NorthStar™

Hard Disk Operating System

02608A

To Modify HDOS + Resave

① LF HDxxDOS,1 5100
DS 5134 → To change Disk DRIVE IDENT,
ØØ FOR 2 DD SS Drives
FF FOR QUAD DRIVES

② SF HDxxDOS,1 5100

NorthStar™

North Star Computers, Inc.
14440 Catalina St., San Leandro, CA 94577 USA
(415) 357-8500 TWX/Telex (910) 366-7001

Hard Disk Operating System Manual

Horizon is a registered trademark of North Star Computers, Inc.
CP/M is a registered trademark of Digital Research, Inc.

TABLE OF CONTENTS

APPENDICES

FIGURES AND TABLES

FIGURES

2 Using the Software

4 BACKUP AND RECOVERY

TABLE

You are now licensed to use the Hard Disk Operating System (HDOS) from North Star Computers, Inc. HDOS was developed by North Star to extend the capabilities of the Disk Operating System (DOS) to work with the expanded capacity of North Star hard disks.

The material in this manual is arranged in sections. Section 1 contains procedures for an initial installation of your software. Sections 2 and 3 cover the normal operation of HDOS. Section 4 details the BACKUP and RECOVER process. Sections 5 and 6 are useful to an assembly language programmer.

The appendices contain reference material for the manual.

Every effort has been made to ensure the accuracy of the material presented here. Nevertheless, experience shows that some textual errors always go undetected. If you find any errors, or have some suggestions on how to improve this manual, please contact North Star at the following address:

NORTH STAR COMPUTERS
ATTN: TECHNICAL PUBLICATIONS
14440 CATALINA STREET
SAN LEANDRO, CA 94577 USA

1.1 HARD DISK OPERATING SYSTEM SOFTWARE

The North Star Hard Disk Operating System (HDOS) is supplied to you on two North Star floppy disks, the HDOS 2.1.x SYSTEM DISK and the HDOS INITIAL RECOVERY DISK. The software includes:

The Hard Disk Operating System (HDOS). The operating system includes the hard disk File Manager, the floppy disk drive control routines, and the standard peripheral input-output device drivers.

The Command Processor. This program accepts commands from a terminal to manipulate disk files, accounts, programs, and RAM, and to perform miscellaneous monitoring functions.

Hard Disk BASIC (HBASIC). This version of North Star BASIC allows access to files on the hard disk as well as floppy disks with little or no change to existing BASIC programs.

The BACKUP and RECOVER Programs. The programs allow convenient backup and retrieval of files stored on the hard disk drive. Using the complete and incremental data backup program protects your data in the event of power failure, hardware failure, or operator error.

You can also use floppy disk backup to preserve original data before performing major file updates on the hard disk.

The Hard Disk Test Program. This program permits testing and formatting of the hard disk drive(s) when the system is initially set up, during total system recovery, or during daily preventative maintenance.

1.2 INITIAL SYSTEM STARTUP

Once the hardware has been set up and successfully powered-on, the procedures described below must be followed to complete the initial system software installation.

NOTE
If you are running an earlier version of HDOS, skip these procedures and go to Section 1.3, "Upgrading HDOS to HDOS 2.1.x".

First test and format the hard disk with Level Two of the Hard Disk Test Program. Then, perform an initial system recovery to install the system software on the hard disk. Finally make working copies of the floppy disks for everyday use and retire the factory-supplied floppy disks to safe storage.

Normally, the complete process is performed only once, before the hard disk system is used for the first time.

1.2.1 INSERTING FLOPPY DISKS

Insert the HDOS 2.1.x SYSTEM DISK into floppy disk drive 1 (the drive closest to the center of the Horizon's front panel) with the oblong hole entering the slot first and the floppy disk's label facing away from the drive's LED indicator.

Carefully close the door on the drive. If the door does not "lock" into the closed position, re-insert the disk and try to close the door again. Never force the door shut, as this may damage the disk.

1.2.2 INITIALIZING HDOS

Press and release the red reset switch next to the cooling fan on the HORIZON's rear panel. Drive 1 should turn on (the LED indicator on the drive will light up), and this system message will appear on your terminal:

```
-----------------------------------------------------------------
                    HDOS Initial Boot Procedure

This disk supplied from North Star contains two different HDOS
operating systems - one for 5 inch hard disks and one for the HD-18
hard disk. The names of the files are HD5XDOS and HD18DOS,
respectively.

To manually boot into the correct operating system for your disk,
type:

GO HD5XDOS,1 <cr> (If you have a 5 inch hard disk)
                          or
GO HD18DOS,1 <cr> (If you have an HD-18 hard disk)

After you have done this, you can follow the instructions in the
Hard Disk Operating System User Manual, under the heading Initial
System Startup to prepare the hard disk and an automatic bootstrap
disk.
+
-----------------------------------------------------------------
```

When you enter the proper GO command after the "+" prompt, the HDOS sign-on message and command prompt "=" will appear on your terminal screen. If you have an HD18, you will hear the hard disk's motor start running.

1.2.3 STARTUP PROCEDURE FOR SINGLE-SIDED DRIVES

The HDOS is initially configured to operate with two-sided, fast-stepping (quad) floppy disk drives. On startup, if you have any single-sided, normal-stepping (double density) floppy disk drives, enter the following in response to the HDOS prompt (=):

FM 134 0 [RETURN]

This temporarily tells your system to use single sided drives.

NOTE
If you have a mix of single and double-sided drives, you can use the SYSGEN program later to determine the proper configuration byte for your system.

1.2.4 REFORMATTING THE HARD DISK

The next step is to test and reformat the hard disk. See Section 1.7.3 for a discussion of "skips" and the "skip table".

CAUTION
This writes over any previous hard disk data. Use this procedure for an initial setup or on a completely backed up disk. To upgrade earlier versions of HDOS, see Section 1.4, "Upgrading HDOS to HDOS 2.1.x".

Procedure

STEP ACTION

1 Check that the HDOS 2.1.x SYSTEM DISK is in floppy disk drive 1.

2 IF...

o you have a five inch hard disk then enter **GO HD5XTEST,1 [RETURN]** and go to step 3.

o you have an HD18 hard disk then enter **GO HD18TEST,1 [RETURN]** and go to step 4.

3 Your Hard Disk code will be displayed on your terminal. You can verify the code by checking the hard disk label on the rear of the computer. If it is correct, enter **Y**. If it is not correct, enter **N** and the correct code.

4

```
------------------------------------------------
WHICH DIAGNOSTIC LEVEL TO EXECUTE:

(1)  NONDESTRUCTIVE DAILY RUN

(2)  SIMPLIFIED TOTAL DISK CHECK AND REFORMAT
     (DESTRUCTIVE TO ALL DATA!)

(3)  EXTENDED TOTAL DISK CHECK AND REFORMAT
     (DESTRUCTIVE TO ALL DATA!)
------------------------------------------------
```

Procedure (continued)

STEP ACTION

5 **2**

6 IF...

- you have an HD18, you are prompted for the hard disk unit number. Enter **unit # [RETURN].**

```
---------------------------------------------------
**** WARNING ****

PROCEEDING WITH THIS TEST WILL DESTROY ALL EXISTING
DATA ON THE DISK

HIT <RETURN> TO PROCEED OR <ESC> TO ABORT
---------------------------------------------------
```

7 **[RETURN]**

8 When the test is complete the program displays the message:

```
------------------
TEST COMPLETE

POWER DOWN: (Y/N)
------------------
```

9 **N**

The terminal displays:

```
----------------------
HIT <RETURN> TO REBOOT
----------------------
```

10 **[RETURN]**

The program displays the HDOS command prompt.

```
----
=
----
```

1.2.5 INITIAL SYSTEM SOFTWARE RECOVERY

The next step in the initial startup procedure is to create the directory and install the system software on the hard disk. To do this, you will use the TOTREC program and the HDOS INITIAL RECOVERY DISK to install the basic system software in your hard disk SYSTEM account.

Procedure

STEP ACTION

---

1 Ensure that the HDOS 2.1.x SYSTEM DISK is loaded in floppy disk drive 1.

2 IF...

- o you have a five inch hard disk unit, enter:
  GO HD5XDOS [RETURN]

- o you have an HD18 hard disk unit, enter:
  GO HD18DOS [RETURN]

The terminal will display the HDOS command prompt.

```
----
=
----
```

3 IF...

- o you have any single-sided, normal-stepping (double density) floppy disk drives, then enter:
  FM 134 0 [RETURN]

4 **GO TOTREC,1 [RETURN]**

The program informs you that proceeding with TOTREC erases all files and accounts on the hard disk, and questions whether this is in fact what you want to do.

5 **YES [RETURN]**

The program prompts for a hard disk drive number. (Enter drive number 101 to 104. For one hard disk, enter 101.)

Procedure (continued)

STEP ACTION

---

6 **101 [RETURN]**

The program sends a list of messages to your terminal. When you receive the message:

```
-----------------------
Initialization complete
-----------------------
```

and then prompts for a listing destination for the recovered files.

7 **Option #**

The program prompts for the "Master backup disk" drive number.

8 IF...

- o you have a system with one floppy disk drive, remove the HDOS 2.1.x SYSTEM DISK from floppy disk drive 1, insert the HDOS INITIAL RECOVERY DISK in floppy disk drive 1 and enter 1 **[RETURN]**
- o you have more than one floppy disk drive, put the HDOS INITIAL RECOVERY DISK in floppy disk drive 2 and enter **2 [RETURN]**

```
------------------------
1.  Recover all accounts.

2.  Specify accounts.

3.  Specify exceptions.
------------------------
```

9 **1**

The program displays a message similar to:

```
-----------------------------------------------------
Allocated space for file TRANSIENT, SYSTEM: 50 BLOCKS
.
Allocated space for HBASIC, SYSTEM:         60 BLOCKS
-----------------------------------------------------
```

Procedure (continued)

STEP ACTION

---

10 IF...

- o you have one floppy disk drive, re-insert the HDOS 2.1.x SYSTEM DISK into floppy disk drive 1 when requested and **[RETURN]**.

11 The program creates a SYSTEM account and file directory on the hard disk and allocates disk space for your initial software files.

12 IF...

- o you are using one floppy disk drive, the program prompts for the "Master Backup Disk". Insert the HDOS INITIAL RECOVERY DISK into floppy disk drive 1 again.

At this point the data for each file is copied to the hard disk.

13 **Floppy disk drive #**

The program creates hard disk SYSTEM files, using the files from the HDOS INITIAL RECOVERY DISK. For each file recovered to the SYSTEM account, a dot (.) appears on your screen.

In addition to the backup data you have just used, the HDOS INITIAL RECOVERY DISK also contains several additional files which you may wish to copy onto your hard disk. If you do, use the CF (Copy File) command to copy the files to hard disk. Use the LI (List File) command to check the filenames in the HDOS INITIAL RECOVERY DISK directory.

The files REDIRECT and CP/M.FIX will be useful if you have been using HDOS Revision 1.B on an HD18. If this is the case, see section 1.3 below.

There are also various versions of HBASIC for hardware floating point and extended precision arithmetic. Before using extended precision HBASIC, you should consult the North Star BASIC manual.

1.2.6 INITIAL COMPLETE BACKUP

Now you should perform your first COMPLETE backup. There are two reasons for doing a backup at this time.

1. You always should have a copy of your HDOS INITIAL RECOVERY DISK to use if something happens to the factory supplied floppy disk that makes it unuseable.

2. The second reason is that now is the time for you to begin a formal backup procedure for your hard disk to ensure the best possible recovery situation if anything happens to your hard disk.

The procedure below is simplified and useful only for this portion of the Initial System Startup. Read Chapter 4, 'Backup and Recovery', for more information about the BACKUP and RECOVER process.

NOTE
You will need at least one blank floppy disk to run an initial complete backup.

Procedure

STEP ACTION

1

```
----
=
----
```

2 Insert a blank floppy disk into floppy disk drive 1.

3 **IN 1[RETURN]**

```
-------------
Are you sure?
-------------
```

4 **Y**

```
----
=
----
```

Procedure (continued)

STEP ACTION

---

5 **GO BACKUP [RETURN]**

```
---------------------------------------
1.  Complete backup.

2.  Incremental backup.

3.  Selected files or accounts backup.

4.  Explanation.
---------------------------------------
```

6 **1**

The program prompts for the date. (Do not use blanks.)

7 **Date [RETURN]**

The program prompts for a listing device. If you do not want a printed copy, enter **0** to send the list to yo terminal.

8 **Listing device number**

Note that selection '3' will cause the program to prompt for a printer device number. Next, the program prompts for the hard disk drive number.

9 **101 [RETURN]**

Now the program prompts for the floppy disk drive number into which you have put the blank disk.

Procedure (continued)

STEP ACTION

10 **Floppy disk drive #**

The program indicates 'BACKUP STARTED'. The hard disk directory is compressed and written to the Master disk.

As each file is copied, the filename and length is displayed. The heading BACKED UP shows how much of the file or account fit on the floppy disk.

```
--------------------------------------------------------
ACCOUNT         NAME                 SIZE       BACKED UP
***
SYSTEM          BACKEXP                 6               6
*********************************************************
SYSTEM          HBASIC                 58              58
*********************************************************
.
.
SYSTEM          RECOVERS               46              46

Please remove BACKUP.M from drive 1 and label it.

File data backup complete.

File cleanup started.
--------------------------------------------------------
```

A dot (.) is displayed on your screen as each file is cleaned.

11 Since this is the first floppy disk of the session, it now contains the disk directory and is labeled "BACKUP.M" by the system.

You should label the disk 'BACKUP.M' and date it.

12 When the program ends **[RETURN]** to HDOS command level.

1.2.7 COPY THE SYSTEM DISK

To create an "auto-boot" floppy disk to automatically initialize HDOS you must first copy the HDOS 2.1.x SYSTEM DISK onto a working disk. A working disk is a copy of the factory-supplied floppy disk that will be used daily, while the original is stored for safe keeping. If the working disk is damaged or destroyed, another copy can be made from the original. Only use factory supplied HDOS floppy disks for the initial start-up and copy.

Procedure

STEP ACTION

1

```
----
=
----
```

2 Insert the HDOS 2.1.x SYSTEM DISK into floppy disk drive 1.

3 **CF ,1 CR IMAGE [RETURN]**

on 15 Meg Dr Copy of HDOS 2.2.
is Called IMAGEHD220

Copy the floppy disk to the hard disk default account SYSTEM with a filename of IMAGE.

After approximately 30 seconds the red drive indicator light turns on. The total copy takes about two minutes, then displays:

```
--------------
COPY COMPLETED
--------------
```

4 Remove the HDOS 2.1.x SYSTEM DISK from disk drive 1.

5 Insert a blank floppy disk into disk drive 1.

6 **CF IMAGE TO ,1 [RETURN]**

Copy IMAGE from hard disk to the blank floppy disk in drive 1. This creates a working copy of the HDOS 2.1.x SYSTEM DISK. Repeat this command for more copies.

1.2.8 USING SYSGEN

You can use the SYSGEN program at this point to configure your working copy of the HDOS 2.1.x SYSTEM DISK to:

o 'auto-start' your copy of the HDOS,
o set your screen length,
o enable or disable interrupts,
o and/or auto-start an application such as HBASIC.

Procedure

STEP ACTION

---

1 Ensure that the working copy of the HDOS 2.1.x SYSTEM DISK is in floppy disk drive 1.

2 **GO HBASIC [RETURN]**

```
READY
```

3 **LOAD SYSGEN,1 [RETURN]**

```
READY
```

4 **RUN [RETURN]**

The SYSGEN program displays an initial message and the main menu:

```
------------------------
Configuration options:

D)os
B)asic
E)nd configuration
------------------------
```

5 **D [RETURN]**

```
-----------------------------------
Which DOS do you want to configure:

H)ard disk HDOS
F)loppy disk DOS
E)xit to main menu
-----------------------------------
```

Procedure (continued)

STEP ACTION

6 **H [RETURN]**

```
------------------------------------
Which HDOS do you wish to configure:

A) any 5 inch hard disk
B) an HD-18 hard disk
------------------------------------
```

7 IF...

- you have a five inch hard disk, enter **A [RETURN]**
- you have an HD18 hard disk, enter **B [RETURN]**

```
---------------------------------------------
Is HDxxDOS,1 the desired file to be configured?

Y)es, use this name
N)o, fetch alternate name
E)xit to main menu
---------------------------------------------
```

8 **Y [RETURN]**

```
-----------------------------------------
When you boot from this disk, do you want
HDxxDOS,1 to automatically begin execution?
-----------------------------------------
```

9 **Y [RETURN]**

The program now prompts for the number of lines you want to appear on your terminal. This number is usually 24.

10 **Number of lines [RETURN]**

The program prompts for your floppy drive type:

```
----------------------------------------------
Q) double sided (quad capacity), fast stepping
D) single sided (double density)

What type of floppy disk drive is on your system?
[Q, D, or M)ixed]:
-----------------------------------------------
```

Procedure (continued)

STEP ACTION

---

11 IF...

- o you have only double sided quad capacity drives, enter: **Q [RETURN]**
- o you have single sided double density drive(s), enter: **D [RETURN]**

The program displays the highest HBASIC MEMSET for your system, then prompts:

```
------------------------------
Press any key to continue ...
------------------------------
```

12 **Any key**

You now have the option of enabling or disabling interrupts:

```
----------------------------------------------
Run with interrupts E)nabled or D)isabled --
----------------------------------------------
```

13 IF...

- o you will be running any North Star multi-user operating system, such as TSS/A or TSS/C enter **D [RETURN]** to disable interrupts

NOTE
See the section titled "Configuring the HORIZON for Multi-user Operation" in the TSS/A and TSS/C manuals for more information on interrupt handling and its relationship to hardware.

- o you wish to enable interrupts, enter **E [RETURN]**

Procedure (continued)

STEP ACTION

14 SYSGEN presents the option of automatically starting ar application program.

```
---------------------------------------------
HDxxDOS,1 can be configured to automatically
start a program.

It is NOT currently set to do this.

Do you wish to change this?
---------------------------------------------
```

15 IF...

- o you would like to automatically enter a program such a HBASIC every time you boot up your system disk, ente **Y [RETURN]**. SYSGEN prompts for the new auto-star command. The command should be in the form: 'G xxxxx', such as **GO HBASIC.**

- o you do not want to automatically enter a program, ente **N [RETURN]**.

```
-----------------------------------------------
Press any key to return to the main menu . . .
-----------------------------------------------
```

16 **Any key**

```
------------------
D)os
B)asic
E)nd configuration
------------------
```

17 **E [RETURN]**

```
---------------------------------------------------------
All changes are complete and the disk may be removed.
Thank you.

READY
---------------------------------------------------------
```

Procedure (continued)

STEP . ACTION

18 **BYE [RETURN]**

```
-----------------------------------------------------
North Star Hard Disk Operating System, Version 2.1.x
=
-----------------------------------------------------
```

1.3 UPGRADING HDOS TO HDOS 2.1.X

If you are running earlier versions of HDOS and want to update to HDOS 2.1.x you will need to upgrade some files on your hard disk to make this possible.

