished by

```
D1,   DSTAT      /READ DSTAT
      ANDZ (36   /SKIP IF BITS 1-4 are 0
      JMP D1
```

Reading from the Disk

The RDISK command reads and assembles one word from the disk and sums it with the checkword. It transfers this word to the AC and performs a skip when this is accomplished. RDISK will have no effect until the heads are positioned over the beginning of a track. This assures that the information will come out starting at the same point that it was written onto the disk in the first place.

Once the first word has been read, the user has a maximum of 25.6 μsec to read the next data word. Since the 1080 has an instruction execution time of 3.3 μsec for direct instructions and 4.9 μsec for indirect instructions, a rather tight loop must be constructed so as not to miss any data words. The following program will accomplish this:

```
        MEMA TRACK      /GET TRACK NUMBER FROM MEMORY
        A+MA UNIT       /ADD UNIT NUMBER
        LTRACK          /LOAD TRACK AND UNIT
D1,     DSTAT           /WAIT FOR HEAD MOVEMENT
        ANDZ (36        /WAIT FOR DSTAT TO ZERO BITS 1-4
        JMP D1          /LOOP UNTIL READY
R1,     RDISK           /READ DISK WHEN READY AND SKIP
        JMP R1          /LOOP TILL READY
        ACCM @ POINT    /STORE AC IN LOCATION
        MPOM POINT      /INCREMENT POINTER
        MMOMZ COUNT     /CHECK COUNTER FOR DONE
        JMP R1          /LOOP UNTIL ALL WORDS READ
        . . .
TRACK,  0
UNIT,   0
POINT,  0
COUNT,  0
```

The above program assumes that the index pointer POINT and the word counter COUNT have been set before entering the loop. Each track may contain a maximum of 1536_{10} or 3000_8 words. It may contain less, but the number read from the disk must always equal the number of words which were written onto the disk. If this is not done, the checkword will not agree with the information read.

The checkword always follows the last word written on a track and is automatically read and compared by the 294 controller hardware. Disk reading errors occur very seldom, about once for every 10^{11} bits read. However, since this error rate is finite, it is desirable to check for read errors regularly and re-read the offending track if they occur. This can be accomplished by checking DSTAT after the reading of a track is complete. This operation is accomplished automatically within the monitor head.

DSTAT cannot be examined, however, until the reading and checkword comparison ceases. For this reason, it is necessary to write a program that waits for bit 1-4 to go to zero and then checks bit 0 to see if it is 0. Such a loop might be as follows:

```
D1,   DSTAT          /READ DSTAT
      ANDZ (36       /WAIT FOR BITS 1-4
      JMP D1         /LOOP UNTIL READY
      EXCT AC0       /TEST AC0
      JMP ERR        /GO TO ERROR ROUTINE IF 1
```

Alternatively one can perform some other task for 50 μsec before reading DSTAT.

Writing Onto the Disk

The writing procedure is exactly the same as that used for reading except that once the appropriate track is loaded, the WDISK command is given instead of the RDISK command. There is the additional distinction that the accumulator must remain unchanged for at least 12.8 μsec after a WDISK command to give the controller time to decompose the information into a bit string to be written onto the disk. For this reason, a loop such as the following is required. Note that there are two dummy instructions at the bottom of the loop so that the last word of the list is written before the accumulator changes. This program assumes that POINT and COUNT have already been set and that the track has been loaded.

```
W0,   MEMA @ POINT     /GET WORD FROM LIST
W1,   WDISK            /SKIP AND WRITE
      JMP W1           /WAIT UNTIL READY
      MPOM POINT       /INCREMENT POINTER
      MMOMZ COUNT      /TEST FOR END OF LIST
      JMP W0     /     /LOOP UNTIL LIST DONE
      ACCA             /DUMMIES TO FILL WHILE LAST WORD IS
      ACCA             /WRITTEN
```

When the entire track has been written, the programmer can then test his own program to see if there are more tracks to be written and continue if desired. If he is done, the checkword will automatically be written if another datum is not presented to the controller within 25.6 μsec.

IV. PROGRAMMING WITH THE DISK USING THE DEMON/II HEAD

The heart of the DEMON/II Disk Monitor is the disk handler. This handler is a subroutine which transfers data to and from the disk and memory. The entry point of this subroutine is 7612. Upon entry to the subroutine, if the AC contains a zero, a read operation is assumed. If the AC is non-zero, a disk write operation is assumed. The three locations following the subroutine call are used as arguments in controlling the I/O transfer. The first location following the call contains the disk unit in bits 15-18 (see Section III). Bits 0-8 of this word contain the track number which is to be the beginning of your transfer. The second location following the call contains the number of words to transfer. The third word contains the address of the beginning of the buffer for the transfer. For instance, the format of a write memory onto the disk would be coded as follows:

```
        ONEA            /SET AC TO NON-ZERO, WHICH SIGNALS WRITE
        JMS @ DISK      /CALL DISK HANDLER (IF CURRENT PAGE, NO @)
        ARG1            /BITS 15-18 CONTAIN DISK UNIT, BITS 0-8
                        /CONTAIN THE STARTING TRACK
        ARG2            /CONTAINS THE NUMBER OF WORDS TO TRANSFER
        ARG3            /CONTAINS THE CORE ADDRESS OF TRANSFER.
        . . .           /CONTROL RETURNS HERE
        . . .
        . . .
DISK,   7612
```

The subroutine automatically selects the next sequential track when the current one is filled. The user need not concern himself with the number of tracks to be written and the remainder if any. While reading, if a bad checksum is encountered at the end of a track, the subroutine will attempt the read operation two more times before signalling a hardware error. When this happens, location 7704 is set to a +1.

When transfer of program control is made to 7600, the start of the disk head, the head performs the following operations. First, it writes locations 3000-7577 onto tracks 1 and 2 of unit 1. It then reads the directory of unit 1 and the Keyboard Monitor into those same locations. A user program may also use tracks 1 and 2 for scratch storage but they will be destroyed when the monitor is called. Tracks 1 and 2 can be restored into core locations 3000-7577 by calling subroutine DISOLVE whose entry point is 7751.