NOTE

If you have been running CP/M on an HD-18, you should copy the correct HDBOOT file to your CP/M system disk. You must do this before you rearrange the hard disk. (You may have already done this if you are already running HDOS 2.0.0.)

1. Connect to the HDOS system file CP/M.FIX by specifying it as a CP/M volume (see the North Star CP/M Preface). If you do not have CP/M.FIX on your hard disk, use the CF command to copy it from the HDOS 2.1.X INITIAL RECOVERY DISK.

2. PIP the correct HDxxBOOT.COM file from CP/M.FIX to your working copy of the CP/M system disk. (The correct HDBOOT name will depend on the type of hard disk). This allows CP/M to find HDOS files that are volumes under the rearranged disk.

3. DElete CP/M.FIX from your SYSTEM account.

Several files need updating. To do this, the old versions must be deleted. There is a program on the HDOS 2.1.x SYSTEM DISK that will automatically delete the correct files. The program name is SHORTCUT. Enter:

GO SHORTCUT,1 [RETURN]

A sequence of commands appear on you terminal, deleting several files. Some of the file names may not appear on your hard disk, but will be deleted anyway.

When this program has finished execution and you have the HDOS '=' prompt on your screen, run the program TOTREC, with one crucial difference from a Total Recovery. Enter:

GO TOTREC,1 [RETURN]

and in response to the first question that asks whether you want to delete all files and accounts on the hard disk, answer NO.

If you don't answer NO, all your data will be destroyed!

Answering NO leaves the hard disk directory unchanged and creates those files found on the HDOS INITIAL RECOVERY DISK.

Follow the rest of the instructions as per a normal TOTREC. (Refer to Section 4.3.3, Using TOTREC.)

If you have been using HDOS Revision 1.B on an HD18 hard disk, you can now rearrange the hard disk directory to make directory accesses faster.

Enter:

GO REDIRECT,1 [RETURN]

1.4 NORMAL SYSTEM STARTUP

All the programs necessary for each startup of the hard disk system should now be on your working copy of the HDOS 2.1.x SYSTEM DISK.

This disk should always be inserted into floppy disk drive 1, the drive nearest to the center of the HORIZON, to load the HDOS into the computer's memory (RAM).

When the computer is first powered up, you should press and release the red reset switch on the HORIZON's rear panel. This starts the "bootstrap" program which in turn activates the floppy disk drive and loads the HDOS into the computer's memory. The HDOS command prompt ("=") appears on the terminal screen each time HDOS is successfully loaded, unless you have configured a special auto-start.

After the computer has been powered on, whenever you want to re-boot the system from HDOS you should use the HDOS IL (Initial Load) command.

If you have an HD18, after the hard disk drive motor has started allow three minutes before any operation. The internal control system of the HD18 imposes this delay while the drive motor comes to full speed and stabilizes.

Five inch hard disks have no significant delay.

1.5 TURNING OFF THE SYSTEM

Before turning off the power to the system, remove any disks from the floppy disk drives.

If you have a system with one or more HD18 hard disks, you should use the HDOS OF command to turn off the motors. The OF command retracts the read/write heads to their special landing zones on the disk before stopping the drive motor. You can turn off the power to the computer, terminal, hard disk drives, and peripherals, in any order.

Although a North Star HORIZON with a five inch hard disk does not require you to enter the OF command when you power down the computer, there is a preferred landing zone for each type. We recommend using OF, especially when you move the machine.

1.6 HARD DISK TEST PROGRAM

The Hard Disk Test Program (HD5XTEST or HD18TEST) is a three level diagnostic test program for detecting potential hardware problems in North Star hard disk units.

The diagnostic program is contained on the factory supplied HDOS 2.1.x SYSTEM DISK.

The Level One test performs a non-data-destructive scan of the key signals and data on the disk. Run this test daily to provide early warning of possible disk problems. If the test detects no errors, assume the disk is functioning correctly and terminate the diagnostic program.

If the Level One test does detect a potential problem, the program indicates what steps to take. In most cases the program will advise the user to:

1. Perform a preventive maintenance procedure

or

2. Proceed to the Hard Disk Diagnostic Level Two test.

The Level Two test is run as above in section 1.2.4 "Reformatting the Hard Disk". Refer to the section on Advanced Diagnostics below for Level Three test procedures.

NOTE
Perform a Level Two or Level Three diagnostic test <u>only</u> if the data on hard disk has been completely backed up on another medium. Level Two and Level Three destroy all data on the hard disk.

All hardware modifications included in the HD18 Upgrade Kit should be installed before the Hard Disk Test Program is run for the first time.

1.6.1 THE DAILY RUN PROGRAM

An important responsibility of the user is routine testing of the computer. The Non-destructive Daily Run program performs this function. It will not affect the data on the disk.

Use this program on a regular basis. Once a week should be sufficient.

The Daily Run verifies the performance of the hard disk. It will indicate a problem, called a "read error," if any exists.

You should power up your HORIZON and all peripheral hardware, then follow this procedure:

Procedure

STEP ACTION

---

1 Insert your working copy of the HDOS 2.1.x SYSTEM DISK in floppy disk drive 1.

2 When the floppy disk is seated in the drive, close the drive latch.

3 Press and release the RESET switch on the back of your computer.

```
----
=
----
```

4 Enter the appropriate command to begin the test.

IF...

---

o you have a five inch hard disk, then enter:
GO HD5XTEST [RETURN]
and go to step 5.

o you have an HD18, then enter
GO HD18TEST [RETURN]
and go to step 6.

Procedure (continued)

STEP ACTION

---

5 Your Hard Disk code will be displayed.

Verify the code by checking the hard disk label on the rear of the computer. If it is correct enter **Y**. If it is not correct, reply **N** and enter the correct code.

6
```
---------------------------------------------
WHICH DIAGNOSTIC LEVEL TO EXECUTE:

(1)  NONDESTRUCTIVE DAILY RUN

(2)  SIMPLIFIED  TOTAL DISK CHECK AND REFORMAT
     (DESTRUCTIVE TO ALL DATA!)

(3)  EXTENDED TOTAL DISK CHECK AND REFORMAT
     (DESTRUCTIVE TO ALL DATA!)
---------------------------------------------
```

7 Select 1 from the main menu.

IF...

---

o you have an HD18 you are prompted for a unit number. Enter the **unit # [RETURN]**.

```
----------------------
Level 1 test...
HARD DISK UNIT TO CHECK
(101,102,103, or 104)
----------------------
```

8
```
---------------------------------------------------------
Sector pulse count correct
Sector pulse timing range correct
Testing usable tracks for read errors
---------------------------------------------------------
---------------------------------------------------------
Number of read errors: 0
Testing reserved track
-------------------------------------------------------
Reserved track test passed

TEST COMPLETE

Press RETURN
---------------------------------------------------------
```

Procedure (continued)

STEP ACTION

9 IF....

- o the number of read errors is 0 then **[RETURN]** to exit the program and return to the main menu.
- o the number of read errors is greater than 0, then note the number and call a North Star service representative

10 At this point the program begins the specified diagnostic routine. The program requests no further information for the daily Level One test.

1.7 ADVANCED DIAGNOSTICS

1.7.1 PROBLEMS WITH THE HARD DISK

For purposes of this discussion, problems with the hard disk have been divided into several broad categories:

- computer failure which affects the hard disk
- loss of the skip table on the hard disk not related to a computer failure
- loss of the hard disk label

The skip table is explained in the next section.

Here is a brief list of some of the situations which might indicate or result in a computer failure:

- the computer has been dropped or jolted
- you receive an error message when you try to boot up the computer
- the results of the Daily Run show a significant number of "read errors"
- there has been a series of power failures in your building

1.7.2 DIAGNOSING PROBLEMS

There is little you can do alone if your computer has failed. If you suspect a failure, you may do the Daily Run (Level 1 Test) to confirm hard disk errors. You should then call your North Star service representative.

The Simplified (Level 2) and Extended (Level 3) Check programs have more sophisticated tests than the Daily Run. But these will destroy your data. You should not run them unless your hard disk is empty or has been successfully backed up.

In general, then, your ability to diagnose hard disk failures is limited to errors detected on the Daily Run.

1.7.3 SKIPS

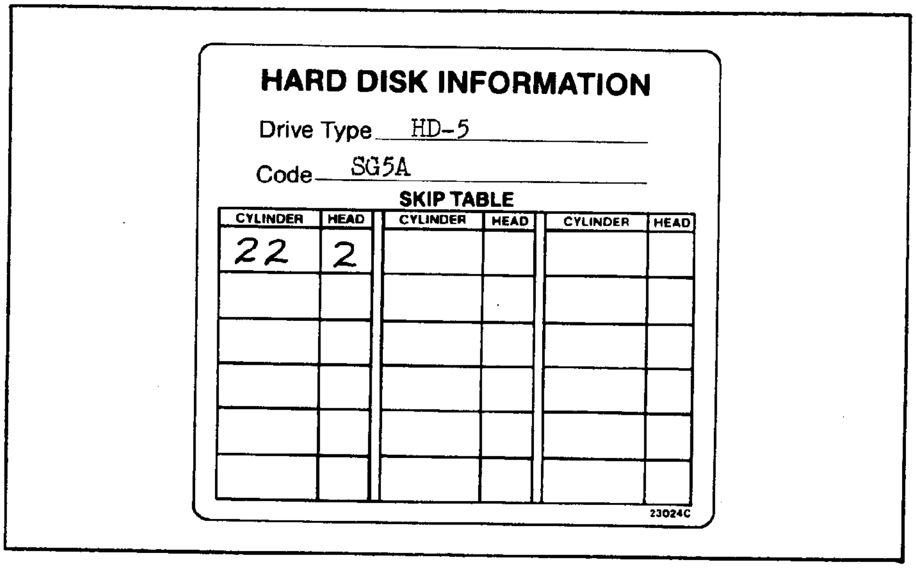
A skip is a portion of the hard disk that is not reliable. Normally, a hard disk will have a few skips. They do not mean that the disk itself is defective, nor do they measurably reduce the capacity of the disk.

The hard disk is divided into cylinders and heads. Skips are identified according to the cylinder and head on which they are located. The hard disk is carefully checked and all skips are identified before it is shipped. The locations of the skips are then recorded in two places:

o on a sticker on the back of the computer

o on a sticker on the outside of the hard disk

Diagram



HARD DISK INFORMATION

Drive Type HD-5

Code SG5A

SKIP TABLE

CYLINDER	HEAD	CYLINDER	HEAD	CYLINDER	HEAD
22	2				

23024C

1.7.4 THE SKIP TABLE

The skip table is the computer's internal information about skips, and is located on sector 2 of the hard disk. The hard disk must have this information to operate properly.

Skips themselves are not serious problems. However, if the skip table is somehow lost, the hard disk will not function properly. Here is a partial list of situations in which this might occur:

- o If the Level 2 or Level 3 hard disk test program is interrupted by a power failure or a system reset.
- o If the hard disk has been dropped or jolted during shipping or handling. The message "CURRENT SKIP TABLE INVALID" would appear on the screen when the Level 1 test is run.

If either of these situations arises with your hard disk, you will need to replace the skip table on the disk. You must:

- o Read the the information on the skip sticker about the cylinder and head location of each skip.
- o Run the Extended Check (Level 3) program to reformat and test the hard disk. The Extended Check program will require you to give the skip information from the sticker.

The procedure for running the Extended Check program is found in the next section.

1.7.5 THE EXTENDED CHECK PROGRAM

The Extended Total Disk Check and Reformat (Level 3) program is the most sophisticated of the Hard Disk Test programs. Its function is to place the skip table on the hard disk. It is intended primarily for North Star service representatives.

CAUTION

The Extended Check program is destructive to data. Run this test yourself only if there is no data on your hard disk or if you have completely backed up your hard disk. In all other situations, you should call a North Star service representative.

You may use this test if the skip table on your hard disk has somehow been lost.

The Extended Check program is essentially the same as the Simplified Check program, which you used in Section 1.2.4. The difference is that you must answer a series of questions regarding your disk before the test is run.

1.7.6 RUNNING THE EXTENDED CHECK PROGRAM

This program presents a series of options. Most of these are useful only for North Star service representatives. This procedure shows the simplest method for running the program.

Procedure

STEP ACTION

1 Select option **3** from the main menu.

IF...

o you have an HD18, you are prompted for the hard disk unit number. Enter **unit # [RETURN]**.

```
---------------------------------------------------------------
**** WARNING ****             Level 3 test...

PROCEEDING WITH THIS TEST WILL DESTROY ALL EXISTING DATA
ON THE DISK

Press RETURN to proceed or ESC to abort
---------------------------------------------------------------
```

2 **[RETURN]**

3 IF...

o you see this message, then check the hard disk information sticker and go to the next step.

```
----------------------------
CURRENT SKIP TABLE

CYLINDER xxx, HEAD xx
CYLINDER xxx, HEAD xx

ENTER ADDITIONAL SKIPS(Y/N)?
----------------------------
```

o you see this message, then check the hard disk information sticker and go to the next step.

```
----------------------------
SKIP TABLE INVALID -
STARTING WITH NO SKIPS

ENTER ADDITIONAL SKIPS (Y/N)?
----------------------------
```

Procedure (continued)

STEP ACTION

4 IF...

- o the sticker shows no skips, then **N** and go to step #8.
- o the sticker show skips, then go to the next step.
- o the sticker and the display do not agree, then reset the computer and call your North Star dealer.

5 **Y**

```
---------
CYLINDER:
---------
```

6 Enter the cylinder number of the first skip recorded on the sticker.

```
-----
HEAD:
-----
```

7 Enter the head number of the skip. Enter the same information for each skip recorded.

8 **[RETURN]** after the next CYLINDER prompt when you have finished entering the skips.

9 The skip table is complete.

```
-----------------------------------------------
Press RETURN to accept, ESC to reject skip table?
-----------------------------------------------
```

10 **[RETURN]**

```
----------------------------
HALT IF ERROR DETECTED (Y/N)?
----------------------------
```

11 **N**

```
------------------------------
REPEAT TEST CONTINUOUSLY (Y/N)?
------------------------------
```

Procedure (continued)

STEP ACTION

12 **N**

```
------------------------
RUN TEST ON SKIPS (Y/N)?
------------------------
```

13 **Y**

```
---------------------------------------------------
TYPE THE NUMBER OF ITERATIONS FOR EACH TEST SECTION

PATTERN READ/WRITE:
---------------------------------------------------
```

14 **1 [RETURN]**

```
--------------------
SERVO HARMONIC TEST:
--------------------
```

15 **1 [RETURN]**

```
------------------
SERVO RANDOM TEST:
------------------
```

16 **1 [RETURN]**

```
--------------------------------------
OUTPUT TO TERMINAL (0) OR PRINTER (1)?
--------------------------------------
```

17 IF...

- o you want a printed record of the backup session, then select **1.**
- o you do not want a printed record, then select **0.**

18 You are ready to begin the test.

```
---------------------------
Press RETURN to start test:
---------------------------
```

Procedure (continued)

STEP ACTION

---

19 The test begins after the formatting is complete. This is a long test. You may want to do other work while it it running.

[RETURN]

```
-------------------------------------------------------
CONTROL-C CAN BE USED TO INTERRUPT TEST
(EXCEPT WHILE FORMATTING)
DISK NOW BEING FORMATTED
.......................................................
.......................................................
-------------------------------------------------------
```

20 Any errors detected will be recorded at the end of the test. These errors are rare. A small number of them is acceptable. If your disk has more than this, call your North Star dealer or service representative.

```
--------------------
SEEK ERRORS        0
HDCOM ERRORS       0
.
.
--------------------
```

21 Any skips will also be recorded.

```
-----------------------
CYLINDERS  xxx, HEAD  x
CYLINDERS  xxx, HEAD  x

TEST COMPLETE

Press RETURN
-----------------------
```

22 A **[RETURN]** will reboot HDOS.

USING THE SOFTWARE 2

2.1 OVERVIEW

The Hard Disk Operating System (HDOS) enables you to communicate with and control your floppy disk and hard disk drives. The HDOS programs reside on a floppy disk, which you must insert into a floppy disk drive and load into the computer's RAM memory each time you turn on the computer. Once the HDOS is loaded, you can enter HDOS commands to create and manipulate files, perform maintenance and debugging functions, or execute programs. You can program in assembly language, HBASIC, or any other language implemented by North Star.

2.2 THE LINE EDITOR

Before you attempt to enter commands or data on the keyboard, you should know how to use the line editor. Not only does the line editor send lines of input to the system, it enables you to correct typographical errors. The editing features described here work at the HDOS command level. Additional editing features are available when you invoke various programming environments. Those features are described in the manuals that accompany the programming languages.

2.2.1 Sending a Line to the System

A line typed in response to the HDOS prompt (=) is sent to the system by "typing a carriage return." Type a carriage return by pressing the RETURN key. Carriage returns are indicated in this manual by the symbol <CR>. Whenever a line of typing is shown followed by a <CR>, the operator should press RETURN.

You cannot correct an error in a line after you have pressed the RETURN key. If a command is unacceptable to the system, the system produces an error message and prompts you again. If the system accepts a command which includes incorrect parameters or data, enter a new command to correct or counteract the original

2.2.2 Correcting Typographical Errors

Typographical errors can be corrected BEFORE you type a carriage return. You can delete the entire line or you can correct individual characters within the line.

To delete a line, type an "at" sign (@) or control-N. The line is deleted and a new prompt appears.

To delete one character from the screen, use the backspace or control-H to move the cursor back to the error. As the cursor moves, each character that it encounters is erased. When you reach the character in error, re-enter that character and all the characters that follow.

To delete a character from a hard copy terminal, use the DELETE, RUBOUT, underscore, (depending on your terminal), or control-Q to produce an underscore character on the hard copy. Each underscore represents one deleted character, moving backward from the current position. When the underscores equal the character positions to be deleted, type the replacement characters.

Example:

```
SL PROGTSG___RAM 25_3
```

is read by the system as:

```
SL PROGRAM 23
```

2.2.3 Displaying the Previous Command

You can display your previous command under HDOS by typing a control-G. You can repeat the command by pressing RETURN or typing a control-J, or you can modify or delete the command using the line editor.

NOTE

If the first character in a command is a '=', the command is not placed in the 'last command' buffer. A control-G typed at this point displays the previous command, not the one just sent.

Example:

1. Type: AL 102 <CR>

   The system displays the accounts on Hard Disk 102.

2. Type: =LI 1 <CR>

   The system lists the files on floppy disk drive 1 but does not enter the command into the last command buffer.

3. Type: Control-G

4. The system displays:

   ```
   =AL 102
   ```

2.2.4 Multiple HDOS Commands

You can key multiple HDOS commands on one line if you separate them with backslashes. Since the backslash is a legal character in filenames and accountnames, precede it with a space to make the command unambiguous. The commands are executed in the order entered and can be displayed with control-G then modified as needed.

Example:

```
AL 101 \LI 1
```

prints all account names and ID numbers from Hard Disk Drive 101 to your screen, then prints a directory listing from floppy disk drive 1.

2.3 DISK AND FILE INFORMATION

2.3.1 Floppy and Hard Disk Organization

Each hard disk or floppy disk consists of concentric TRACKS. The outermost track is identified as TRACK 0. Each track is subdivided into SECTORS, and each double-density sector holds 512 bytes of data. Every sector is identified by a unique DISK ADDRESS. Each sector has an address of 10X+Y, where x is the track number and y is the sector number. For example, sector 3 of track 27 on a floppy disk has the disk address of 273.

You may access data on a hard disk or floppy disk by file name, or by relative position within a named file. On a floppy disk, you can also access data by giving a physical disk address, such as 273.

2.3.2 Files

A file is an integral number of logically sequential blocks of data on a floppy or hard disk. A FILE BLOCK is defined as a unit of information equal to 256 bytes; therefore, a sector can contain two file blocks of information (one block on single-density floppy disks). Files always begin on sector boundaries. For example, a particular diskette file might occupy disk address 17 through 95 (track 1, sector 7 through track 9, sector 5).

The first four sectors on each floppy disk contain directory information; these sectors, 0 through 3 must not be specified as file addresses.

2.3.3 File Types

Each file is identified by its file type. Eleven file types are currently defined. More may be assigned in later versions of HDOS.

Type 0 - Default type. New files created by HDOS are assigned this type until explicitly changed by the TY command.

Type 1 - A file containing a machine language program (object code) that can be executed directly from HDOS with the GO command.

Type 2 - HBASIC program that can be loaded or saved from HBASIC.

Type 3 - HBASIC data file.

Type 4 - Backup diskette index.

Type 5 - Hard disk backup data.

Type 6 - CP/M workfile

Type 7 - CP/M unit

Type 18- ASP Sequential access file

Type 19- ASP Random access file

Type 20- ASP Index file

Types 32-63 - Unassigned by North Star. May be defined by user.

2.3.4 HDOS Data Structures

The following figures illustrate HDOS Data Structures on the hard disk and should be used in conjunction with the Equates listing in Appendix E. The exploded view of the hard disk in figure 2-1 is meant to show logical relationships between the structures but is not a physical representation of actual locations on the disk.

HARD DISK UNIT STRUCTURE

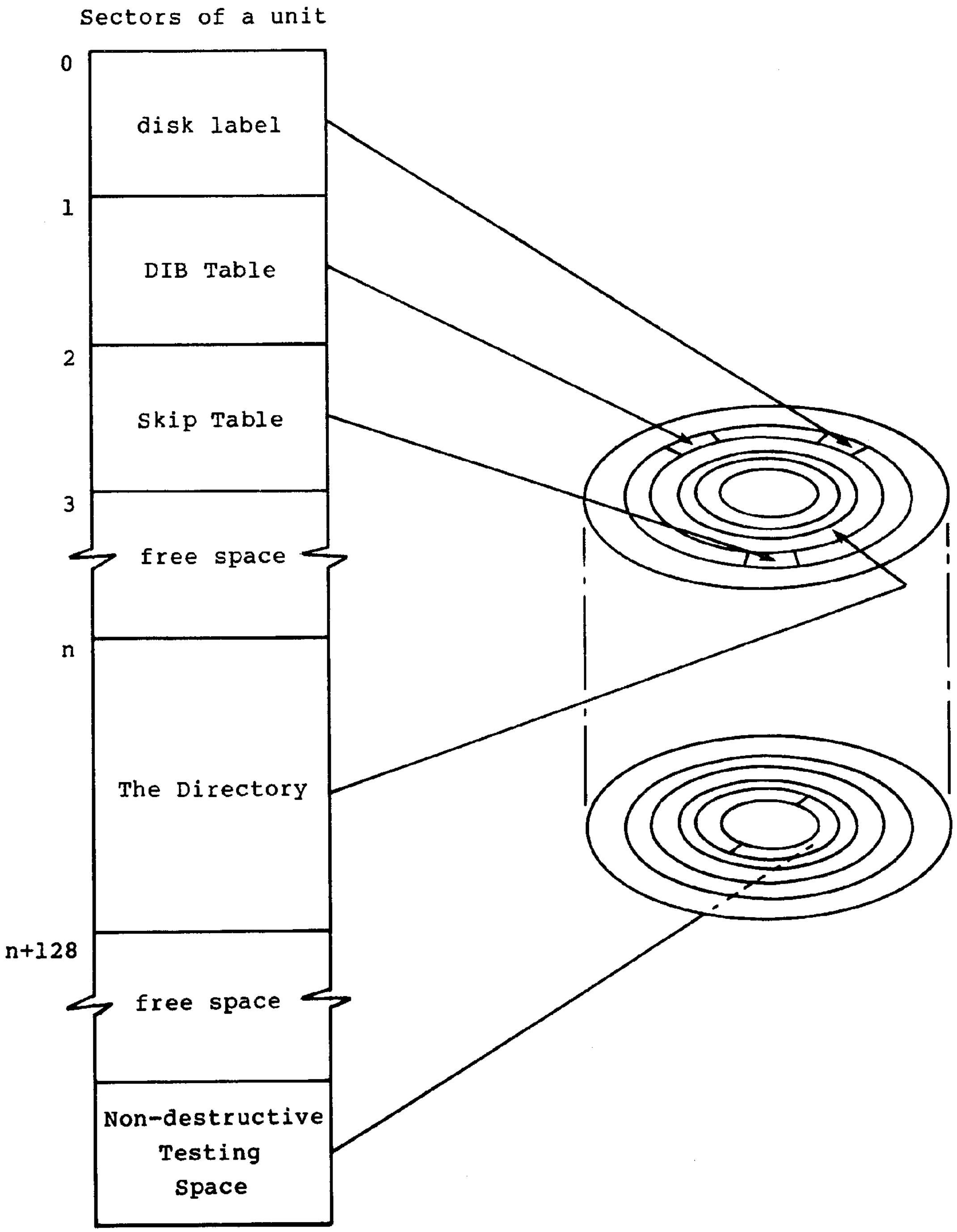
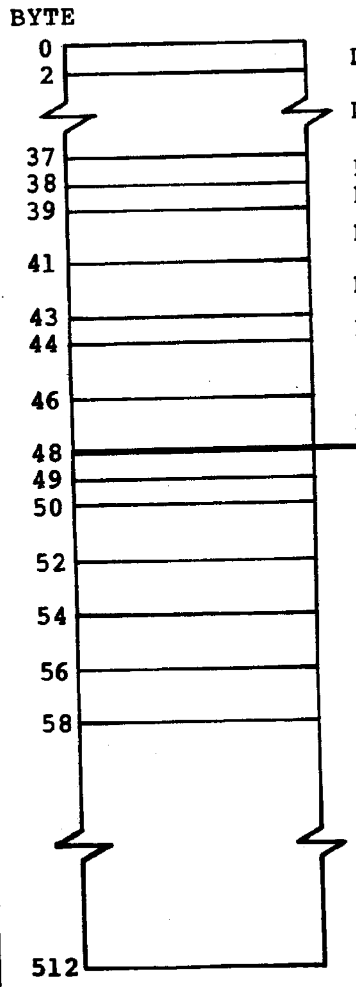


Figure 2-1

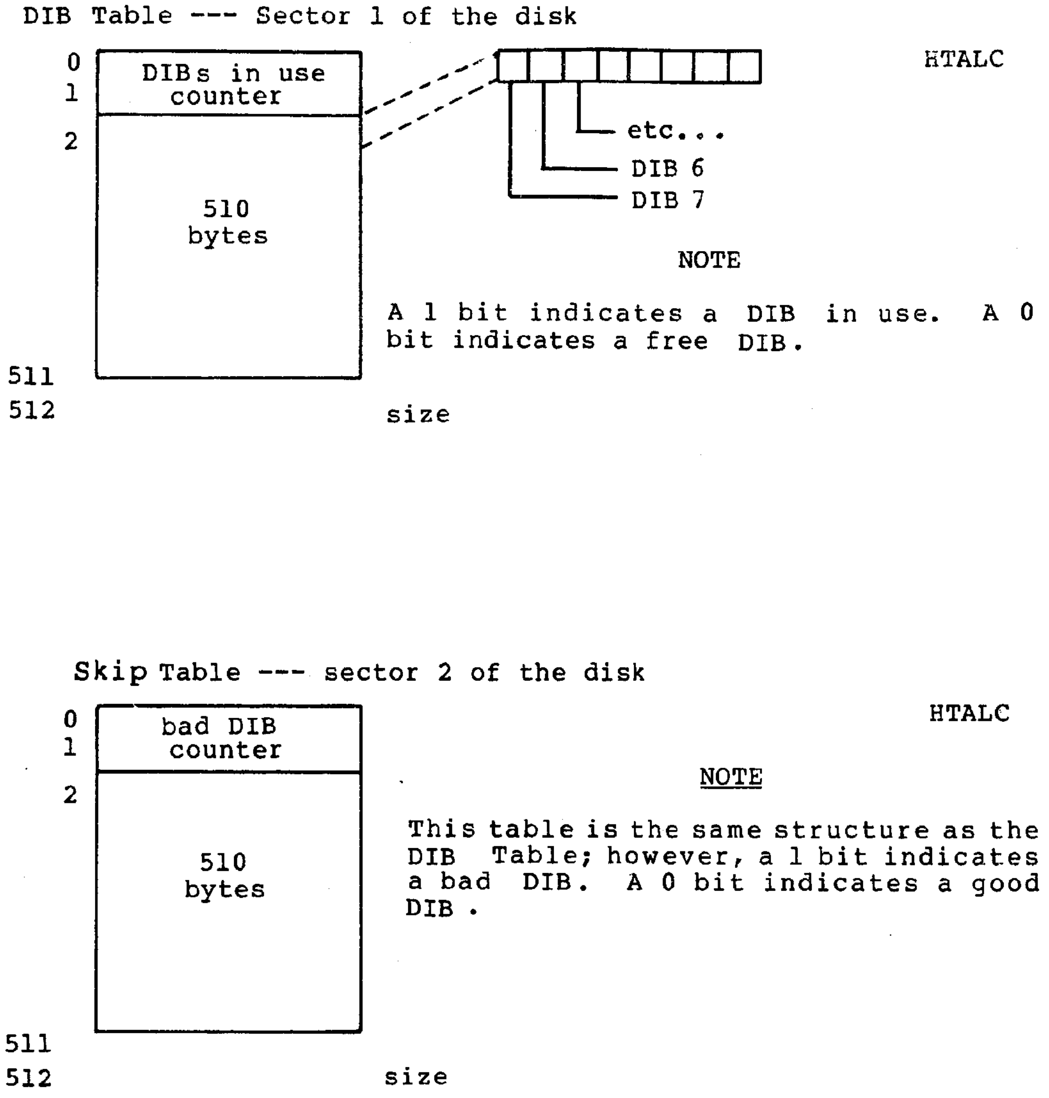
HARD DISK LABEL



BYTE			
0	DLILL	Illegal Directory Address	North Star HD5 and HD18 hard disks
2	DLAX	Auto-Load-and-Execute Pathname	
37	DLMAJOR	Major Disk Structure Revision Level	
38	DLMINOR	Minor Disk Structure Revision Level	
39	DLDSZE	Disk Size (Sectors per disk)	
41	DLNSRT	Number of Sectors reserved for TEST	
43	DLNHSZ	DIB Size as a power of 2	
44	DLDRSZ	Directory Size (Sectors)	
46	DLDIR	Directory Disk Address	
48	DLSST	Stepping Speed	Other North Star 5.25" hard disks
49	DLMXH	Head Number (Maximum)	
50	DLMXC	Cylinder Number (Maximum)	
52	DLPRC	Precompensation Cylinder Number (Lowest)	
54	DLLCC	Low Current Cylinder Number (Lowest)	
56	DLOFC	Safe Cylinder Number	
58			
512			

Figure 2-2

DIB TABLES



NOTE

A 1 bit indicates a DIB in use. A 0 bit indicates a free DIB.

NOTE

This table is the same structure as the DIB Table; however, a 1 bit indicates a bad DIB. A 0 bit indicates a good DIB .

Figure 2-3

FILE STRUCTURE

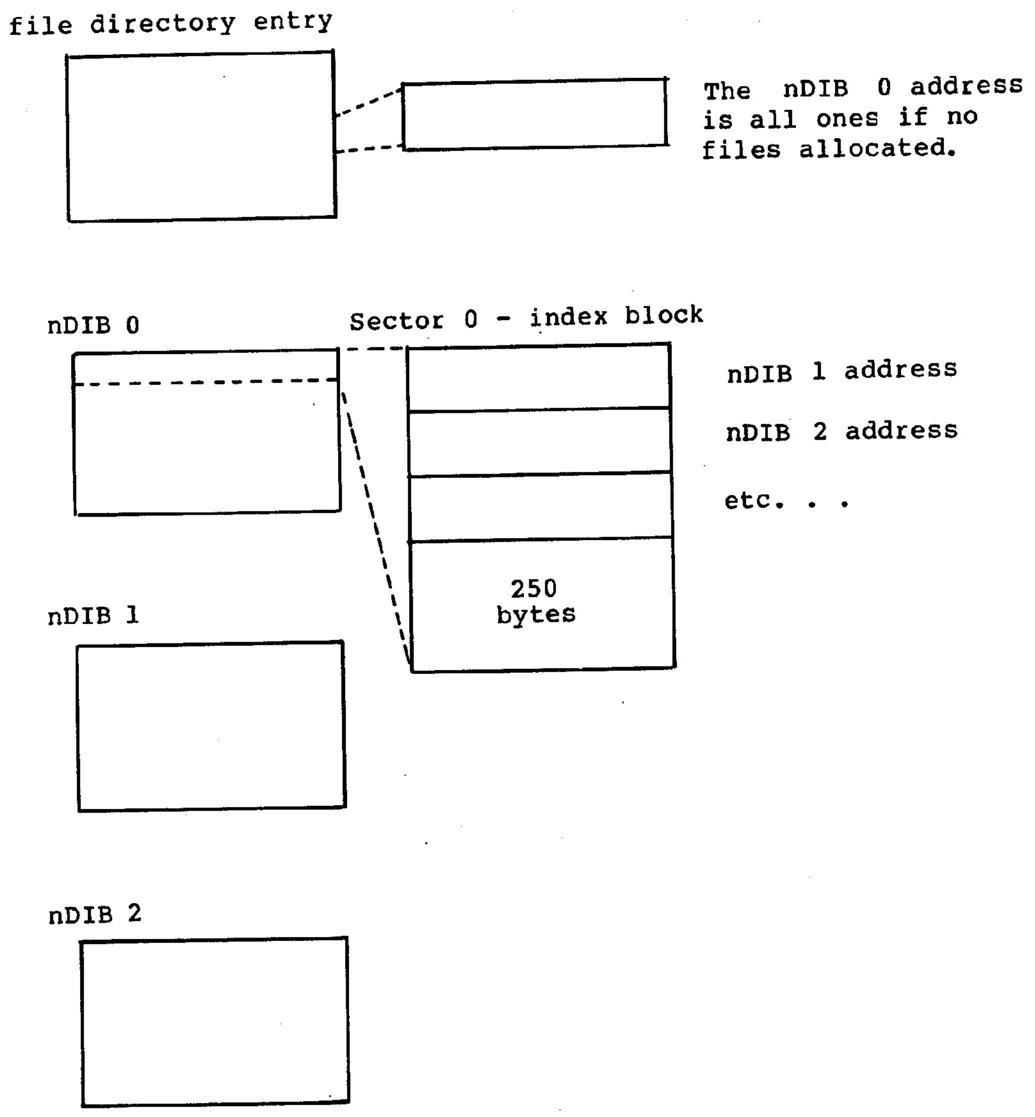


Figure 2-4

FILE DIRECTORY ENTRY

0		entry type	DETYP
1 2		account ID number	DEACN, DESYM
3 16	14 bytes	file name, nul file	DENAM
17		attributes	FDEAT
18 19		file size	FDEFZ
20 21		bytes in last sector	FDELB
22		nDIB size	FDEHS
23		reserved space	FDEX1
24 25		nDIB 0 address divided by SPH	FDEH0
26		file type	FDEFT
27 28 29		type dependant data	FDETD
31	2 bytes	reserved	FDEXX
32		size	DESZ

This is the structure description. It is copied to the OFB when the file is opened.

NOTE
The nDIB 0 address (FDEH0) will be FFFFH,if there is no disk space allotted to the file

Figure 2-5

ACCOUNT DIRECTORY ENTRY

0		entry type	DETYP
1 2		two bytes that must be zero	DEACN, DESYM
3 16	14 bytes	account name	DENAM
17 18		account ID	ADEAN
19 31	13 bytes	reserved	ADEXX
32		size	DESZ

Figure 2-6

FILE INDEX ENTRY

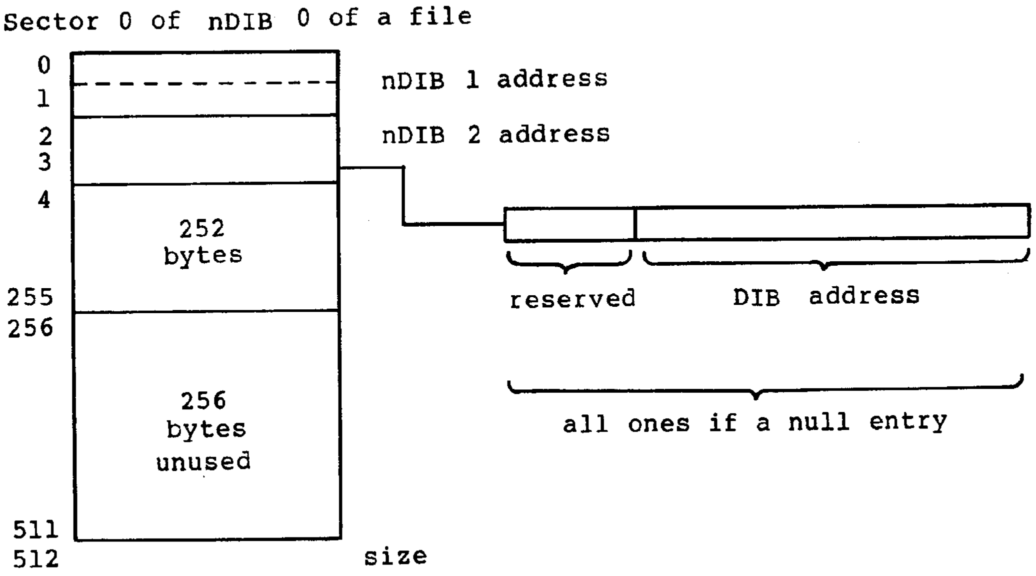


Figure 2-7

OPEN FILE BLOCK

0		directory entry number	OFBDE
1 2 3		disk address of directory	OFBDB
4		index length	OFBIL
5		attributes	OFBAT, OFBSD
6 7		file size in 512 byte blocks	OFBFZ
8 9		bytes in last block	OFBLB
10		nDIB size	OFBHS
11		unit number	OFBMU
12 13		nDIB 0 address	OFBH0
14	256 bytes	index	OFBI
269			
270		check byte	OFBCB
271		size	OFBSZ

The file structure description copied as a contiguous set of bytes from the directory.