Locations 7760-7777 can be used for scratch storage but they too will be destroyed when the Keyboard Monitor is called or when the Directory Function subroutine is called.

V. PROGRAMMING USING THE DIRECTORY FUNCTION SUBROUTINE (DIRFUN)

All monitor operations dealing with the directory are performed by the Directory Function Subroutine (DIRFUN) which resides on track 7 and is 600_8 locations long. When in core this routine must reside from 7000-7577. DIRFUN overlays itself with the Directory Listing Routine (DIRLST) which resides on track 10 and is 400_8 locations long whenever a directory listing operation (LIST, EXPUNGE and KILL) is signalled. This routine resides from 7200-7577. DIRFUN performs the following functions:

1. Search for a file and return directory information.
2. Close a file. That is, make a directory entry including new information and then delete the old entry if any.
3. Delete a file (subfunction of Close).
4. Perform a directory listing in one of two formats.
5. Expunge all non-protected file entries.
6. Kill a directory.
7. Search for an empty file of a specified or unspecified length in words.

The following coding will call DIRFUN into core:

```
        ZERA
        JMS @ DISK      /READ OPERATION
        0100007         /UNIT 1, TRACK 7
        600             /600 WORDS
        7000            /LOAD AT 7000
        . . .
        . . .
DISK,   7612
```

When the subroutine is called into core, it is the user's responsibility to save the area in which the routine resides (7000-7577) and the directory area (3000-5777), unless the destruction of these locations can be permitted. The user should therefore not call the routine from locations 3000-5777 or 7000-7577 as these areas will be occupied by the directory and the routine itself. Upon first calling the routine, location 7764 should be set to zero. After this, if this location is not rezeroed, DIRFUN will assume the directory is in core, and will not perform a read of the directory. This insures that the correct directory is read in multiple disk systems. The entry point of the subroutine is 7000.

Following the subroutine call, the next three locations should contain arguments that will specify the operation to be performed. The fourth location is where the subroutine will return if a hard or soft error occurred. The fifth location is where the subroutine returns if no errors occurred. This is diagrammed below.

The first argument should contain the octal number of the disk whose directory you want to assess. Only numbers 1-4 are significant. Numbers outside this range will produce unpredictable results. The second argument contains the

number of the directory operation. Zero signals a directory Listing, 1 signals a Close Directory Entry and 2 signals a Search operation. In Search and Close the third argument is a pointer to the location of a two word block which contains the filename in stripped ASCII to be used. In the directory listing operation, the third argument signals which subset of that operation is to be performed. If that argument contains a 0, a partial listing of the directory is signaled. If it contains a 1, a full listing is signaled. A 2 will kill the directory and 3 will expunge all non-protected files. In any of the listing operations two or more consecutive empty file entries are combined to a single entry.

Close and Search use locations 7760 and 7770-7772 in the Monitor Head to communicate with the user program. In the Monitor Head these locations are labeled SYSTRT, OARG1, OARG2 and OARG3. OARG1 contains the starting track of the file, OARG2 contains the size of the file, OARG3 contains the core address and SYSTRT contains the program starting address. Close uses these arguments to create a directory entry and Search returns these constants when it finds the directory entry. Close will first search the directory to see if a file with the same name is present. If there is, it will type DELETE: after which it expects the user to type a Y if the operation is to proceed. Otherwise a normal return will be taken but the operation will not continue. Close will also create an empty file to represent the empty space created if your file did not use all the tracks that were in the empty slot it used. If the directory overflows, the error return will be taken.

DIRFUN CALLING EXAMPLES

```
JMS @ DIRFUN
    DISK UNIT # (1-4)
0 = Listing
0 = Short listing  1 = Full listing
    2 = Kill  3 = Expunge
Error return
Normal return
```

```
JMS @ DIRFUN
    DISK UNIT # (1-4)
1 = Close  2 = Search
NAMPNT    /pointer to file name
Error return
Normal return
```

7751	DISOLV	-1 causes search for empty file, largest file if OARG2 = 0
7760	SYSTRT	program starting address
7770	OARG1	starting track number
7771	OARG2	file size, delete if 0
7772	OARG3	core start point

Below is an example of a Close call on disk #2 of the file ABCDEF:

```
        /ASSUME DIRFUN IS IN CORE. THE DIRECTORY INFO IS IN
        /THE MONITOR HEAD
        JMS @ DIRFUN      /CALL THE SUBROUTINE
        2                 /DISK UNIT 2
        1                 /CLOSE
        NAMPNT            /ADDRESS OF FILE NAME
        JMP ERR           /ERROR RETURN
        . . .             /NORMAL RETURN
DIRFUN, 7000
NAMPNT, 414243            /ABC
        444546            /DEF
```

A file can be deleted by simply closing a file with OARG2 (the word count) set equal to 0.

Search complements the Close operation in that it will return the directory entry information to the Monitor Head if it finds the entry. The error return is taken if the entry is not found. The coding below will search for file ABCDEF on disk unit 1.

```
        /ASSUME DIRFUN IS IN CORE
        JMS @ DIRFUN
        1                 /DISK UNIT 1
        2                 /SEARCH OPERATION
        PNTNAM            /POINTER TO FILENAME
        STOP              /ERROR RETURN, FILE NOT FOUND
        . . .             /CONTROL RETURNS HERE WITH DIRECTORY
        . . .             /INFORMATION IN OARG1-3, SYSTRT.
        . . .
PNTNAM. 414243            /ABC
        444546            /DEF
DIRFUN, 7000
```

The empty search operation is signaled by setting location 7751 (DISOLVE) to -1 and having the pointer to the filename point to a location containing zero. If OARG2 is not zero, the first entry that contains the same or greater number of words is returned in the Monitor Head. If OARG2 contains a zero, the largest empty file will be returned in the Head. When searching for the empty file, the error return is always taken. For this reason, the user must check to see if the operation returned an empty space large enough to contain the transfer selected.