Figure 2-8

CREATE INFORMATION BLOCK

0	type	CBTYP
1	attributes	CBATR
2	nDIB size	CBNHZ
3 4	pathname address	CBPNA
5 6 7	type dependent data	CBTDD
8	size	CBSIZ

Figure 2-9

TRANSFER COMMAND BLOCK

0	operation code	TCOP
1 2	beginning memory address	TCMEM
3 4	beginning sector number	TCSEC
5	transfer length in sectors	TCLEN
6 7	ending memory address	TCEMA
8 9	ending sector number	TCESN
10	sectors not transfered	TCSNT
11 12	bytes available in last sector	TCBLB
13	"or" of all dirty bits	TCDRT
14	size	TCSIZ

bytes 0 - 5 associated with request

bytes 6 - 13 associated with result

Figure 2-10

3.1 OVERVIEW

The HDOS command processor allows you to work with files on diskette or hard disk(s) by typing commands on a console terminal keyboard. When you press the RETURN key, the command and its arguments are processed and, if valid, executed. If the command is invalid, the system returns an error message.

With the HDOS commands, you can:

* Initialize a diskette.
* Create, work with, and delete files and accounts.
* List file directories and accounts.
* Load and execute files.
* Access RAM addresses and I/O ports.
* Control output devices.
* Perform maintenance and debugging functions.
* Rename Diskette files.
* Copy files from diskette to diskette, hard disk to diskette, and diskette to hard disk.

3.2 SYMBOL CONVENTIONS

When the syntax of a command is given in this section, the commands and words indicated with uppercase letters are to be typed exactly as shown. Words or arguments for which you must supply a value are indicated in lowercase. For example:

```
AC account
```

means that you must type the letters AC followed by a valid account name. For example, in the command AC TEST6, the name TEST6 replaces account.

Certain optional arguments only refer to a hard disk and are not used to refer to a diskette. These arguments are always surrounded by square brackets ([ ]). For example, in the command syntax:

```
AL [#n]  [d]
```

enter the command as shown and one, both or neither of the arguments. If you do include both arguments, enter them in this order.

Other optional arguments only refer to a diskette and never refer to a hard disk. These arguments are always surrounded by braces ({}). In the command syntax:

```
IN d  {dens}
```

enter the command, a value for d, then choose whether or not to enter a value for dens.

Occasionally an argument has both braces and square brackets. In the command syntax:

```
LI  {[#n]} [account] {d}
```

n is optional for either diskette or hard disk; account is optional if you are referring to a hard disk and is not used otherwise; d is optional if you are referring to a diskette and is not used otherwise.

3.3 ARGUMENTS

Most HDOS command arguments and the rules that govern them are described below. Arguments that apply to only one command are described with that command.

In general, numeric values that refer to disk addresss are expressed in decimal notation. Numeric values that refer to RAM (including GO addresses) or port addresses are expressed in hexadecimal notation. Any numeric argument (except a drive number appended to a file name or an account name) can be entered in hexadeciaml notation if immediately followed by an H, or decimal notation if followed by a T.

Example:

```
CR JONES 20H
```

This command creates a 32-block (20 Hex) file named Jones. Ordinarily the length of the file is specified in decimal notation. If you specify hex notation, the system makes the conversion.

<u>Argument</u> <u>Description</u>

#n Refers to an I/O device number from 0 to 7 inclusive. Generally, n is optional and defaults to 0. The Hard Disk Operating System assigns specific device numbers to specific peripheral devices. If your system has been customized, your device numbers may be different. The assigned device numbers are:

0 = Console terminal, left serial port.

1 = Printer, right serial port.

2 = Another device, parallel port.

3 - 7 (not implemented)

Argument Description

d

Diskette or hard disk drive number. Diskette drive numbers range from 1 to 4. Hard disk drive numbers range from 101 to 104.

NOTE

The default if no drive number is specified is 101, the first hard disk drive.

accountname

Name of an account on a hard disk drive. Account names consist of 1 to 14 printing characters, and can not begin with a number.

account

Used to organize files on the hard disk into groups. Accounts are specified by an accountname optionally followed by a comma and a hard disk drive number. If the drive number is omitted, drive 101 is assumed.

filename

Name of a file on a diskette or hard disk. On a diskette, a filename may contain from 1 to 8 printing characters; on a hard disk, a filename may contain from 1 to 14 printing characters.

pathname

Uniquely identifies a file on a particular drive and account. On a diskette, a pathname consists of a filename followed by a comma and a drive number. A hard disk pathname consists of a filename optionally followed by a comma and an account then another comma and a drive number. If an account is not included, the current default account will be used. This is initially account SYSTEM on hard disk drive 101, but may be changed by the user.

Argument **Description**

pathname (continued)

For example,

```
TESTFILE6,ACCT5,102
XINPUT,2
MARCHDATA,GLACCT27
99INFO,102
PAYROLLPROGRAM
JONES,ACCI1
jones,acct2
```

are all valid pathnames.

Upper and lowercase letters are different in a pathname. If you create a file named JONES, you cannot later refer to it as jones.

len

Length of a file or part of a file in blocks of 256 bytes. len is expressed in decimal notation.

dens

Density specification, used only when referring to diskettes. Density may be either S or s for single density, or D or d for double density. The default is double density.

alloc

is the size, in DIBs (Data Incremental Blocks, previously called "hunk") of the areas on a hard disk allocated to a file. A DIB is a group of sixteen contiguous sectors. The allocation factor is the number of DIBs grouped into a contiguous area on the disk, an area called a "segment".

Valid allocation factors are 1 (default), 2, 4, 8, and 16. An allocation factor of 8 will put segments on hard disk consisting of eight DIBs times sixteen sectors, for a total of 128 sectors each.

If the file uses more than one megabyte of disk space, specify an allocation factor greater than 1.

Argument Description

daddr Disk address. The disk address is usually expressed in decimal notation. The format for daddr is:

tracksector

For example, a diskette address of 357 means track 35, sector 7. This is sector 7 of the innermost track on side B of a double-sided diskette. On the hard disks, the address is a logical rather than a physical connection to track and sector.

paddr Port address. This address is expressed in hexadecimal notation (0-FFH).

raddr RAM address. The RAM address is usually expressed in hexadecimal notation (0-FFFF).

region A contiguous block of random-access memory (RAM) specified in one of the following ways:

1. A single address to specify a one-byte block at the given memory address.
2. Two addresses separated by a hyphen to specify the first and last byte of the block.
3. An address and a number separated by a comma to specify the beginning address and the length of the block.

bval Byte value--the value that fills a single byte. The value may be specified as either:

1. A decimal number from 0 through 255 (followed by the letter T).
2. A hexadecimal number from 0 through FF.
3. A printing character or a space enclosed in single or double quotation marks. A printing character is any character entered without using the control key or other function keys such as carriage return, line feed, tab, etc.

3.4 HDOS COMMANDS

3.4.1 ACCOUNT COMMANDS

Accounts are used to group files on the hard disk. All files on the hard disk are assigned to an account when they are created, and the account name becomes part of the "pathname" used to identify that particular file. The system assigns each account an account ID number. This ID number is associated with the files in that account.

HDOS provides the following commands to create, change, and delete account names.

AC Account Create

This command creates a new account name and assigns it an account ID number. The syntax of the AC (Account Create) command is:

```
AC account
```

where: account identifies the account to be created.

Example 1:

```
AC PROGONLY
```

The new account named PROGONLY is created on hard disk drive 101 .
Example 2:

```
AC PROGTWO,102
```

A new account, PROGTWO, is created on hard disk drive 102.

AR Account Rename

This command allows you to change the name of an existing account. The syntax of the AR (Account Rename) command is:

```
AR account TO newaccount
```

where: account identifies the current account.

newaccount will be the new name of the account.

The new account name can not already exist on the same drive.

Example:

```
AR SOFT1 TO SOFT1A
```

The account name SOFT1 is changed to SOFT1A. The account ID number associated with the account is not changed.

AS. Account Set

The HDOS assigns account SYSTEM on hard disk drive 101 as the default account. The AS command changes the default account to any other existing account name on any hard disk drive. This change remains in effect until the system is rebooted. The syntax of the AS (Account Set) command is:

AS account

where: account identifies the existing account to substitute for the current system default account.

Example:

AS TEST1CL

AD Account Delete

This command deletes an account name from a hard disk's account directory. Delete all files from the account before you delete the account (See the DE and MD commands). The syntax of the AD (Account Delete) command is:

```
AD account
```

where: account identifies the account to be deleted.

Example:

```
AD JONES
```

The account JONES is removed from the hard disk's directory.

3.4.2 FILE COMMANDS

These command allow you to create new files, change the size and attributes of existing files, copy files, and delete files.

CR Create File

This command creates a new file on either a diskette or hard disk. On a diskette, CR creates a file directory entry only; no accessing of the file occurs. On a hard disk, the allocated file space is completely initialized to zeros. The syntax of the CR (Create) command is:

```
CR pathname len {daddr} {dens} [alloc]
```

where: pathname identifies the file to be created.

len is the length of the new file in file blocks of 256 bytes.

daddr is the disk address at which the file is to begin. On a diskette, the default is the address immediately after the last file.

dens is the density of the file to be created.

alloc is the allocation factor. See the section on command arguments in this chapter.

Example 1:

```
CR JOBDATA,4 8 56 D
```

A new file named JOBDATA is created on a double-density diskette in Drive 4. The file is eight blocks long, and begins at disk address 56.

Example 2:

```
CR BASICII,2 10
```

A new file named BASICII is created on the diskette in Drive 2. The file is given a length of ten blocks. Because no disk address is specified, the new file starts immediately after the end of the last file on the diskette. The file defaults to double density.

Example 3:

```
CR HARD.DISK.FILE,JONES,102 1200 8
```

A new file named HARD.DISK.FILE is created on Drive 102 and associated with account JONES. The length of the file is 1200 blocks and space is allocated in segments of eight DIBs (128 sectors) each.

SL Set Length

This command changes the length of a file to the specified length. The syntax of the SL (Set Length) command is:

```
SL pathname len
```

where: pathname identifies the file whose length is changed.

len is the new length of the file, specified in blocks.

If the file is on a hard disk, this command will succeed. If the file is on a diskette, however, this command succeeds only if the new file length is shorter than the original file length, or if all the diskette space after the specified file is unassigned.

Example:

```
SL JONES2 16
```

A file named JONES2 in the default account has its file length changed to 16 blocks of 256 bytes each.

TY Type Files

All files created by HDOS are given a file type of 0. The TY command changes the current file type to the file type specified. It also assigns attributes to hard disk files. The syntax of the TY (TYPE) command is:

```
TY pathname [filetype] {[raddr]} [attr....]
```

where: pathname identifies the file whose type and/or attributes are to be changed.

filetype is a number from 0 to 63 that identifies the contents and use of the file.

raddr is the GO address of the file in RAM. It is only specified when a file type of 1 is declared.

attr is one or more attributes assigned to a hard disk file. These attributes can be:

SC = scratch file, not to be backed up.

BU = to be backed up, not a scratch file.

RW = read/write file, not write protected.

RO = read only file, write protected.

DP = delete protected, cannot be destroyed.

DE = delete enabled, can be destroyed.

The default file attributes of a newly created file are BU, DE and RW. Unspecified attributes are not changed.

If an error occurs during execution of this command no attributes, with the possible exception of file type, are changed.

Example 1:

```
TY NEWFILE  1 6666 SC
```

File NEWFILE on Drive 101 is given a file type of 1. Because the new file type is 1, it is a GO file and receives a RAM address of 6666. SC identifies this file as a scratch file. The other attributes of the file are not changed.

Example 2:

```
TY BASPROG,1 2
```

BASPROG on Drive 1 contains a BASIC program. No RAM address is permitted, since this is not a GO file. Attribute specification does not apply to diskette files.

CF Copy File

The command can:

1. Copy from one file into another.
2. Copy a diskette to an image file on hard disk.
3. Copy a hard disk image file to diskette.

1. Copy one file to another.

There are three variations for copying the contents of one file into another. The first copies into an existing file. The command syntax is:

```
CF pathnamel TO pathname2 {dens}
```

The second variation copies into a new file. Its syntax is:

```
CF pathnamel CR pathname2 {len} {dens} [alloc]
```

The third variation assumes that a destination file exists on a hard disk and sets its length. If the length is not specified, the destination file is set to the same length as the source file. Its syntax is:

```
CF pathname1 SL pathname2 [len]
```

where: pathname1 is the name of the file to be copied.

pathname2 is the name of the file into which the first file is copied. In CF-TO and CF-SL, pathname2 must name an existing file. In CF-CR, pathname 2 must not name an existing file.

d is the diskette drive number.

len is the length of the new file. If not specified, the new file will be the same length as the old file.

dens is applicable only when writing to a diskette. The default value is double density. If the density is changed, the directory is updated to reflect the change.

alloc is the allocation factor. See the section on arguments in this chapter.

The CF-TO command does not change the length of the destination file. If the destination file is shorter than the source file, the error message:

```
-----------------------------
WARNING: Making Partial Copy
-----------------------------
```

is displayed at the console.

Example 1:

```
CF XDATA,MYACCT TO XSAVE,YOURACCT
```

The file named XDATA in account MYACCT is copied to the file name XSAVE in YOURACCT.

The CF-CR command creates the specified destination file only if the destination file name does not already exist. If no length is given, the new file is set to the length of the source file.

Example 2:

```
CF BIG,1 CR BIGGER,2 100
```

Create a file named BIGGER, 100 blocks long, on the diskette in drive 2, then copy the file BIG on diskette drive 1.

The CF-SL command requires that the destination file exist on the hard disk. The length of the destination file is set to the specified length; if no length is typed, it is set to the length of the source file.

Example 3:

```
CF SMITH,1 SL SMITH
```

Note that a copy from a diskette file many find some source sectors with incorrect density. These sectors are not copied as is; instead, they are initialized to ASCII blanks in the correct density to preserve relative addressing within the file.

2. Copy a Diskette to Hard Disk

As in the file to file copy, there are three variations for creating a complete diskette copy into a hard disk file. The first copies the contents of the diskette into an existing file. The command syntax is:

```
CF ,d {dens} TO pathname
```

The second variation copies into a new file.
Its syntax is:

```
CF ,d {dens} CR pathname
```

The third variation assumes that a destination file exists on a hard disk and sets it to the same length as the source file. The syntax of the command is:

```
CF ,d {dens} SL pathname
```

3. Copy Hard Disk Image to a Diskette

This command will copy a diskette image file from hard disk back onto a diskette. Essentially, you are recreating a diskette, complete with diskette directory. The syntax for the command is:

```
CF pathname TO ,d {dens}
```

Example:

```
CF TEST,JEAN2,102 TO ,2
```

Copies the diskette image file TEST from account JEAN2 on hard disk 102 to the diskette in drive 2.

MC Multiple Copy

This command copies all or selected files from a diskette or account to another diskette or account. Any files already on the destination diskette or account are not disturbed. The syntax of the first variation of the MC (Multiple Copy) command is:

```
MC {d1} [account1] TO {d2} [account2] {len} {dens} [alloc]
```

The second variation of the MC command requires confirmation before copying each file:

```
MC {d1} [account1] YN {d2} [account2] {len} {dens} [alloc]
```

A third variation creates the destination account, then copies all files from the source diskette or account to the destination account. The syntax of the command is:

```
MC {d1} [account1] CR account2 [alloc]
```

where: d1, account1 is the diskette or account containing the files to be copied.

d2, account2 is the diskette or account receiving the new files.

len is the length of the new files. If not specified, each new file will be the same length as the old file.

dens is the density of the destination diskette

alloc is the allocation factor for hard disk files.

The MC - TO and the MC - CR commands display the name of each file before copying and "Copy Completed" when done. The MC - YN command displays the name of each file followed by a question mark; enter a "Y" to copy the file, or "N" to skip it.

The MC command executes the CF - CR command for each file copied. Note that the optional parameters are typically not used with the MC command.

The MC command can compress the contents of a diskette by copying all files to a freshly initialized diskette.

Any files copied from hard disk to diskette with file names from 9 to 14 characters in length will have the name truncated to the first eight characters.

If this command finds a file with the same filename, the message "Name already in use" will appear and the command will fail from that point.

DE Delete File

This command deletes any file that has the attributes, Read/Write and Deletable. The syntax of the DE (Delete) command is:

```
DE pathname
```

where: pathname identifies the file to be deleted.

If no drive is specified, the system looks for the file in the default account.

MD Multiple Delete

This command deletes all or selected files on a diskette or hard disk account. The syntax for MD (Multiple Delete) is:

```
MD {d} [account]
```

where: d is a diskette drive number.

account identifies an account on the hard disk.

The command displays the name of each file on the console terminal followed by a question mark. If a "Y" is entered, the file is deleted; if an "N" is entered, the file is left unchanged.

RN Rename Diskette File

This command renames a diskette file. The syntax of the RN (Rename) command is:

```
RN  filename1,d TO filename2
```

where: filename1 is the original diskette filename.

d is a diskette drive number.

filename2 is the new diskette filename.

Example:

```
RN TESTER,1 TO TEST
```

This sequence renames filename TESTER on diskette drive 1 to filename TEST.

3.4.3 DATA TRANSFER COMMANDS

These commands allow you to read files or parts of files from disk into RAM, and to write disk files or parts of files from RAM.

LF Load File
SF Save File

These commands transfer files directly between a specifed area in RAM and a diskette or hard disk. The syntax of the LF (Load File) command is:

```
LF pathname raddr
```

The syntax of the SF (Save File) command is:

```
SF pathname raddr
```

where: pathname is the name of the file to be transferred.

raddr is the file's address in RAM

RD Read Disk to RAM
WR Write RAM to Disk

These commands directly transfer blocks of data between a specified area in RAM and a specified portion of a diskette, or a file on either diskette or hard disk.

The syntax of the RD (Read) command is:

```
RD  len  {dens} FROM daddr{,d} {[OF Pathname ]} TO raddr
```

The syntax of the WR (Write) command is:

```
WR len {dens} FROM raddr TO daddr{,d} {[OF pathname]}
```

where: len is the length of the data to be transferred (in blocks).

dens is the data's density specification.

daddr is the disk address.

raddr is the RAM address.

d is the drive number.

pathname is the name of the file.

These commands do not support absolute addressing on a hard disk. If a pathname is specified, the disk address is used as a relative address within that file (expressed in blocks), and must fall on a sector boundary. If the pathname is omitted, then the disk is interpreted as an absolute address on a diskette and must be followed by a comma and a drive number.

Example 1:

```
RD 4 FROM 0,3 TO 5000
```

This command reads the first four blocks (the file directory) from the diskette in Drive 3 to RAM.

Example 2:

```
RD 2 FROM 0 OF HBASIC TO 5000
   .
   .
   .
WR 2 FROM 5000 TO 0 OF HBASIC
```

This sequence of commands could be used to personalize your copy of HBASIC. RD reads the two blocks from sector 0 of HBASIC to RAM address 5000H. After the change, WR sends the data back to its original location.

Example 3:

```
RD 6 FROM 23768 OF LOTS.OF.DATA TO 5000
```

The RD command is also good for moving a part of a very large file into RAM for changes or reading.

3.4.4 PROGRAM EXECUTION

These commands exit from HDOS and transfer control to a program that is present in RAM, or loaded from a disk file.

GO Load and Execute

This command loads a type 1 file into RAM from the indicated drive and begins execution of that file. This command reads the entire file into RAM beginning at the GO address, then jumps to the GO address. Therefore, the first byte of the file must be the entry point of the program. The syntax of the GO command is:

```
GO pathname {[args]}
```

where:	pathname	is the name of a type 1 file to load and execute.
	args	are the arguments sent to a program through the command string. Maximum length of the entire command line is 80 characters.

The GO command sets the HL register pair to a value that points to the remainder of the command line (any characters typed after the pathname).

Example:

```
GO HBASIC
```

The machine language program HBASIC is loaded into RAM and executed.

JP Jump to a RAM Address

The JP command executes programs currently in RAM by jumping to the specified RAM address. The syntax of the JP (Jump) command is:

```
JP raddr {[args]}
```

where: raddr is the RAM address.

args are the arguments to be sent to a program through the command string.

Like the GO command, you can send arguments to the program as part of the command line. JP sets the HL register pair to point to the remainder of the command string.

3.4.5 LISTINGS AND STATISTICS

These commands enable you to list account names and, fle directories, and determine the available work space in RAM.

AL Account List

This command produces a list of all accounts existing on a specified hard disk drive. The syntax of the AL (Account List) command is:

```
AL [#n]  [d]
```

where: n is the device number of the output device on which the list is to be printed or displayed.

d is the drive number.

Example 1:

```
AL #1 102
```

All account names and ID numbers from hard disk drive 102 are printed on the printer.

Example 2:

```
AL
```

All account names and account ID numbers from hard disk drive 101 are displayed at the console terminal.

LI List File Directory

This command produces directory listings from either a diskette or an account on hard disk. The syntax of the LI (List File Directory) command is:

```
LI  {[#n]} [account] {d}
```

where: n is the output device number on which the directory is displayed or printed.

account identifies the account whose directory is displayed. If no account is specified, the current default account is assumed.

d is the diskette drive number.

For each file in a diskette directory the LI command causes the output drive to display:

- Starting disk address
- Length
- Density
- Type

For each file in a hard disk directory the LI command displays

- Filename
- Length in blocks
- Allocation factor in DIBs
- Attributes
  S = Scratch
  W = Read/Write
  U = Updated but not backed up
  D = Deletable
- File type

Certain file type dependent information is displayed, such as GO addresses for type 1 files and account ID numbers for type 5 files.

To stop listing, type a control-C.

LI List File Directory (continued)

Example 1:

```
LI #1 JONES
```

The directory for account name JONES is printed on the printer.

Example 2:

```
LI
```

All the files in the current default account are listed on the console.

ML Multiple List

This command produces a directory listing of all accounts on all hard disk drives. The syntax of the ML (Multiple List) command is:

```
ML [#n]
```

where: n is the output device number.

WS Work Space

The WS command displays the amount of work space available in memory. The syntax of the WS (Work Space) command is:

```
WS {[#n]}
```

where: n is the output device number.

The system lists starting and ending addresses for available work space in hexadecimal and decimal notation.

ST Display Hard Disk Statistics

This command prints hard disk statistics on the number of bad spots, system overhead on the disk, and used and unused disk space. The syntax of the ST (Statistics) command is:

```
ST [d]
```

where: d is the hard disk drive number.

The ST command displays all hard disk statistics in DIBs.

3.4.6 MEMORY COMMANDS

These commands allow you to display, search, change and move the contents of memory.

EM Examine Memory

This command examines the contents of a specific memory address. The syntax of the EM (Examine Memory) command is:

```
EM {[#n]} raddr
```

where: n is the output device number.

raddr is the address of the memory location to be examined.

The output from the EM command consists of the RAM address (in hexadecimal) and the contents of that address, expressed in binary, decimal, hexadecimal, and ASCII notation.

Example:

```
EM 41FE
```

The system returns the following information:

```
41FE   1100  0111   199T   C7H   -G
 ^         ^          ^     ^     ^
 |         |          |     |     |
address    |       decimal  |   valid ASCII char.
         binary         hexadecimal
```

DH Display in Hexadecimal

This command displays a regions contents in a format of two hexadecimal digits per byte, with sixteen bytes on each line. The syntax of the DH (Display In Hex) command is:

```
DH {[#n]} region
```

where:		
where:	n	is the output device number on which the addresses are displayed.
	region	is the area from which the display is taken.

Example:

```
DH 1200T,100T
```

The contents of memory from 1200 to 1299 (decimal) are printed at your console in hexadecimal notation.

DA Display in ASCII

This command displays the contents of a region in the same format as DH, with additional lines showing the ASCII character represented by the low order seven bits of each byte. A control code is printed as a blank and each character is preceded by a minus sign if the high order bit of the byte is a one. The syntax of the DA (Display in ASCII) command is:

```
DA {[#n]} region
```

where: n is the output device.

region is the area of memory whose contents are displayed.

Example:

```
DA 1200T-1299T
```

The contents of memory from 1200 to 1299 (decimal) are to be printed both in hexadecimal and as ASCII characters.

DS Display and Substitute Memory Values

This command displays the contents of a specified memory area one byte at a time, and allows you to substitute a new value for each byte displayed. The syntax of the DS (Display And Substitute) command is:

DS raddr

where: raddr is the starting address of the memory area from which byte values are to be displayed.

After each byte is displayed, a new hexadecimal value from 0 through FF may be entered, followed by a terminator. If you do not wish to substitute a new value, simply enter a blank, comma, or carriage return.

1. A blank displays the next byte for replacement.

2. A comma skips the next byte and goes directly to the following byte.

3. A carriage return ends the command and returns you to command level.

NOTE
Since replacement takes place immediately, a typing error must be corrected with backspace commands before the terminator is entered.

Example:

Type:

DS 3233 <CR>

The system prompts:

```
3233 64=
```

Type: 0 <CR>

The user displays the contents of address 3233 for possible replacement. The system returns 3233 64=. The user types 0 followed by RETURN to replace 64 Hex with a zero.

SM Search Memory

The SM command searches a specified area of memory to find and print each address of a specific byte value. The syntax of the SM (Search Memory) command is:

```
SM {[#n]} region bval{[,bval,........,bval]}
```

where: n — is the output device number.

region — is the area of memory searched.

bval — is the byte value searched for. If a sequence of byte values is used as the search pattern, separate the values with commas.

Example 1:

```
SM 2000-29FF 1
```

Byte values of 1 are searched for in the region from 2000 through 29FF (Hex).

Example 2:

```
SM 4000-5000 "M","I","N","E"
```

This command lists starting addresses of each occurrence of MINE in the specified region.

FM Fill Memory

The FM command fills a specified area of memory with an arbitrary string of byte values. The syntax of the FM (Fill Memory) command is:

```
FM region bval{[,bval,........,bval]}
```

where: region — is the area of memory to be filled.

bval — is the byte value. If a string of byte values is used, separate the values with commas.

Example:

```
FM 4000,100 FF
```

The 256 bytes of memory starting at address 4000 (Hex) are filled with the byte value "FF".

MM Move Memory

The MM command moves the contents of an area of memory to another area of the same size. Overlapping areas of memory are allowed. The syntax of the MM (Move Memory) command is:

```
MM region raddr
```

where: region is the memory area containing data to be moved.

raddr is the memory area the data is moving into.

Example:

```
MM 9000T,100T F000
```

This moves the contents of the one hundred (decimal) byte region starting at address 9000 (decimal) to the area starting at F000H.

VM Verify Memory

This command compares the contents of two memory areas and prints the address and contents of all non-identical bytes. The syntax of the VM (Verify Memory) command is:

```
VM {[#n]} region raddr
```

where: n — is the output device number on which the addresses are displayed.

region — is the first area to be compared.

raddr — is the starting address of the second area.

Example:

```
VM 3400,7 E385
```

The contents of the seven bytes starting at address 3400 (Hex) are compared with the seven bytes starting at address E385.

3.4.7 I/O COMMANDS

These commands allow you to directly access I/O devices and ports.

DO Device Output

This command sends any sequence of print or control characters to an output device. The DO command uses the software driver for the specified device. The syntax of the DO (Device OUtput) command is:

```
DO  {[#n]} {[char]}
```

where: n — is an output device number. The default is 0.

char — is a single printing character that terminates execution of the command. If a character is not specified, a RETURN terminates the command.

After the RETURN key is pressed to execute the command and the carriage return and line feed are echoed at the console terminal, no prompt appears for the next command until the selected terminator is entered again. If the second argument is omitted, the next RETURN acts as the terminator. All characters entered before the terminator, including control characters that normally activate the line editor, are sent directly to the specified or default output device.

EP Examine Port

The EP command examines the value at the specified input port. The syntax of the EP (Examine Port) command is:

```
EP [{#n}] paddr
```

where: n is the output device number.

paddr is the address of the input port.

The output from this command is the same as for the EM (Examine Memory) command.

PO Send Value to an Output Port

This command sends a byte value to the specified output port. The syntax of the PO (Port Output) command is:

```
PO bval TO paddr
```

where: bval is the value sent to the port address.

paddr is the output port address.

3.4.8 DISKETTE COMMANDS

These commands initialize, copy, and test floppy diskettes.

IN Initialize a Diskette

Before you can use a new diskette you must initialize it. You can also initialize a used diskette. This process removes all data on a diskette, initializes a new directory, and guarantees that no read errors will result from access to an uninitialized file block. Needless to say, choose the proper diskette before issuing this command since all the previous data on the diskette will disappear forever. The syntax of the IN (Initialize) command is:

```
IN d {dens}
```

where:	d	is the drive number of the uninitialized diskette.
	dens	specifies whether the diskette is initialized to single or double density. The default is double-density.

The IN command writes each block on the specified diskette drive with ASCII blank characters. The system initializes both sides of a double sided diskette if the drive is double sided, but only Side A if a single sided drive is used. This command takes about 45 seconds to execute.

Example:

```
IN 2 D
```

The diskette in Drive 2 is initialized to double-density.

CD Copy a Diskette

This command copies one diskette to another. The syntax of the CD (Copy Diskette) command is:

```
CD d1 TO d2 {sides}
```

where: d1 is the drive containing the diskette to be copied.

d2 is the drive containing the diskette that receives the copy. Note that any previously existing data on this diskette is overwritten.

sides indicates which sides of the diskette are copied. Y or y indicates that both sides are copied. Enter N or n if either of the diskettes is single sided or if only side A of the source diskette contains significant data. The default is Y.

Example:

```
CD 1 TO 3
```

The diskette in drive 1 is completely copied to the diskette in drive 3.

Any effort to copy the second side of a double sided diskette to a singe sided diskette gives you an error message at sector 350. Also, any attempt to copy the phantom second side of a single-sided diskette results in the same message.

3.4.9 MISCELLANEOUS COMMANDS

These commands perform control and monitoring functions.

IL Initial Load

This command performs an initial load of any operating system diskette to RAM. The syntax of the IL (Initial Load) command is:

```
IL
```

The command jumps to the bootstrap loader in ROM. Use this command instead of pressing the reset switch on the back panel of the computer.

RS Reset the File Manager

The RS command resets the File Manager and closes any open files. It also restarts the hard disk drive motors if you have turned them off. The syntax of the RS (Reset) command is:

```
RS
```

OF Turn Off the Hard Disk Drive Motors

The OF command turns off one or all of the HD18 hard disk drive motors. Use the OF command before you power down the HD18 hard disk unit to maintain the integrity of your data. The syntax of the OF (Off) command is:

```
OF [d]
```

where: d is the hard disk drive to be turned off.

If no drive number is specified, all hard disk drives are turned off.

NOTE
Five inch hard disk units are powered down without using this command. However, it is good practice to use the OF command to move the disk heads to their landing zones whenever you move either the five or eighteen inch disks units. An HD-18 will survive an occasional power failure, but should not be routinely powered off without using the OF command.

EB Examine Byte

The EB command displays a single byte value in binary, hexadecimal, decimal, and ASCII. The syntax of the EB (Examine Byte) command is:

```
EB {[#n]} bval
```

```
where:  n            is an output device number.

        bval         is a byte value.
```

The format of the output from EB is the same as for the EM and EP commands.

OD Specify an Additional Output Device

The OD command causes all output directed to the console terminal (device #0) to go to the additional output device as well. The syntax of the OD (Output Device) command is:

```
OD {[#n]}
```

where: n is the additional output device.

This command stays in effect for any program using the jump table, until set back to device 0.

SX Set Auto-Execute File Name

The SX command tells the File Manager the name of the file containing the transient part of the Hard Disk Operating System (that part of the HDOS overwritten by HBASIC and reloaded each time control returns to the operating system). The syntax of the SX (Set Auto-Execute File Name) command is:

```
SX pathname
```

where: pathname — is the name of the file containing the transient part of the system.

Note that this command must not be used to Auto-Execute HBASIC. To re-load the normal transient portion of the HDOS Command Processor, enter:

```
SX TRANSIENT,SYSTEM,101
```

PA Set Listing Page Length

This command sets the listing page length. The syntax of the PA (Page) command is:

```
PA n
```

where:

n	is the listing page length, a number between 0 and 254.

HE Help

When this command is used without an argument it lists al HDOS commands at your console. If you use an HDOS command as an argument, the command syntax is listed at your terminal. The above syntax of the HE (Help) command is:

```
HE [command]
```

where: command — is the HDOS command for which you need help.

Example:

```
HE MC
```

4.1 OVERVIEW OF THE PROGRAMS

One of the most important tasks for you to do on a regular basis is to maintain backup copies of your hard disk files. You can lose data on the hard disk in several ways: you may have a hardware failure, enter an incorrect command, run a program which creates unforseen changes, encounter a power transient or failure which destroys data. As a consequence, it is extremely important for you to backup on some type of routine cyclical basis, all data that you want maintained over time.

The HDOS BACKUP and RECOVER utility programs allow you to transfer hard disk data to diskettes. Then, if one or all of your files on hard disk becomes unusable, you can easily restore the files from backup diskette to hard disk. BACKUP records on diskettes any changes in the contents of any portion of the hard disk to the state of the data at the time of the desired BACKUP.

Section 4.2 describes the BACKUP program and provides instructions on how to use it. Section 4.3 describes the RECOVER program and provides instructions on how to recover data from backup diskettes.

4.2 BACKUP

The basic purpose of BACKUP is to copy the contents of the hard disk onto diskettes. If the information stored on the hard disk is accidentally destroyed, those files can be recovered from the diskette to the hard disk. BACKUP also extends the storage capabilities of the hard disk system by storing data off line.

HDOS provides three types of backup: COMPLETE, INCREMENTAL, and SELECTIVE.

Each run of the BACKUP program is called a SESSION. Each session creates a backup MASTER diskette, containing the hard disk account and file directory, and a series of CONTINUATION diskettes, the number of diskettes depending on how much data you have to backup from hard disk.

Each session becomes part of a SERIES. A series is a collection of sessions, always beginning with a COMPLETE backup and followed by any number of INCREMENTAL backup sessions.

A COMPLETE backup copies all sectors of all files found in the hard disk directory except those specified as SCratch files. Once completed, the diskettes contain a complete image of the hard disk. If something happens to the hard disk, the files can be safely recovered from the backup diskettes. A COMPLETE backup should always be done before any diskettes in the previous series are erased or re-used.

An INCREMENTAL backup copies only those portions of the hard disk that have been changed since the last COMPLETE or INCREMENTAL backup. An INCREMENTAL backup is always part of a series, and the information from this backup procedure is added to the information stored in previous backup sessions. For example, one initial COMPLETE backup and two INCREMENTAL backups constitute a series of three backup sessions.

A SELECTIVE backup copies only files that you specify. It cannot be part of a series and can only be initiated outside such a series. This option is useful when there are only a few files on the hard disk that are worth saving and you do not want to spend the time or diskettes to backup the complete disk. There is, however, one disadvantage to this option. If the contents of the disk are completely destroyed, a total recovery can only be done with a series that began with a COMPLETE backup.

The SELECTIVE backup procedure can save the entire contents of individual HDOS files or CP/M units. Since each CP/M unit is associated with an HDOS file, a particular CP/M unit may be backed up using the selective backup procedure and specifying the connected HDOS file for backup. This option gives you the means for storing and transporting on diskette any file or unit which will not fit on one floppy disk.

The following figures show the order of Backup and Recovery for three series of three sessions each. North Star recommends keeping at least three complete series of Backup diskettes.