The following code searches for the filename SPEC1 and deletes it if found, then stores it on the disk. The size and starting address are stored in DSIZE and DSTART.

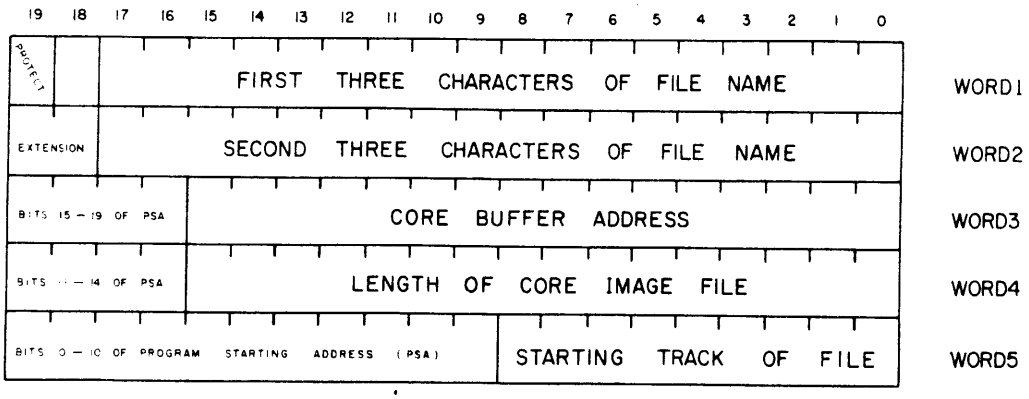
```
ONEA                    /SET WRITE
JMS @ DISK
100001                  /UNIT 1, TRACK 1
  4600
  3000                  /WRITE OUT 3000-7577 ON TRACKS 1 AND 2
ZERA                    /READ
JMS @ DISK
100007                  /DIRFUN ON TRACK 7
   600                  /600 WORDS
  7000                  /LOAD AT 7000
JMS DPATCH              /REMOVE "DELETE" MESSAGE, IF DESIRED
ZERM @ OARG2            /SIGNALS DELETE
JMS @ DIRFUN            /DELETE ANY PREVIOUS FILE BY THAT NAME
     1                  /ON UNIT 1
     1                  /CLOSE
FILNAM                  /POINT TO FILE NAME
ACCA                    /ERROR RETURN OK
MEMA DSIZE              /SET CORE LIMITS
ACCM @ OARG2            /IF 0, SEARCHES FOR LARGEST EMPTY FILE
MEMA DSTART
ACCM @ OARG1
MONM @ DISOLV           /SIGNALS EMPTY SEARCH
JMS @ DIRFUN            /SEARCH FOR LARGE ENOUGH SPACE
     1                  /UNIT 1
     2                  /SEARCH
ZERZ                    /ALWAYS TAKES ERROR EXIT
STOP                    /"IMPOSSIBLE" ERROR
/OARG1 NOW CONTAINS TRACK # AND OARG2 SIZE OF SPACE FOUND
MEMA DSIZE
A-MA OARG2
SKIP AC19 ZAC
STOP                    /NO SPACE LARGE ENOUGH
MEMA DSIZE
ACCM @ OARG2            /RESTORE ACTUAL SIZE
MEMA @ OARG1
A+MA KUNIT              /ADD ON UNIT BIT
ACCM DTRACK             /SAVE FOR DISK WRITE
/UPDATE DIRECTORY
JMS @ DIRFUN
     1                  /UNIT 1
     1                  /CLOSE
FILNAM
STOP                    /ERROR
```

```
/NOW RESTORE MEMORY
/READ IN LOCATIONS 3000-7577 FROM TRACKS 1 AND 2
          ZERA
          JMS @ DISK
          100001
            4600
            3000
/NOW WRITE FILE INTO DISK SLOT
          ONEA
          JMS @ DISK
DTRACK,   0              /LOADED ABOVE
DSIZE,    nnnn           /SET BEFORE ENTRY
DSTART,   nnnn           /SET BEFORE ENTRY
          .
          .
          .
DISK,       7612
DIRFUN,     7000
OARG1,      7770
OARG2,      7771
OARG3,      7772
/SUBROUTINE REMOVES "DELETE:" MESSAGE
DPATCH,       0
          MEMA (1372                  /JMP 7372
          ACCM @ DP1
          JMP @ DPATCH
DP1,        7346

FILNAM,   636045                      /SPEC1
          432100
```

VI. FORMAT OF DIRECTORY ENTRY

A directory entry is five words long with the core information packed in the following format.



Bit 19 is set if a file is protected. Bits 18 and 19 of WORD 2 constitute the extension. A binary number from 0-3 can be expressed in these bits. On a directory listing a letter will be printed for each number.

0 = SPACE (no extension)
1 = A
2 = B
3 = C

These extensions are used to represent ASCII, BASIC and Core Image files.