COMPLETE BACKUP CYCLE

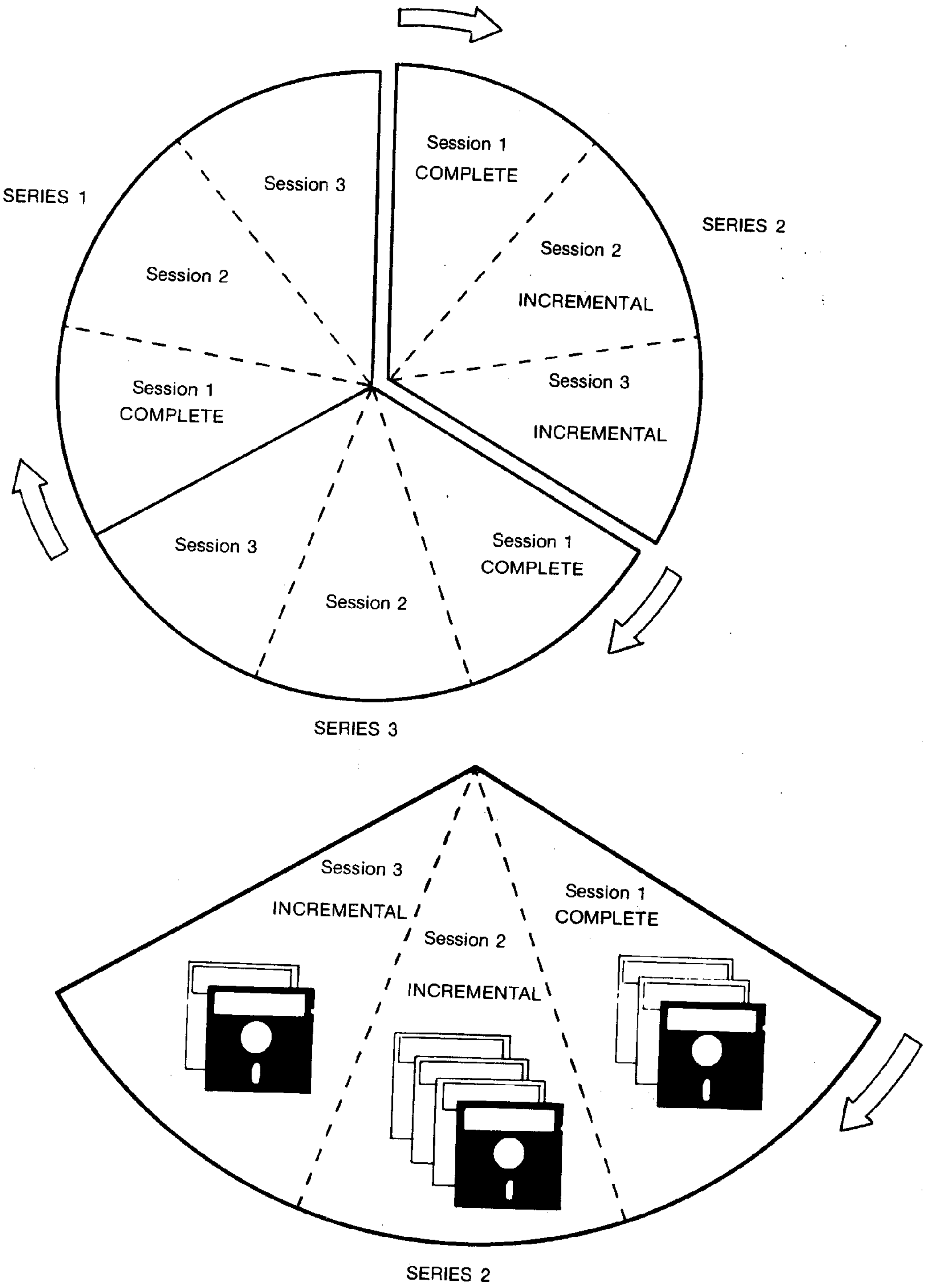


Figure 4-1

COMPLETE RECOVERY FROM SERIES 2

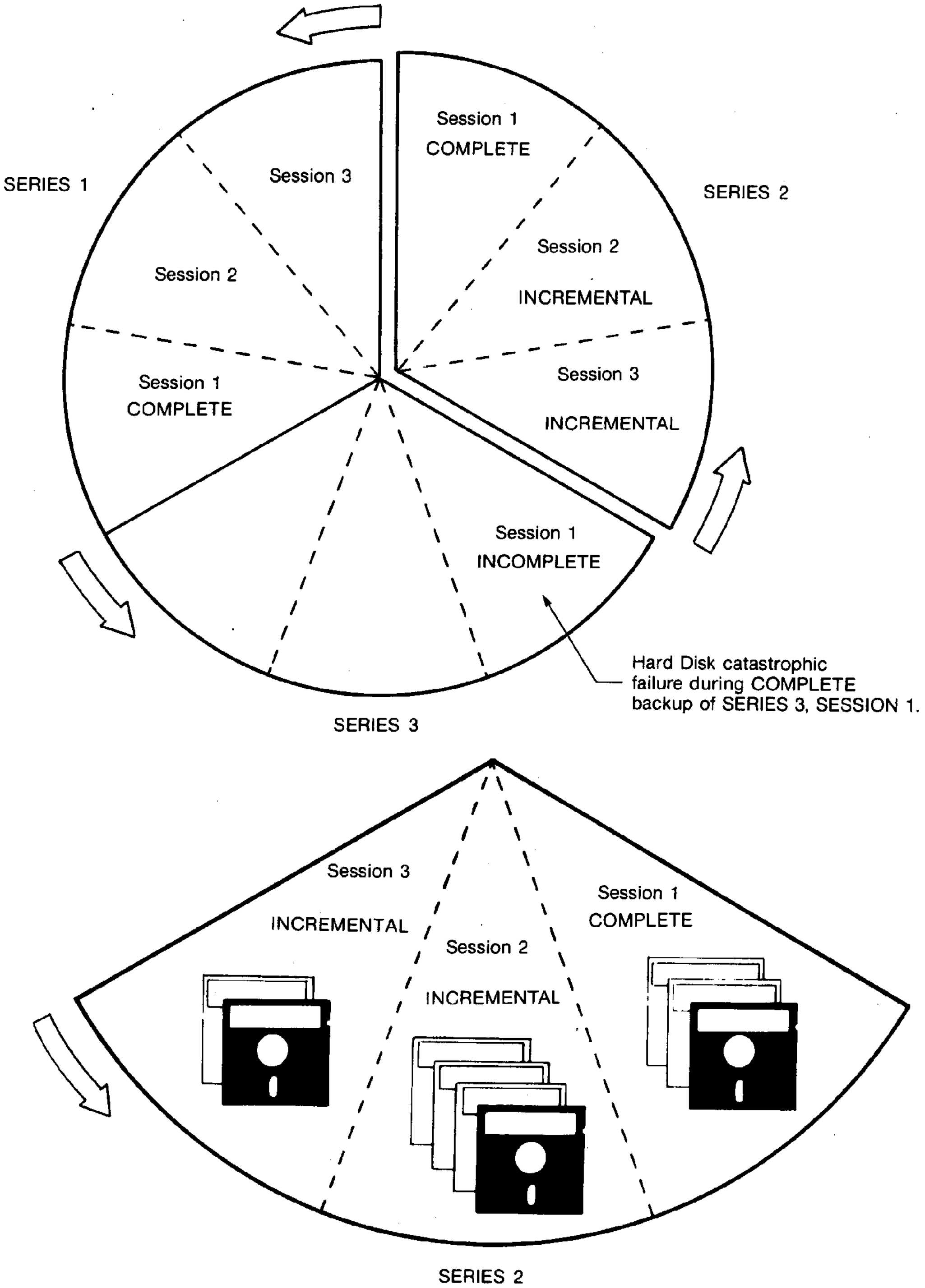


Figure 4-2

4.2.1 USING BACKUP

To initiate the BACKUP program,

Type: GO BACKUP <CR>

The program prompts:

```
HARD DISK BACKUP ON FLOPPY DISKETTES

        NORTH STAR COMPUTERS, INC.
         VERSION *.*.*

         1. Complete backup
         2. Incremental backup
         3. Selected files or accounts backup
         4. Explanation

Selection (or ESCAPE to exit to HDOS) :
```

NOTE

> A hard copy listing is recommended for all BACKUP procedures. It provides a ready reference for any future file recovery.

4.2.2 COMPLETE BACKUP

Execute the following procedure to run selection 1 of the BACKUP menu, COMPLETE BACKUP.

ENTER	ACTION OR PROMPT
1	Select COMPLETE Backup. The program prompts for today's date. (Do not use blanks.)
Today's date <CR>	The program prompts for listing device.
Listing device number	Selection '3' causes program to prompt for printer device number. Program prompts for hard disk drive number.
Hard disk drive # <CR>	Program prompts for diskette drive number.
Diskette drive #	The program indicates 'BACKUP STARTED' and the hard disk directory is compressed and written to the Master diskette.

For each file or account found in the directory, a dot appears on the screen. All sectors of all files found in the directory are copied onto the diskette except those with the attribute of SC for scratch file. The backup is done account by account; each account with its files is displayed on the screen after the copy.

After each file is copied, the filename is displayed along with the length of the file. The SIZE heading indicates the length of the file in blocks of 256 bytes. The heading BACKED UP shows how much of the entire file or account fit onto a single diskette. The following is a partial screen display from a COMPLETE backup:

```
ACCOUNT        NAME                  SIZE   BACKED UP

***
SYSTEM         BACKEXP                 6          6
*******************************
SYSTEM         FPHBASIC               58         58
*******************************
SYSTEM         RECOVERS               46         46
```

At the end of the COMPLETE backup, the program prompts:

```
Please remove BACKUP.x from drive 1 and label it

File data backup complete
File cleanup started
```

If the diskette is the first in the session, it contains the directory and is the MASTER diskette called BACKUP.M. Any other diskettes called for in the session are CONTINUATION diskettes and are named BACKUP.C.

The file cleanup procedure resets the dirty bits associated with each sector on the hard disk, and removes the U from the directory listing to indicate that all non-scratch files have been backed up. As soon as you alter one of these files, however, the U is restored to the directory listing for that file. This allows the INCREMENTAL backup procedure to identify files changed since the last COMPLETE backup, and to copy only those specific files in the next INCREMENTAL backup sesson. A dot is displayed on your screen as each file is cleaned.

The directory listing of an HDOS diskette includes a column of letters indicating the status of the files.

S = scratch file (not backed up)
W = write-enabled
U = used since last backup
D = delete-enabled

The absence of a letter indicates the opposite status. For example, if there is no S, the file is backed up. If there is no D, the file cannot be deleted.

4.2.3 INCREMENTAL BACKUP

An INCREMENTAL Backup never begins a series but is always a sesson within a series. The information from the INCREMENTAL backup follows information stored from previous backup sessions. Only sectors changed since the last COMPLETE or INCREMENTAL backup are copied.

ENTER	ACTION OR PROMPT
2	Select INCREMENTAL Backup from main BACKUP menu.
	Program prompts for today's date. (Do not use blanks.)
Today's date <CR>	Program prompts for listing device.
Listing device #	Selecting a '3' for "Other printer" brings prompt for printer device number.
	Program prompts for hard disk drive number.
Hard disk drive# <CR>	Program prompts for diskette drive number.
Diskette drive #	Listing device displays drive capacity (Quad, etc), the program compresses the hard disk directory, writes it to the Master diskette, then begins the INCREMENTAL backup.

The INCREMENTAL backup copies the sectors of files onto the diskette account by account. For each sector that is copied, a dot (.) is printed on the screen. After each file has been copied, the filename is displayed along with the length of the file. The number found under the heading 'SIZE' is in file blocks (256 bytes).

4.2.4 SELECTED FILES OR ACCOUNTS

If you type GO BACKUP and choose Selected Files or Accounts from the main menu, each individual file or account you specify is copied onto diskette.

ENTER	ACTION OR PROMPT
3	Choose Selected Files Or Accounts from the main menu.
	Program prompts for today's date. (Do not use blanks.)
Today's date <CR>	Program prompts for listing device number.
Listing device #	Selecting a '3' causes a prompt for printer device number.
Hard disk drive # <CR>	Program prompts for diskette drive number. (The first diskette is your Master for this session).
Diskette drive # <CR>	The hard disk directory is compressed, written to diskette, the screen indicates 'Backup started', and a dot is displayed for each file found.
	The program prompts for the Hard disk 'Account Name' you wish to backup.

ENTER	ACTION OR PROMPT
Account name <CR>	A menu appears with the following selections.
1. The entire account	Searches for all files of the given account name, list the files, then backs them up.
2. List the files	Lists all files in a specified account with file size, then prompts for the next name to list or backup.
3. List the name and ask for confirmation	Lists each file in the account followed by a question mark. Type 'N' to omit a file from the backup, 'Y' to copy the file to the backup diskette. Each file backed up lists: . Account Name . File Name . File Size . Number of blocks stored on diskette.
4. Name a specific file	Prompts for the name of file each file to copy. As each sector of a file is copied an asterisk (*) apears on the screen. When all the files you want to select are copied:
<CR>	The program prompts for another 'accountname' to continue the Backup process.

ENTER	ACTION OR PROMPT
New Accountname <CR>	If there are no more accounts to backup you should enter a RETURN.
<CR>	Program ends and returns you to HDOS command level.

4.3 RECOVERY

There are two ways to recover data from diskette to the hard disk: TOTREC and RECOVER.

The TOTREC utility program is designed for situations where the hard disk has crashed, been completely erased, or contains useless data. This program reinstates the disk directory from the master backup diskette and recovers the entire contents of the hard disk.

The RECOVER utility program is used when the hard disk as a whole remains good. This program allows you to recover a file or account after the material was accidentally deleted from the hard disk, or to reinstate an earlier version of data currently stored on the hard disk.

RECOVER retrieves data put on floppy diskette for long term storage. The program also provides portability for files too large to fit on a single floppy diskette.

A RECOVER can begin from any backup session in a series and include previous sessions of the series if these are needed. It can also recover data from a SELECTIVE backup that is not part of a series.

A RECOVER always begins with a session MASTER and is followed, in any order, by the CONTINUATION diskettes for that session. Previous sessions, if needed, always begin with the session Master for each particular session.

4.3.1 USING RECOVER

To initiate the RECOVER program, enter:

GO RECOVER <CR>

The program will prompt with the initial RECOVER menu.

```
----------------------------------------------
        RECOVER PROGRAM FOR FLOPPY DISKETTES

             NORTH STAR COMPUTERS, INC.

                VERSION * RELEASE *

1.  Accounts Listing

2.  Recover files or documents

3.  Explanation

Selection (or ESCAPE to exit to HDOS):

----------------------------------------------
```

4.3.2 ACCOUNTS LISTING

This selection searches the directory for a specified Backup session and prints the name of each account found.

ENTER	ACTION OR PROMPT
1	The program prompts for a listing device. If you select '3', the program prompts for a printer device number.
The listing device #	The program prompts for the diskette drive number for the Master backup diskette.

ENTER	ACTION OR PROMPT
The diskette drive #	The program displays the Master diskette label then prints the names of every account found in the directory.
<CR>	Return to the main RECOVER menu.

4.3.3 RECOVER FILES OR ACCOUNTS

This option recovers all or selected files from a specified account, or lists the files in an account.

ENTER	ACTION OR PROMPT
2	The program prompts you for a listing device. If you select '3', the program prompts for a printer device number. The program prompts for the diskette drive number in which you should insert your session backup Master.
The diskette drive #	The program displays the Master diskette label then prompts for the hard disk drive to RECOVER to.
Hard disk drive # <CR>	The program prompts for the 'Old accountname'. This name must exist on master diskette directory.

ENTER	ACTION OR PROMPT
Old accountname <CR>	The program lists four options for recovery:
1. The entire account	Lists all files in an account and the accountname they will recover to. This procedure does not copy data into the files; it names the files and CReates them into a new, non-existing account.
2. List the files	Displays the name and file size of the files in the 'Old account name'. Will then re-prompt for Selections 1, 3, or 4.
3. List the name and ask for confirmation	Prompts for a new account name, then displays the name of each file in the old account followed by a question mark. A reply of 'N' indicates no desire to recover the file; 'Y' indicates you do want to recover the file.
4. Name a specific file	Prompts for an individual file name for recovery to the hard disk. The account name must already exist on hard disk but the filename to recover into must be new.
New Accountname <CR>	Enter the new account name for options 1,3, or 4. If you select option 4 the program prompts for an old filename.

<table>
<tr><th>ENTER</th><th>ACTION OR PROMPT</th></tr>
<tr><td>New Filename &lt;CR&gt;</td><td>The program prompts for an existing accountname to recover into.</td></tr>
<tr><td>Accountname &lt;CR&gt;
(must already exist)</td><td>Selection 4 will not recover a file into an existing file if the file is RO (Read Only) or DP (Delete Protected). You can recover to a new filename that does not exist on the directory. The program then prompts:</td></tr>
<tr><td>1. Next file or account</td><td>To repeat the process for a file or account.</td></tr>
<tr><td>2. Recover files</td><td>To begin the recovery process. A dot (.) prints for each sector copied.</td></tr>
<tr><td>3. Top level menu</td><td>To return to the original Recover menu.</td></tr>
<tr><td colspan="2">NOTE
If you Control-C out of option 2, or if you use option 3 to exit to HDOS before the message that the file recovery is complete, your recovery files are CReated on hard disk, but contain no data. If you try and load these files from HBASIC you get the error: NO FILE OR PROGRAM TOO LONG.</td></tr>
<tr><td>The desired option &lt;CR&gt;</td><td>If you choose selection 2 the system prompts you for a diskette drive number for the Master Backup diskette.</td></tr>
</table>

ENTER	ACTION OR PROMPT
Diskette drive # <CR>	When selection 2, Recover Files completes you receive the system message below and the program returns you to HDOS command level.

```
-----------------------------------------------------------
Recovered "N" sectors to file "filename 2, accountname"
you may remove the diskette from drive "d".  File
recovery complete.  Thank you for waiting.
-----------------------------------------------------------
```

4.3.4 EXPLANATION

The Explanation option on the RECOVER menu presents a short description of the major alternatives available with the RECOVER program.

4.4 USING TOTREC

The Total Recovery Program, performs three major functions. First, TOTREC assumes that the hard disk has been completely erased or contains meaningless data, and prepares the disk for new data. Second, it reinstates the disk directory from the last session done in the backup series. Third, the program recovers all files as described for RECOVER.

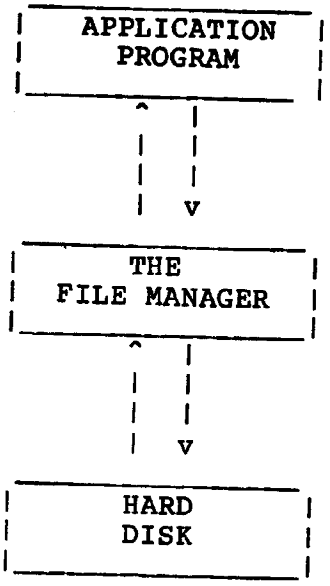
ENTER	ACTION OR PROMPT
	Insert working copy of HDOS SYSTEM DISKETTE in diskette drive 1.

ENTER	ACTION OR PROMPT
GO TOTREC,1 <CR>	The program asks if you really want to erase all the files and accounts on the hard disk. Type "YES" to proceed with a Total Recovery of your files. A response of "NO" does not end the TOTREC program. It does leave your hard disk directory intact, and proceeds as if a normal RECOVER is in progress.
YES <CR>	The program prompts for your hard disk drive number.
Hard disk drive # <CR>	The program prompts for an option for a listing device.
Listing device #	The program prompts for a recovery of your SYSTEM account directory from the HDOS INITIAL RECOVERY DISKETTE in floppy disk drive #1.
1	The program prompts: 1. Recover all accounts. 2. Specify accounts. 3. Specify exceptions.

ENTER	ACTION OR PROMPT
The option #	Option 1 generates a message indicating the files to be recovered into the SYSTEM account and the number of sectors associated with each file. Options 2 and 3 do the same, only listing individual accounts or exceptions. The System prompts you to insert the HDOS SYSTEM DISKETTE, (remember to use your working copy) into a specific drive. Press <CR> to continue.
<CR>	The program prompts for the diskette drive number for the Master backup diskette. This diskette should be from the last session of the series you wish to RECOVER from.
The diskette drive #	The system displays a message for every file recovered, and returns you to HDOS when the recovery is complete.

5.1 INTRODUCTION

The File Manager allows access to hard disk-resident data, and maintains the data structures necessary to support data storage on the hard disk. The File Manager is a permanent part of the HDOS, and is always resident in memory. The flow of data through the File Manager is shown below. Generally, the File Manager is of interest only to system programmers.



The operation codes and data structures maintained by the File Manager are listed in Appendix A.

5.2 CALLING THE FILE MANAGER

To execute an HDOS File Manager operation, use the following assembly language sequence:

```
.
.           load parameters.
.
.
MVI A, FMxxx      ;place the specified operation
                  ;code in the A register
CALL   FMNGR      ;call the HDOS File Manager.
.
.          on return from the  call, the Z80's Z
.          flag is set if no exceptional
.          situation is encountered; otherwise,
.          the Z flag is cleared.
```

With the exception of FMABT (Abort), FMFIN (Finish), and sometimes FMLX, control is always returned to the calling program.

5.3 FILE MANAGER OPERATION CODES

The operation code in the A register tells the File Manager what operation to perform. The operation codes are listed in Table 5-1. Following the table is a description of each operation and its associated input and output. Note that File Manager Message codes are listed in Appendix B.

Table 5-1
Operation Code List

Operation code	Definition
FMCRE	Create a file
FMDEL	Delete a file
FMOPN	Open a file
FMCLO	Close a file
FMTFR	Transfer data to or from a file
FMLOK	Look up information on a file
FMLAC	Look up information on an account
FMCTY	Change file type
FMCAT	Change file attributes
FMCFS	Change file size
FMCAP	Change auto-load-and-go pathname
FMABT	Abort the Calling Program
FMRST	Reset the File Manager
FMLX	Load and execute a type 1 file
FMFSZ	Return file's size
FMCA	Create an account
FMDA	Delete an account
FMSDV	Switch default value
FMCM	Compose message
FMFIN	Finish processing
FMRDS	Return Disk Statistics
FMADE	Access directory entry
FMRDV	Return unit and account ID defaults
FMCAN	Change account name
FMPD	Power down a unit
FMPU	Power up a unit
FMFBU	Flag buffer used
FMINI	Initialize the File Manager

The equates for the File Manager message and operation codes are found in Appendix E, under File Manager Definitions and in the file EQUS in the SYSTEM account on hard disk.

5.4 OPERATION CODE DESCRIPTION

FMCRE

Creates a file on the hard disk. Once a file is created you can open it with the FMOPN code and allocate disk space to it with the FMCFS code.

Input:
    IX - address of the Create Information Block.

Output:
    IX - (preserved)
    A - message code

FMDEL

Deletes a file from the directory. The file must not have any space allocated to it. An attempt to delete an open file or one that has disk space allocated to it results in an error.

Input:
    HL - pathname address

Output:
    A - message code
    HL - address + 1 of last byte of pathname accessed by the File Manager. If there was no error in the pathname, HL points to pathname terminator +1; if there is an error, HL points to the bad character +1.

FMOPN

Opens a disk file. This makes the file's sectors accessible to a program (See FMTFR.)

Input:
    IX - address of an area of memory OFBSZ bytes long in which the Open File Block (OFB) will be constructed.
    HL - pathname address

Output:
A - message code
A = MOK: H - open count: the number of times the file is currently open.
B - file type
CDE - type-dependent data
IY - address of the File Directory Entry for the opened file

FMCLO

Closes a disk file. The space occupied by the OFB may be used for other purposes after return from FMCLO.

Input:
IX - the address of the OFB

Output:
IX - the address of a block of memory, OFBSZ bytes long
A - message code

FMTFR

Performs data transfer operations on sectors of a file. The specific operation to be performed is given in the first byte of the Transfer Command Block (TCB).

Input:
IX - OFB address
IY- Transfer Command Block address

Output:
IX - (preserved)
IY - Transfer Command Block address
A - message code

FMLOK

Searches the directory for the specified pathname's directory entry and returns the memory address of the directory entry and entry number.

Input:

```
    HL -  pathname address
```

Output:

```
    HL -  address of last byte of pathname accessed +1
    A  -  message code
          A= MOK:
             IY- directory entry memory address
             DE - entry number
```

FMLAC

Looks up the specified account name's entry in the directory.

Input:

```
    HL - account name address
```

Output:

```
    HL - address of last byte of pathname accessed +1
     A - message code
     A - MOK:  IX - directory entry memory address
               DE - entry number
```

FMCTY

Changes the type and type-dependent information of file. The file must not be open. If the new file is type FTMI, then DE of the type-dependent information is the file's load-and-execute address.

Input:

```
    B   - new file type
    CDE - type-dependent information
    HL  - pathname address
```

Output:

```
    HL - address of last byte of pathname accessed +1
    A - message code
```

FMCAT

Changes the attrubutes of a file. The file must not be open.

Input:
- B - new file attribute(s)
- C - attribute(s) mask
- HL - pathname address

Output:
- HL - address of last byte of pathname accessed +1
- A - message code

FMCFS

Changes the amount of disk space allocated to a file. After you have created a file and before using it, use FMCFS to allocate disk space to it. Before you can delete a file, you must remove its disk space with FMCFS.

Input:
- IX - the address of the file's OFB
- HL - the file's new size in 512-byte sectors
- DE - the number of bytes used in the last sector (1 to 512)

Output:
- A - message code
- IX - (preserved)

FMCAP

Changes the auto-load-and-execute pathname. This pathname is used by FMFIN and FMABT.

Input:
- HL - pathname address

Output:
- A - message code
- HL - address of last byte of pathname accessed +1

FMABT

Is typically used to terminate execution of a program when an unexpected error has occurred. It prints a message based on the message code in B and loads and executes the auto-load-and-execute file. The message is produced with the CHO routine using device code 0. FMABT does not return to the caller.

Input:
B - message code

Output:
none, does not return to the caller

FMRST

Closes all files and resets the File Manager. FMRST returns to the calling program when complete. The default drive is set to 101; the default account ID is set to 1.

Input:
none

Output:
A - message code

FMLX

Loads and executes the specified load-and-execute file. Only the first 128 blocks can be loaded.

```
Input:
    HL - pathname address
    DE - moved to HL for the loaded program
    B  - option code:
         B[6 to 0] = 0: return on load errors
         B[6 to 0] = 1: use FMABT on load errors
         B[6 to 0] = 2: return on load errors and
                        do not exeucte the loaded
                        program.
         B[7]      = extended path option bit
```

When bit 7 (the extended path option bit) is set, it indicates that if the pathname given does not resolve to an existing file, then the pathname will be evaluated as if the default account ID number is 1 and the drive number is 101. FMFIN and FMABT use FMLX with this option set to load the command processor.

Examples:

When bit 7 = 1:

QUEUE - searches for QUEUE on the default account. If not found, then account number 1 on Drive 101 is searched.

HEX,103 - searches for HEX on the default account on Drive 103. If not found, account number 1 on Drive 101 is searched.

DATA,A1,102 - searches for DATA on account A1 on Drive 102. If not found, account A1 is searched for DATA on Drive 101.

STUFF,ACCOUNT - searches for STUFF on account ACCOUNT on the default drive. If not found, then account ACCOUNT on drive 101 is searched.

```
Output to calling program (if B[6 to 0]=2)
    A - message code
        A = MOK: IX - load-and-execute address
```

```
Output to loaded program (if B[6 to 0] <> 2):
    A - system dispatch table upper byte
    SP - same as the caller's SP with return address
         removed from the top of the stack.
    DE - the last address loaded + 1
    HL - the input value from DE
```

FMFSZ

Returns the size of an open file

```
Output:
    IX - (preserved)
    A - message code
        A = MOK:  HL - the size of the file in 512-byte
                       sectors
                  DE - the number of bytes used in the
                       last sector


                  NOTE: If the file is allocated no disk
                  space, HL is 0 and DE is meaningless.
```

FMCA

Creates an account.

```
Input:
    HL - address of the account name

Output:
    A  - message code
    HL - address of last byte of pathname accessed +1
    DE - if A = MOK, then DE is the new account's
         account ID number
```

FMDA

Deletes an account. The account must not contain any files.

Input:
- HL - the address of the account name

Output:
- HL - address of last byte of pathname accessed +1
- A - message code

FMSDV

Sets the default account ID and the default drive number. It also returns the previous values for these defaults.

Input:
- B - drive number: 101 to 104
- HL -account ID number: 1 to 65535

Output:
- A - message code
  - A = MOK:
    - B - old default drive number: 101 to 104
    - HL - old default account ID 1 to 65535

FMCM

Composes a message based on the message code in the B register.

Input:
- B - message code

Output:
- A - message code
  - A = MOK: HL - address of the message
    - BC - length of the message

FMFIN

Is called when a program is finished executing. FMFIN loads and executes the default auto-load-and execute file and executes it. FMFIN does not return to the calling program.

Input:
    none

Output:
    none, does not return to the caller

FMRDS

Returns disk statistics on the hard disk label, the number of DIBs in use, and the number of bad DIBs.

Input:

B - drive number: 101 to 104

Output:

A - message code

```
A=MOK:IX - address of the disk label buffer
      DE - number of bad hunks
      BC - number of DIBs in usee
```

FMADE

Returns a memory pointer to the directory entry for the specified directory entry number.

Input:
    DE - directory entry number
    B - drive number

Output:
    A - message code
        A = MOK: IY - address of the directory entry

FMRDV

Returns the default values for the drive and account ID.

Input:
    none

Output:
```
A - message code
    A = MOK: B - drive number: 101 to 104
             HL - account ID number: 1 to 65535
```

FMCAN

Changes an account name.

Input:
```
DE  - address of the name to which the old name
      should be changed.
HL - address of the old account name
```

Output:
    A - message code

FMPD

Powers down any specified hard disk drive.

Input:
    B - drive number: 101 to 104

Output:
    A - message code

FMPU

Powers up any specified hard disk drive.

Input:
    B - the drive number (101 to 104)

Output:
    A - message code

FMFBU

Is used before any program uses the File Manager's internal buffer. The diskette software uses FMFBU; it should not be considered for general use by the application programmer.

Input:
none

Output:
A - message code

FMINI

Is called by the bootstrap program to initialize File Manager. This operation also powers up all hard disk drives.

Input:
none

Output:
A - message code

PERSONALIZING HDOS 6

6.1 INTRODUCTION

Any changes to the Operating System that customize or set options for particular hardware configurations should be done on your copy of the HDOS 2.OH SYSTEM DISKETTE. If you decide to make further changes, do them on a copy of the working copy, to provide an appropriate Backup and Recovery cycle.

The HBASIC program, SYSGEN, provides an easy way to perform any personalization you are likely to need. This program is included in your SYSTEM account after you perform the INITIAL SYSTEM STARTUP procedure in Chapter 1.

The entry point and flags necessary to customize HDOS are listed in Appendix F.

6.2 PERSONALIZING THE CONFIGURATION BYTE

If your system has any single sided, normal-stepping (double-density) diskette drives, rather than double-sided fast-stepping (quad capacity) drives, you must personalize the configuration byte on your working copy of the HDOS System Diskette.

The byte configuration is:

Bit	7	6	5	4	3	2	1	0
Drive	1	2	3	4	4	3	2	1
	2-sided				Fast-stepping			

Bits 7 and 0 correspond to Drive 1, bits 6 and 1 correspond to Drive 2, etc.

Initially, the value in the byte is FF, identifying four double-sided, fast-stepping drives. If all your drives are that type, there is no need for change. If, however, any of your disk drives is a single-sided drive, change the contents of the corresponding bits to 0. Use the chart in Appendix C to figure the appropriate Hex number.

6.3 PARALLEL I/O PORT USE NOTES

Standard I/O routines for device 0 (standard serial port), device 1 (second serial port) and device 2 (parallel output port) are included in HDOS. The standard parallel output routine controls a Centronix-type parallel printer as device 2. Connect it to P3 of the final hard disk drive rather than the parallel output port of the HORIZON. If you write your own parallel I/0 routine consider the status of the hard disk controller as well as that of the peripheral. The examples of possible program modifications below assume that the headers for input-output operation are in controller 1 and the peripheral is connected to controller 1.

Input

A. The SPARE signal is not used by the peripheral. In this case it is only necessary to deselect all controllers and use the old program.

```
MVI A, 020H
OUT 6          ;Clear PO-FLAG flip-flop (output)
MVI A, 0FFH
;OUT 0         ;Deselect all controller
:Old program
```

B. The SPARE signal is used.

```
MVI A, 020H
OUT 6          ;Clear PO-FLAG flip-flop
MVI A, 0FEH
OUT 0          ;Select controller 1
MVI A,0E1H
OUT 0          ;Set I/O mode
;Old program
MV  A, 0FFH    ;Clear I/O mode; deselect controller
OUT 0
```

Output

A. The FLAG output is used to strobe the data, using the flip-flop contained in the controller.

```
MVI A, 020H
OUT 6            ;Clear FLAG flip-flop (Horizon)
MVI A,0FEH
OUT 0            ;Select controller 1
MVI A,0E1H
OUT 0            ;Set I/O mode
MVI A, DATA
OUT 0            ;Load proper data onto bus
MVI A, 060H
OUT 6            ;Set PO-FLAG flip-flop (Horizon)
                 ;This enables data onto the output port
IN 0             ;Clear controller FLAG flip-flop,
                 ;and generate FLAG output from
                 ;controller
ALPHA:IN 2
BIT3,A
JNZ ALPHA        ;Wait for ACK from peripheral to clear
                 ;controller FLAG flip-flop

MVI A,020H
OUT 6            ;Clear FLAG flip-flop; put output
                 ;data into high impedance

MVI A,0FFH
OUT 0            ;Deselect controller and clear I/O mode
```

B. The DATA MSB is used to strobe the peripheral and the FLAG output is not used.

```
MVI A,020H
OUT 6            ;Clear PO-FLAG flip-flop
MVI A, 0FEH
OUT 0            ;Select controller 1
MVI A,0E1H
OUT 0            ;Set I/O mode
MVI A, 60 H
OUT 6            ;Set FLAG, enable data to output port
MVI A, DATA.OR.080H
OUT 0            ;Data with MSB = 1
MVI  A,  DATA.AND.07FH
OUT 0            ;Data with MSB = 0
MVI  A,  DATA.OR.080H
OUT 0            ;Data with MSB = 1
MVI A, 20H
OUT 6            ;Clear FLAG flip-flop; put data
                 ;to high impedence
MVI A,0FFH       ;Deselect controller and clear I/O mode
```

Appendix A

HDOS COMMAND SUMMARY

Name	Command	Parameters
Account Create	AC	account
Account Delete	AD	account
Account List	AL	[#n] [d]
Account Rename	AR	account TO newaccountname
Account Set	AS	account
Copy Diskette	CD	dl TO d2 {sides}
Copy File	CF	pathnamel TO pathname2 {dens}
	CF	pathnamel CR pathname2 {len} {dens} [alloc]
	CF	pathnamel SL pathname2 [len]
Create File	CR	pathname len {daddr} {dens} [alloc]
Display ASCII	DA	{[#n]} region
Delete File	DE	pathname
Display Hex	DH	{[#n]} region
Device Output	DO	{[#n]} {[char]}
Display & Substitute	DS	raddr
Examine Byte	EB	{[#n]} bval
Examine Memory	EM	{[#n]} raddr
Examine Port	EP	{[#n]} paddr
Fill Memory	FM	region bval {[,bval,bval...}]
Go (Load & Execute)	GO	pathname {[args]}
Help	HE	command
Initial Load	IL	
Initialize Diskette	IN	d {dens}
Jump	JP	raddr {[args]}
Load File into RAM	LF	pathname raddr
List File Directory	LI	{[#n]} [account] {d}
Multiple Copy	MC	{d} [account] TO {d} [account] {len} {dens} [alloc]
	MC	{d} [account] YN {d} [account] {len} {dens} [alloc]
	MC	{d} [account] CR account [alloc]
Multiple Delete	MD	{d} [account]
Multiple List	ML	[#n]
Move Memory	MM	region raddr
Output Device	OD	{[#n]}
Off	OF	[d]
Set Listing Page Length	PA	n
Port Output	PO	bval TO paddr

Appendix A

HDOS COMMAND SUMMARY (continued)

Name	Command	Parameters
Read Disk to RAM	RD	len {dens} FROM daddr{,d} {[OF pathname]} TO raddr
Rename Diskette File	RN	filename1, d to filename2
Reset	RS	
Save File from RAM	SF	pathname raddr
Set Length	SL	pathname len
Search Memory	SM	{[#n]} region bval {[,bval,bval...]}
Print Disk Statistics	ST	[d]
Set Auto-Execute File	SX	pathname
Type File	TY	pathname [filetype] {[raddr]} [attr...]
Verify Memory	VM	{[#n]} region raddr
Write RAM to Disk	WR	len {dens} FROM raddr TO daddr{,d} {[OF pathname]}
Work Space	WS	{[#n]}

Appendix B

ERROR CODES

MFDOS RESULT CODES

Code	Symbol	Meaning
1	MFSNF	Sync not found
2	MFCRC	CRC error
3	MFVFY	Verify compare error
4	MFNIP	Drive or diskette not available
5	MFDMM	Density mismatch on read or verify
6	MFWRP	Attempt to write on protected diskette
7	MFCCX	Control-C detected from terminal
8	MFIDW	Illegal call to DWRIT
9	MFIDN	Illegal drive number
10	MFIDA	Illegal disk address
11	MFITL	Illegal transfer length
12	MFIDC	Illegal command to DCOM
13	MFTDM	Track density mismatch error

FILE MANAGER MESSAGE CODES

Code	Symbol	Meaning
0	MOK	Operation okay
20	MANE	Account not empty
21	MDDUP	Attempt to create duplicate directory
22	MDFUL	Directory full
23	MDFND	Matching directory entry found
24	MDBAD	Bad directory structure
25	MDFNF	File name not found in directory
26	MFANF	Account name not found in directory
27	MCADS	Can't allocate requested disk space
28	MOFUL	Open File Table full; can't open file
29	MOAVL	Entry available in the Open File Table
30	MILDN	Illegal decimal number
31	MILFN	Illegal file name
32	MILAN	Illegal account name
33	MILUN	Illegal unit number
34	MILID	Illegal account ID number
35	MWRP	Attempt to write on write-protected file
36	MDEP	Attempt to delete a delete-protected file
37	MADEP	Attempt to delete a delete-protected account
38	MCCPF	Attempt to change a protected field in File Manager
39	MPARA	Parameter invalid or out of range
40	MFRT	Improper file type specified

Appendix B - ERROR CODES

FILE MANAGER MESSAGE CODES (continued)

Code	Symbol	Meaning
41	MFNO	File not open; open file required
42	MFOPN	File open; unopened file required
43	MFAIL	General failure, usually indicates hardware malfunction
44	MEOLA	End of list with available space
45	MEOL	End of list with no available space
46	MIFMR	Illegal File Manager request
47	MFZNZ	File size not zero
48	MIFZ	Illegal file size
49	MEOF	End of fle reached during data transfer
50	MPEOF	Transfer attempted beyond end of file
51	MMEMP	Memory protect violation
52	MUNPU	Unit not powered up
53	MNYI	Operation not yet implemented
54	MFMO	File multiply opened
55	MDLRE	Disk level revision error
56	MDNSL	Drive label mismatch error
57	MDNSS	Drive size mismatch error
103	MBUFRD	Buffer error
104	MMIPLS	Missing index pulse
105	MSHDR	PLL sync error on read
106	MRDFL	Failure to format drive
107	MRCER	Drive error during command execution
108	MVCRCE	CRC error during verify
109	MVDATE	Compare error in data during verify
110	MDCRCE	Data CRC error
111	MRDSHE	CRC error on read sector header
112	MFWSOR	Found wrong sector during read or verify
113	MDWRTE	Write unsafe or attempt to write on protected cylinder
114	MCNTFL	read/write flip-flop will not set in controller
115	MILLDA	Illegal disk address
116	MHDCRC	CRC error in header during position verify
117	MCYLER	Drive on wrong cylinder
118	MDSLER	Head select error
119	MDERDS	Drive error during seek
120	MBADRV	Drive number too big
121	MTSHDR	Target sector has CRC error in header
122	MDRDFL	Failure in drive read electronics
123	MCNFTS	Can't find target sector
124	MDWNR	Drive went not ready after command started
125	MCNPR	Controller not there
126	MDNACC	Drive not ready for command
127	MDNRDY	Drive not ready - out of speed

CONVERSION TABLE

DECIMAL-ASCII-HEX-BINARY CONVERSION TABLE

The following table is intended to ease the task of conversion between the various numeric representations commonly used in programming, as well as between numbers (of any kind) and the ASCII character code.