LAYOUT OF DEMON/II MONITOR ON DISK

TRACK	FUNCTION
0	DEMON Bootstrap
1-2	Monitor save area
3	Directory
4	Keyboard Monitor
5	Paper Tape Binary Loader
6	Monitor Head
7	Directory Function Subroutine (DIRFUN)
10	Directory Listing Routine (DIRLST)
11	General Input-Output string Decoder
12	Input-Output Handlers
13-14	Reserved for Future Expansion
15-17	Unused

```
/DISK MONITOR READ


/DISK MONITOR READ
/ 7/2/73
/CALLING SEQUENCE
/     AC=1 FOR WRITE, 0 FOR READ
/     JMS DISK
/     ARG1          /BITS 15-18 ARE UNIT, BITS 0-8 ARE TRACK NUMBER
/     ARG2          /NUMBER OF WORDS TO TRANSFER
/       ARG3          /ADDRESS OF TRANSFER
/    CONTROL RETURNS HERE


*7600
RDISK=46526
WDISK=6536
LTRACK=4546
DSTAT=44541
   7600   30000  READ,  ONEA  /PUT 1 IN AC
   7601 2001612     JMS DISK  /CALL DISK HANDLER
   7602  100001          0100001          /UNIT 1, TRACK 1
   7603    4600          4600       /NUMBER OF WORDS
   7604    3000  C3000,  3000       /ADDRESS OF TRANSFER
   7605 2001612  NOSAVE,  JMS DISK  /START HERE WITH AC CLEARED FOR NOSA
   7606  100003          0100003          /UNIT 1, TRACK 3
   7607    4600          4600       /NUMBER OF WORDS TO READ
   7610    3000          3000       /WHERE TO PUT THEM
   7611       0          JMP MONSET  /START KEYBOARD MONITOR
MONSET=6000
   7612       0  DISK,  0          /ENTRY POINT IN DISK HANDLER
   7613  405120          SKIP ZAC /IF AC=0, THEN IT IS READ
   7614    1642          JMP WRSET
   7615 2001654          JMS SETENT  /SET UP POINTERS FOR READ
   7616 2001711  DRERR,  JMS TRWAIT   /WAIT FOR TRACK
   7617   46526  DKR1,  RDISK     /WAIT FOR WORD
   7620    1617          JMP DKR1
   7621 3405701          ACCM @ POINT  /STORE IT
   7622 2125701          MEOM POINT    /BUMP COUNTERS
   7623 2707702          MMOMZ COUNT
   7624    1617          JMP DKR1
   7625 2001725          JMS TRNEXT    /WE'LL HAVE TO ACCESS NEXT TRACK
   7626   44541    DSTAT
   7627    5110    SKIP ACO  /WAS READ SUCCESSFUL(BIT0=0)
   7630    1617    JMP DKR1  /YES
   7631 2707702    MMOMZ TRIES  /3 TIMES?
   7632    1635    JMP ERREIX  /NO, TRY AGAIN
   7633 2025704    ONEM ERRFLG  /SET FLAG AND EXIT
   7634    1617    JMP DKR1
   7635 2111705  ERREIX,  MEMA ERRCON
   7636 2405702    ACCM COUNT  /LET'S TRY READ AGAIN
   7637 2325701    M-AM POINT  /KNOCK DOWN POINTER
   7640 2705724    MMOM FUN1
   7641    1616    JMP DRERR   /TRY AGAIN
   7642 2001654  WRSET,  JMS SETENT  /SET UP POINTERS
```

```
7643 2001711          JMS TRWAIT
7644 3111701  W0,     MEMA @ POINT    /GET WORD
7645    6536  W1,     WDISK
7646    1645          JMP W1
7647 2125701          MEOM POINT      /BUMP COUNTERS
7650 2707702          MMOMZ COUNT
7651    1644          JMP W0          /NOT DONE YET
7652 2001725          JMS TRNEXT      /SWITCH TRACKS?
7653    1644          JMP W0
7654       0  SETPNT, 0       /SET UP POINTERS
7655 3111612          MEMA @ DISK     /GET FUNCTION WORD
7656 2405724          ACCM FUN1
7657    4546          LTRACK      /INTIALIZE DISK TO SOMETHING
7660 2125612          MEOM DISK
7661 3111612          MEMA @ DISK     /GET NUMBER OF WORDS
7662 2471604          A-MA C3000      /IS IT LONGER THAN A TRACK
7663  405124          SKIP AC19 ?AC     /SKIP IF ZERO OR MINUS
7664    1706          JMP REMAIN     /STORE REMAINDER
7665 2165703          ZERM REMAN     /NO REMAINDER
7666 2511604          A+MA C3000     /RESTORE COUNT
7667 2405702  REM1,    ACCM COUNT
7670 2125612          MEOM DISK          /GET NEXT ARGUMENT
7671 2405705          ACCM ERRCON   /SAVE COUNT IN CASE READ FAILS
7672 3111612          MEMA @ DISK    /GET STARTING ADDRESS
7673 2405701          ACCM POINT
7674 2125612          MEOM DISK       /BUMP FOR RETURN
7675  110003          MEMA (3         /SET UP NUMBER OF TRIES FOR ERROR
7676 2405700          ACCM TRIES
7677 1001654          JMP @ SETPNT
7700       0  TRIES,  0
7701       0  POINT,  0
7702       0  COUNT,  0
7703       0  REMAN,  0
7704       0  ERRFLG, 0
7705       0  ERRCON, 0
7706 2405703  REMAIN, ACCM REMAN     /STORE REMAINDER
7707 2111604          MEMA C3000
7710    1667          JMP REM1
7711       0  TRWAIT, 0
7712 2001717           JMS DDSTAT    /CHECK STATUS
7713 2111724          MEMA FUN1          /LOAD FUNCTION WORD
7714    4546          LTRACK
7715 2001717          JMS DDSTAT   /CHECK STATUS
7716 1001711          JMP @ TRWAIT    /YES!
7717       0  DDSTAT, 0   /CHECK STATUS
7720   44541  DO,     DSTAT
7721    2036          ANDZ (36
7722    1720          JMP DO
7723 1001717          JMP @ DDSTAT
7724       0  FUN1,    0
7725       0  TRNEXT, 0                  /GET NEXT TRACK
7726  410000           ACCA              /NOP
7727 2135724          MPOMA FUN1         /RIGHT BITS ARE TRACK
```

```
   7730 2001711           JMS TRWAIT
   7731 2111703           MEMA REMAN          /GET REMAINDER
   7732  405160          EXCT ZAC             /EXEC ON ZERO AC
   7733 1001612           JMP @ DISK          /EXIT, WE'RE DONE
   7734 2471604           A-MA C3000          /SUBTRACT 3000
   7735  405164          EXCT AC19 ZAC  /SKIP ON ZERO AND MINUS AC
   7736    1744           JMP TLAST
   7737 2405703           ACCM REMAN          /STORE THIS REMAINDER
   7740 2111604           MEMA C3000
   7741 2405705           ACCM FRECON
   7742 2405702           ACCM COUNT
   7743 1001725           JMP @ TRNEXT
   7744 2511604  TLAST,    A+MA C3000         /RESTORE VALUE
   7745 2405705           ACCM FRECON
   7746 2405702           ACCM COUNT
   7747 2165703           ZERM REMAN
   7750 1001725           JMP @ TRNEXT
   7751       0  DISOLVE, 0
   7752  170000           ZERA
   7753 2001612           JMS DISK
   7754  100001          0100001     /RESORE SAVE AREA
   7755    4600          4600
   7756    3000          3000
   7757 1001751           JMP @ DISOLVE
/THE FOLLOWING CODE IS A FUDGE TO ALLOW THE PAPER TAPE
/BINARY LOADER TO OVERLAY THE DISK MONITOR.  THIS CODING IS INSERTED
/HERE BY THE COMMAND BINLDR.
   7760   46526  RD,     RDISK
   7761    1760          JMP RD
   7762 3405775          ACCM @ RPNT         /DEPOSIT
   7763 2125775          MPOM RPNT
   7764 2707774          MMOM7 RCNT
   7765    1760          JMP RD              /NOT DONE YET
   7766    1634  BRET, JMP START  /EXECUTE BIN LOADER
   7767 2001751  BINGEN, JMS DISOLVE        /RESTORE CORE
   7770 2111776           MEMA BINLDC
   7771 2405724           ACCM FUN1     /SET UP FOR BIN LDR
   7772 2001711           JMS TRWAIT          /WAIT FOR TRACK

       /BRET HAD BETTER BE CHANGED TO 1634 BY NOW
   7773    1760          JMP RD              /POINTER ALREADY SET
   7774     153  RCNT, 153  /# OF LOCATIONS IN BIN LDR
   7775    7600  RPNT, 7600  /WHERE TO DEPOSIT THEM
   7776  100005  BINLDC, 0100005          /FUN1 FOR BIN LDR
   7777    1634          JMP START           /FOR OLD TIME'S SAKE
START=7634
*5760
/THIS CODING INSERTED HERE BY MONITOR
   5760       0  SYSTRT, 0             /STARTING ADDRESS OF USER PROGRAM
   5761 2001751  SYSTR1, JMS DISOLVE
   5762  170000          ZERA
   5763 1001760          JMP @ SYSTRT /START
   5764       0  DIRDEV, 0          /# OF DEVICE DIRECTORY IN CORE
```

```
5765 2001751  OVRLAY, JMS DISOLVE  /OVERLAY MONITOR
5766   30000           ONEA        /SIGNAL WRITE
5767 2001612           JMS DISK
5770       0  OARG1, 0  /INSERT FUNCTION WORD HERE
5771       0  OARG2, 0  /INSERT # OF WORDS
5772       0  OARG3, 0  /STARTING ADDRESS
5773 2111777           MEMA RUNFLG
5774  405160           EXCT ZAC      /SHOULD WE START PROGRAM
5775    1600           JMP HEAD      /NO, RESTART MONITOR
5776 1001760           JMP @ SYSTRT /YES, GO TO ADDRESS IN SYSTRT
5777       0  RUNFLG, 0
```

VII. LOADING THE DISK

The following list of commands shows most Nicolet Applications and Support programs, and their core addresses. This list assumes that the disk has just been initialized for the first time.

*STORE SYSGEN 0-7577;0 :P	(Place FT-Nmr tape in reader)
*BINLDR	(Place the FT-Nmr DEMON Patch tape in the reader, depress Continue and press Execute. Start at 7600 when tape finishes reading in)
*STORE FTNMR 0-7577;0 :P	(Place NMRCAL tape in reader)
*BINLDR	(Place the NMRCAL DEMON patch tape in the reader, depress Continue, and press Execute. Start at 7600 when tape finishes reading in)
*STORE NMRCAL 0-7577;0 :P	(Place Assembler tape in reader)
*BINLDR	(Start at 7600 when tape finishes reading in)
*STORE ASMBLR 2000-5777;2000	(Place Nicobug II tape in reader)
*BINLDR	(Start at 7600 when tape finishes reading in)
*STORE NICBUG 4000-5777;4000	(Set buttons to 6K Starting and 1K Size)
*LOAD NICBUG:B	(This loads Nicobug in new location for debugging all of 4K program memory)

*STORE HINBUG 114000-115777; 114700 (This creates a separate copy which will load into third stack)

Similarly, the following commands will store other application programs.

*STORE LOPUN 0-252;0	(Low Binary Punch)
*STORE HIPUN 110000-110252; 110000	(High Binary Punch)
*STORE OCTPRN 0-176;0	(Octal Print)

VIII. CONVERTING FROM DEMON TO DEMON/II

The conversion of existing disk cartridges to DEMON/II is extremely easy, as the file structure is almost exactly the same. To initialize the disk, simply load the DEMON/II tape, start it at location 0, and then depress Continue and press Execute. The program will initialize the disk and start DEMON/II. Then store the DEMON/II system generator by typing

```
STORE SYSGEN 0-7577;0 :P
```

and answer DELETE: by Y. Then, other disks may be initialized by typing

```
RUN SYSGEN
```

The computer will halt, allowing you to change disk cartridges and then start the generator program by depressing Continue and pressing Execute.

Files stored on the disks can be used without modification in the usual way. However, if you wish to take advantage of the RUN command, you must restore such files giving them the appropriate starting address. DEMON/II will consider all files stored by DEMON/I to have a starting address of 0. This could be disastrous, especially in the case of programs starting at, say, 2000, or in the case of data files which have no starting address, and are usually given the starting address 7600 by DEMON/II.

For example, to restore the Assembler-Editor using DEMON/II the following commands are needed:

```
LOAD ASMBLR
STORE ASMBLR 2000-5777 ;2000 :P
DELETE: Y
```

This gives the assembler its proper starting address of 2000.

Under some circumstances DEMON/II may assign the length of the last file on the disk to contain all remaining tracks when converting to DEMON/II. This can be easily corrected by loading and restoring this last file.

IX. COPYING FILES BETWEEN DISK CARTRIDGES

The disk monitor seems to be invisible when used as described in this manual, but of course it actually occupies a great deal of memory and is swapped in and out when needed. When transferring a file from one disk to another, it is important to have an appreciation of these facts or the data transferred may contain part of DEMON/II instead of all program.

Whenever the disk monitor is called either from a program or by starting the computer at 7600_8, the following sequence of events takes place.

(a) locations 3000-5777 are written onto track 1
(b) locations 6000-7577 are written onto track 2
(c) the disk directory is loaded into 3000-5777
(d) DEMON is loaded into 6000-7577

We will refer to tracks 1 and 2 as the swapping area, since they contain program information saved from memory. Whenever the program is restarted, using a GO command the swapping area is written back into memory and the program started.

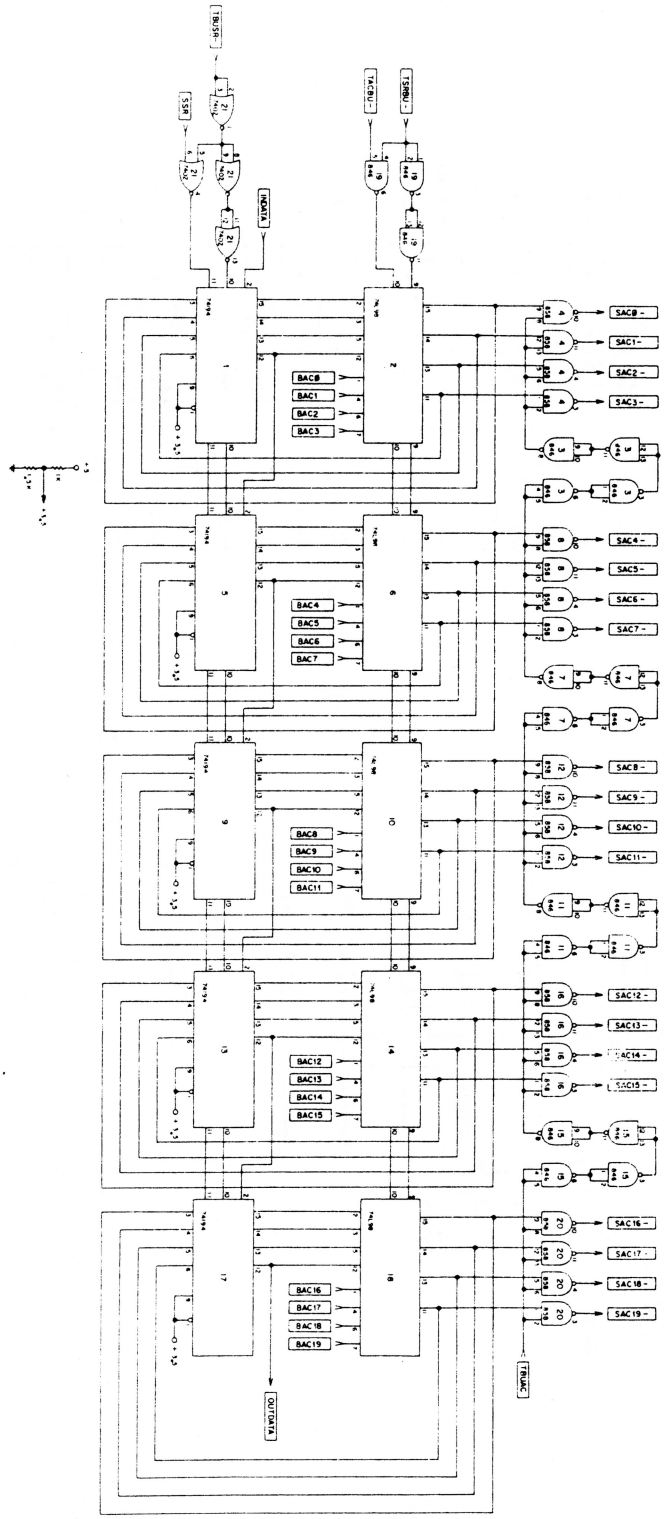
If a LOAD (or STORE) command is given the following set of operations takes place:

(a) the swapping area is read back into 3000-7577
(b) the new information is read in from disk (or written onto the disk)
(c) the new contents of 3000-7577 are written back into the swapping area
(d) the monitor and directory are read back in and restarted

Thus, while the just loaded program is now ready to be put in memory by a GO command, the program is partially contained on two disk tracks and not in memory. Thus, if one were to change disk cartridges at this point and try to store the program, he would store the wrong swapping area.

Therefore, the only safe way to transfer a program between disks is to LOAD it and give the GO command, or give the RUN command. Then change disk cartridges and restart the monitor, putting the program into the swapping area for storage. If the program is to be stored without being started, RUN some other program which can be co-resident, such as HINBUG.

These comments, of course, do not apply to programs which do not occupy locations 3000-7577, and they can be transferred in the naive manner first described.



REVISIONS

SYM	ZONE	DESCRIPTION	DATE	APPROVED

NICOLET INSTRUMENT CORPORATION

TITLE: SHIFT REGISTER AND BUFFER REGIST.
MODEL 294 DISK INTERFACE

CODE IDENT NO | SIZE D | DWG NO (BOARD 7) | SHEET

SCALE

NEXT ASSY | USED ON
APPLICATION

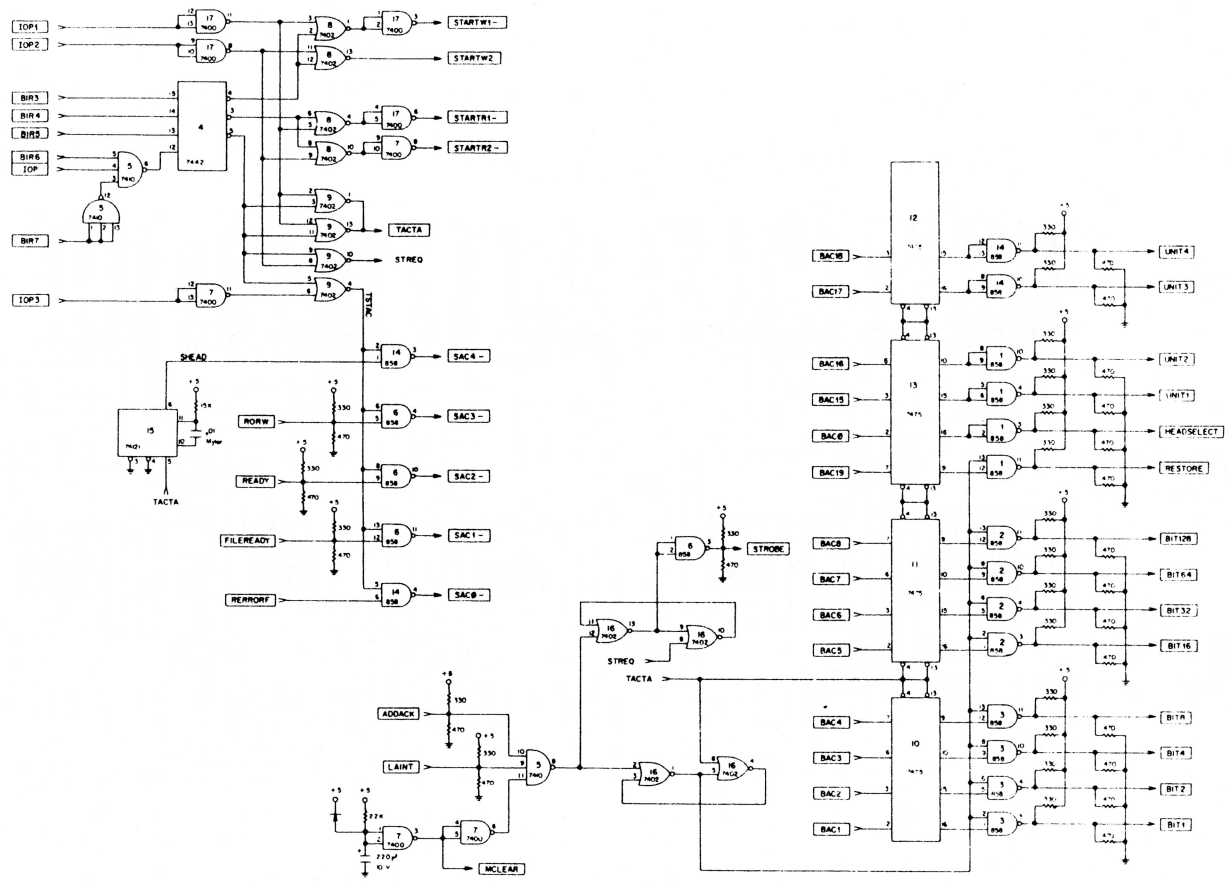
FINISH | MATERIAL | SPEC

REMOVE ALL BURRS
BREAK ALL SHARP CORNERS

DIST

KEY

REVISIONS				
SYM	ZONE	DESCRIPTION	DATE	APPROVED



DIST KEY

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES AND TOLERANCES ARE FRACTIONS DECIMALS ANGLES ± INDICATES SURFACE ROUGHNESS PER MIL STD-10

MATERIAL

SPEC

FINISH

SPEC

REMOVE ALL BURRS BREAK ALL SHARP CORNERS

DRAWN DATE

CHECKED

APPROVED

APPROVED

NEXT ASSY | USED ON

APPLICATION

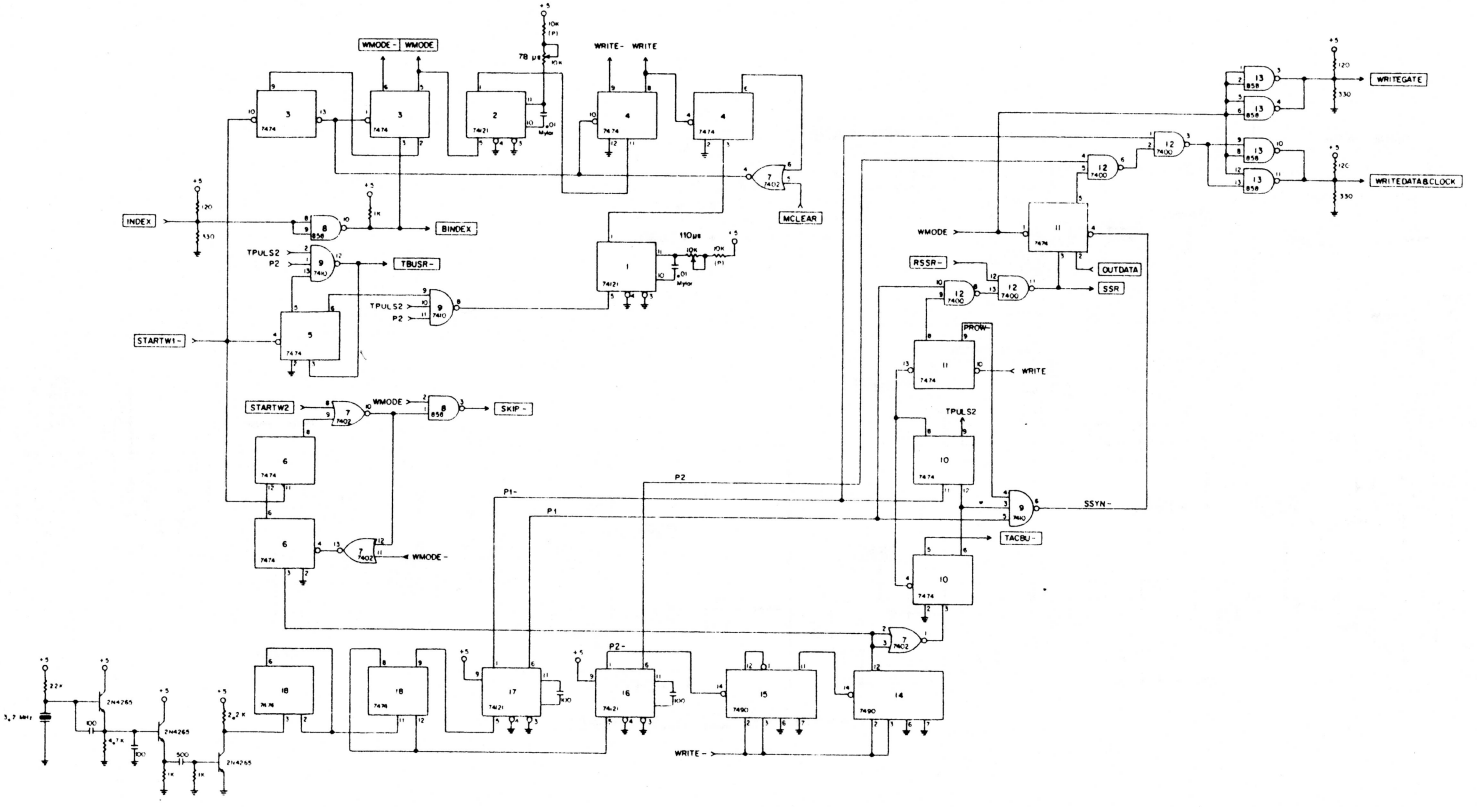
NICOLET INSTRUMENT CORPORATION ATED

TITLE HEAD POSITIONING LOGIC
MODEL 294 DISK INTERFACE

CODE IDENT NO | SIZE D | DWG NO (BOARD B)

SCALE | SHEET

REVISIONS				
SYM	ZONE	DESCRIPTION	DATE	APPROVED



NEXT ASS'Y	USED ON		
APPLICATION			

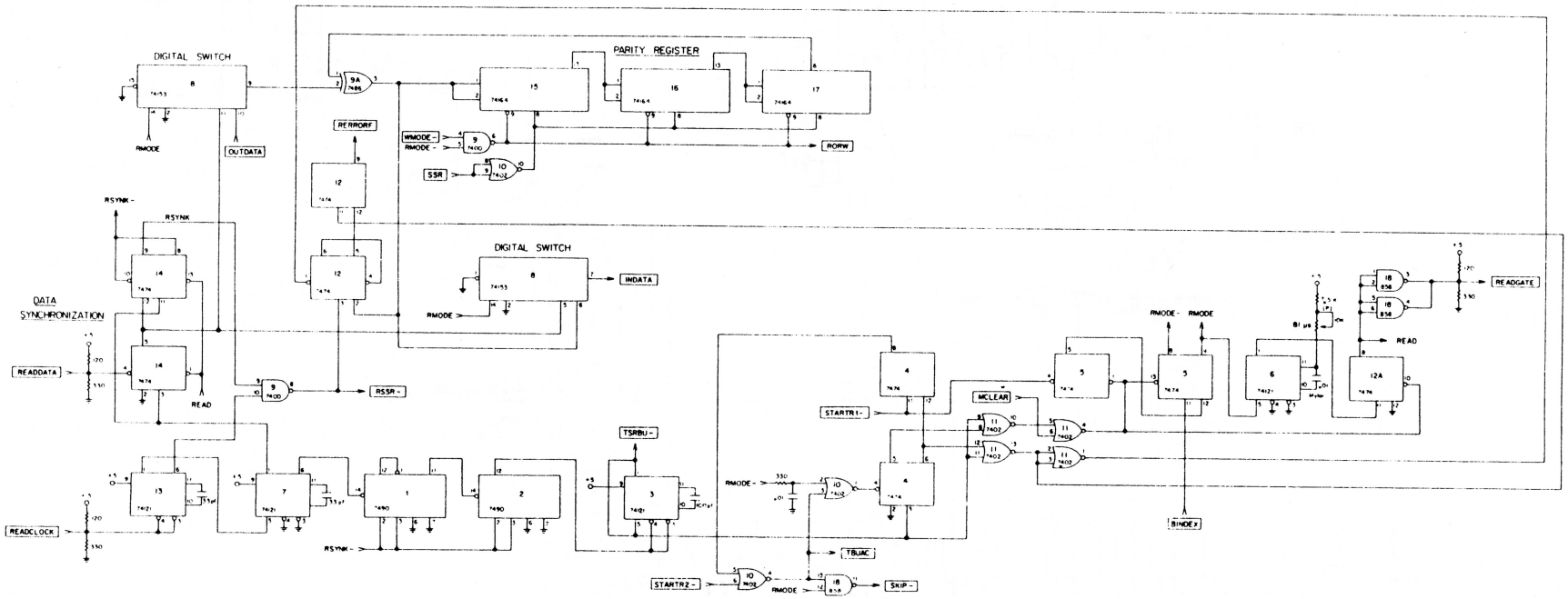
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES AND TOLERANCES ARE FRACTIONS DECIMALS ANGLES ± ± ±
V INDICATES SURFACE ROUGHNESS PER MIL STD-10
MATERIAL
SPEC
FINISH
SPEC

REMOVE ALL BURRS
BREAK ALL SHARP CORNERS
DRAWN DATE
CHECKED
APPROVED
APPROVED

NICOLET INSTRUMENT CORPORATION
TITLE: WRITE CONTROL LOGIC
MODEL 294 DISK INTERFACE
CODE IDENT NO.
SIZE D
DWG NO (BOARD 9)
SCALE
SHEET

DIST KEY

REVISIONS			
SYM	ZONE	DESCRIPTION	DATE	APPROVED



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES AND TOLERANCES ARE FRACTIONS DECIMALS ANGLES ± ± ±	REMOVE ALL BURRS BREAK ALL SHARP CORNERS	NICOLET INSTRUMENT CORPORATION	DIST KEY
√ INDICATES SURFACE ROUGHNESS PER MIL STD 10	DRAWN DATE	TITLE READ CONTROL LOGIC MODEL 294 DISK INTERFACE	
MATERIAL	CHECKED		
SPEC	APPROVED		
FINISH	APPROVED		
SPEC		CODE IDENT NO	SIZE D
NEXT ASS'Y / USED ON APPLICATION		SCALE	DWG NO (BOARD 10) SHEET

INDEX