Note that the ASCII character set only goes as far as decimal 127 (7FH, 01111111 B). Also, many "characters" in ASCII are nonprinting CONTROL CHARACTERS. Whenever a code corresponds to a printable character, that will be given. In the case of control characters, a description or name for the special character will be given in parentheses.

DECIMAL	HEX	BINARY	ASCII
0	00H	00000000	(NUL)
1	01H	00000001	(CONTROL-A)
2	02H	00000010	(CONTROL-B)
3	03H	00000011	(CONTROL-C)
4	04H	00000100	(CONTROL-D)
5	05H	00000101	(CONTROL-E)
6	06H	00000110	(CONTROL-F)
7	07H	00000111	(CONTROL-G, RINGS BELL)
8	08H	00001000	(CONTROL-H, BACKSPACE)
9	09H	00001001	(CONTROL-I, TAB)
10	0AH	00001010	(CONTROL-J, LINEFEED)
11	0BH	00001011	(CONTROL-K)
12	0CH	00001100	(CONTROL-L, FORMFEED)
13	0DH	00001101	(CONTROL-M, CARRIAGE RETURN)
14	0EH	00001110	(CONTROL-N)
15	0FH	00001111	(CONTROL-O)
16	10H	00010000	(CONTROL-P)
17	11H	00010001	(CONTROL-Q)
18	12H	00010010	(CONTROL-R)
19	13H	00010011	(CONTROL-S)
20	14H	00010100	(CONTROL-T)
21	15H	00010101	(CONTROL-U)
22	16H	00010110	(CONTROL-V)
23	17H	00010111	(CONTROL-W)
24	18H	00011000	(CONTROL-X)
25	19H	00011001	(CONTROL-Y)
26	1AH	00011010	(CONTROL-Z)
27	1BH	00011011	(ESCAPE)
28	1CH	00011100	(NON-PRINTING)
29	1DH	00011101	(NON-PRINTING)
30	1EH	00011110	(NON-PRINTING)
31	1FH	00011111	(NON-PRINTING)
32	20H	00100000	(SPACE)
33	21H	00100001	!
34	22H	00100010	"
35	23H	00100011	#
36	24H	00100100	$
37	25H	00100101	%
38	26H	00100110	&
39	27H	00100111	'

DECIMAL	HEX	BINARY	ASCII
40	28H	00101000	(
41	29H	00101001	)
42	2AH	00101010	*
43	2BH	00101011	+
44	2CH	00101100	,
45	2DH	00101101	-
46	2EH	00101110	.
47	2FH	00101111	/
48	30H	00110000	0
49	31H	00110001	1
50	32H	00110010	2
51	33H	00110011	3
52	34H	00110100	4
53	35H	00110101	5
54	36H	00110110	6
55	37H	00110111	7
56	38H	00111000	8
57	39H	00111001	9
58	3AH	00111010	:
59	3BH	00111011	;
60	3CH	00111100	<
61	3DH	00111101	=
62	3EH	00111110	>
63	3FH	00111111	?
64	40H	01000000	@
65	41H	01000001	A
66	42H	01000010	B
67	43H	01000011	C
68	44H	01000100	D
69	45H	01000101	E
70	46H	01000110	F
71	47H	01000111	G
72	48H	01001000	H
73	49H	01001001	I
74	4AH	01001010	J
75	4BH	01001011	K
76	4CH	01001100	L
77	4DH	01001101	M
78	4EH	01001110	N
79	4FH	01001111	O
80	50H	01010000	P
81	51H	01010001	Q
82	52H	01010010	R
83	53H	01010011	S
84	54H	01010100	T
85	55H	01010101	U
86	56H	01010110	V
87	57H	01010111	W
88	58H	01011000	X
89	59H	01011001	Y
90	5AH	01011010	Z
91	5BH	01011011	[
92	5CH	01011100	\
93	5DH	01011101	]

DECIMAL	HEX	BINARY	ASCII
94	5EH	01011110	↑ OR
95	5FH	01011111	_
96	60H	01100000	
97	61H	01100001	a
98	62H	01100010	b
99	63H	01100011	c
100	64H	01100100	d
101	65H	01100101	e
102	66H	01100110	f
103	67H	01100111	g
104	68H	01101000	h
105	69H	01101001	i
106	6AH	01101010	j
107	6BH	01101011	k
108	6CH	01101100	l
109	6DH	01101101	m
110	6EH	01101110	n
111	6FH	01101111	o
112	70H	01110000	p
113	71H	01110001	q
114	72H	01110010	r
115	73H	01110011	s
116	74H	01110100	t
117	75H	01110101	u
118	76H	01110110	v
119	77H	01110111	w
120	78H	01111000	x
121	79H	01111001	y
122	7AH	01111010	z
123	7BH	01111011	{
124	7CH	01111100	|
125	7DH	01111101	}
126	7EH	01111110	~
127	7FH	01111111	(DELETE, RUB OUT)
128	80H	10000000	
129	81H	10000001	
130	82H	10000010	
131	83H	10000011	
132	84H	10000100	
133	85H	10000101	
134	86H	10000110	
135	87H	10000111	
136	88H	10001000	
137	89H	10001001	
138	8AH	10001010	
139	8BH	10001011	
140	8CH	10001100	
141	8DH	10001101	
142	8EH	10001110	
143	8FH	10001111	
144	90H	10010000	
145	91H	10010001	
146	92H	10010010	
147	93H	10010011	

DECIMAL	HEX	BINARY	ASCII
148	94H	10010100	
149	95H	10010101	
150	96H	10010110	
151	97H	10010111	
152	98H	10011000	
153	99H	10011001	
154	9AH	10011010	
155	9BH	10011011	
156	9CH	10011100	
157	9DH	10011101	
158	9EH	10011110	
159	9FH	10011111	
160	A0H	10100000	
161	A1H	10100001	
162	A2H	10100010	
163	A3H	10100011	
164	A4H	10100100	
165	A5H	10100101	
166	A6H	10100110	
167	A7H	10100111	
168	A8H	10101000	
169	A9H	10101001	
170	AAH	10101010	
171	ABH	10101011	
172	ACH	10101100	
173	ADH	10101101	
174	AEH	10101110	
175	AFH	10101111	
176	B0H	10110000	
177	B1H	10110001	
178	B2H	10110010	
179	B3H	10110011	
180	B4H	10110100	
181	B5H	10110101	
182	B6H	10110110	
183	B7H	10110111	
184	B8H	10111000	
185	B9H	10111001	
186	BAH	10111010	
187	BBH	10111011	
188	BCH	10111100	
189	BDH	10111101	
190	BEH	10111110	
191	BFH	10111111	
192	C0H	11000000	
193	C1H	11000001	
194	C2H	11000010	
195	C3H	11000011	
196	C4H	11000100	
197	C5H	11000101	
198	C6H	11000110	
199	C7H	11000111	
200	C8H	11001000	
201	C9H	11001001	

DECIMAL	HEX	BINARY	ASCII
202	CAH	11001010	
203	CBH	11001011	
204	CCH	11001100	
205	CDH	11001101	
206	CEH	11001110	
207	CFH	11001111	
208	D0H	11010000	
209	D1H	11010001	
210	D2H	11010010	
211	D3H	11010011	
212	D4H	11010100	
213	D5H	11010101	
214	D6H	11010110	
215	D7H	11010111	
216	D8H	11011000	
217	D9H	11011001	
218	DAH	11011010	
219	DBH	11011011	
220	DCH	11011100	
221	DDH	11011101	
222	DEH	11011110	
223	DFH	11011111	
224	E0H	11100000	
225	E1H	11100001	
226	E2H	11100010	
227	E3H	11100011	
228	E4H	11100100	
229	E5H	11100101	
230	E6H	11100110	
231	E7H	11100111	
232	E8H	11101000	
233	E9H	11101001	
234	EAH	11101010	
235	EBH	11101011	
236	ECH	11101100	
237	EDH	11101101	
238	EEH	11101110	
239	EFH	11101111	
240	F0H	11110000	
241	F1H	11110001	
242	F2H	11110010	
243	F3H	11110011	
244	F4H	11110100	
245	F5H	11110101	
246	F6H	11110110	
247	F7H	11110111	
248	F8H	11111000	
249	F9H	11111001	
250	FAH	11111010	
251	FBH	11111011	
252	FCH	11111100	
253	FDH	11111101	
254	FEH	11111110	
255	FFH	11111111	

APPENDIX D

RAM ALLOCATION TABLE

The table below gives the allocation of the 64K-byte RAM address space for the standard HDOS system software and hardware.

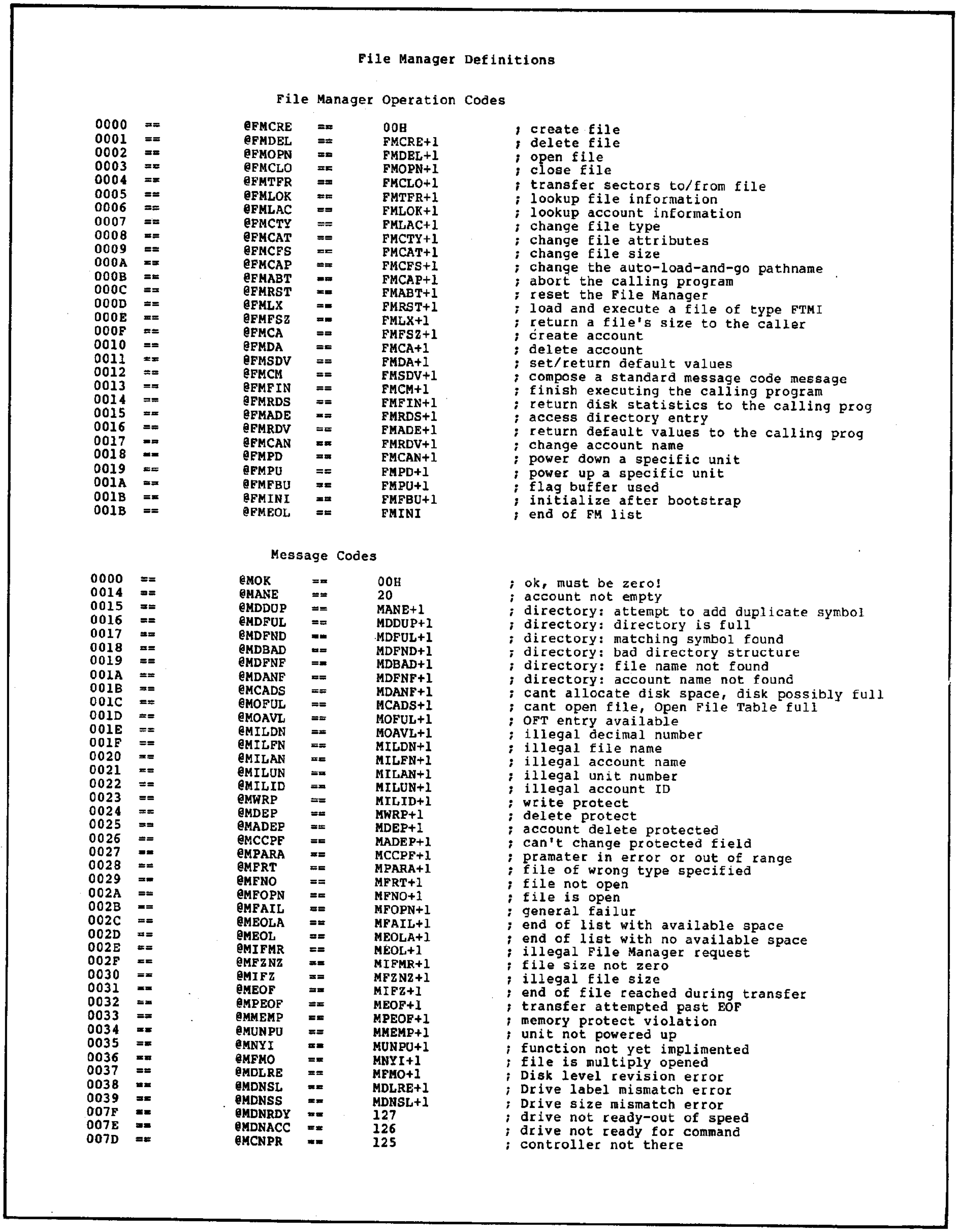
RAM ADDRESS (Hex)	CONTENTS	DESCRIPTION
Resident System		
0100	Dispatch Table	A table of entry points.
0200	2-block buffer	Shared by File Manager and the diskette DOS.
0400	I/O Routines	May be modified by the user.
0500	HDCOM	Origin of the HDCOM, not necessarily the entry point.
0D00	File Manager	Origin of the File Manager, not necessarily the entry point.
Transient System		
1F00	MFDOS	Diskette DOS and the Line Editor. (Includes a Jump Table at 2000 to enable compatability with some existing North Star software.)
2600	Command Processor	Origin of the HDOS Command Processor.

Whenever you initialize (boot) the system, either with a power up or the IL command, the Hard Disk Operating System is read from the HDOS 2.0.0H SYSTEM DISK and loaded into RAM at the addresses shown above. Everything from address 1F00 also resides on the hard disk. HBASIC loads at 2600, overwriting the HDOS Command Processor. When you return to the system, everything from 1F00 to the end of the table loads from the hard disk and overwrites the current contents of RAM at the same addresses. This method of overwriting the same areas for HBASIC and the Command Processor gives you more memory for application programs.

APPENDIX E

HDOS SYMBOLS AND DATA STRUCTURES

```
                         EQUS
                      JULY 19, 1982

   These equates include revision 2.0 of the disk label structure.

    This file contains the equates for use in all modules of the
              North Star Hard Disk Operating System.

0023  ==       @NTRAC    ==     35            ; Number of tracks per side on a Micro Disk
0059  ==       @ZTRAC    ==     18+35+35+1    ; Initial track counter value for Micro Disks
0007  ==       @MAXIO    ==     7             ; Maximum legal I/O device number
0050  ==       @NLINE    ==     80            ; Length of input line for Command Processor
001A  ==       @DFSTP    ==     26            ; Default processor speed constant (Z80A)

                     SYSTEM DISPATCH TABLE ADDRESSES

0100  ==       @TRAKT    ==     DSPCH         ; Micro Disk track table
0104  ==       @REVN     ==     DSPCH+4       ; Sequential revision number
0105  ==       @CNFG2    ==     DSPCH+5       ; Secondary configuration byte
0106  ==       @SUNIT    ==     DSPCH+6       ; Last used Micro Disk drive
0107  ==       @OFTEN    ==     DSPCH+7       ; Poling vector
010A  ==       @CBOOT    ==     DSPCH+10      ; Entry point from boot PROM
010D  ==       @CHO      ==     DSPCH+0DH     ; Character output routine
0110  ==       @CHI      ==     DSPCH+10H     ; Character input routine
0113  ==       @INIT     ==     DSPCH+13H     ; Terminal initialization routine
0116  ==       @CON      ==     DSPCH+16H     ; Control-C check
0119  ==       @HDERR    ==     DSPCH+19H     ; Nonrecoverable Micro Disk error vector
011C  ==       @DLOOK    ==     DSPCH+1CH     ; Micro Disk file lookup routine
011F  ==       @DWRIT    ==     DSPCH+1FH     ; Micro Disk directory update
0122  ==       @DCOM     ==     DSPCH+22H     ; Lowest level Micro Disk driver
0125  ==       @DLIST    ==     DSPCH+25H     ; Micro Disk directory lister
0128  ==       @RSTRT    ==     DSPCH+28H     ; System restart address
012B  ==       @RWCHK    ==     DSPCH+2BH     ; Read after write and interrupt flags
012C  ==       @DOSER    ==     DSPCH+2CH     ; Micro Disk argument error vector
012F  ==       @DEN      ==     DSPCH+2FH     ; Micro Disk density flag
0130  ==       @AUTOS    ==     DSPCH+30H     ; Command processor autostart flag
0133  ==       @PAGES    ==     DSPCH+33H     ; Personalization byte, number of console display lines
0134  ==       @CONFG    ==     DSPCH+34H     ; Personalization byte, Micro Disk drive combination
0135  ==       @RESLT    ==     DSPCH+35H     ; Storage for result of last disk operation
0136  ==       @HDEMC    ==     DSPCH+36H     ; Type of last HDCOM error
0137  ==       @HDEDA    ==     DSPCH+37H     ; Sector address of last HDCOM error
0139  ==       @HDEDN    ==     DSPCH+39H     ; Drive number of last HDCOM error
013A  ==       @MDCB     ==     DSPCH+3AH     ; Personalization byte, address of Micro Disk Controller
013B  ==       @FTPTM    ==     DSPCH+3BH     ; Storage for processor speed indicator
013C  ==       @HMEM     ==     DSPCH+3CH     ; Memory limit indicator
013D  ==       @ADEV     ==     DSPCH+3DH     ; Additional output device number
013E  ==       @AOUT     ==     DSPCH+3EH     ; Character output bypassing additional device feature
0141  ==       @ISTAT    ==     DSPCH+41H     ; Input device status check
0144  ==       @OSTAT    ==     DSPCH+44H     ; Output device status check
0147  ==       @LINED    ==     DSPCH+47H     ; Line input editor
014A  ==       @FMNGR    ==     DSPCH+4AH     ; Hard disk file manager entry point
014D  ==       @HDCOM    ==     DSPCH+4DH     ; Lowest level hard disk driver entry point


                           MFDOS result codes


0001  ==       @MFSNF    ==     1       ; sync not found
0002  ==       @MFCRC    ==     2       ; CRC error
0003  ==       @MFVFY    ==     3       ; verify compare error
0004  ==       @MFNIP    ==     4       ; no index pulse found
0005  ==       @MFDMM    ==     5       ; density mismatch on read or verify
0006  ==       @MFWRP    ==     6       ; attempt to write on protected diskette
0007  ==       @MFCCX    ==     7       ; control C detected from console
0008  ==       @MFIDW    ==     8       ; illegal call to DWRIT
0009  ==       @MFIDN    ==     9       ; illegal drive number
000A  ==       @MFIDA    ==     10      ; illegal disk address
000B  ==       @MFITL    ==     11      ; illegal transfer length
000C  ==       @MFIDC    ==     12      ; illegal command to DCOM
000D  ==       @MFTDM    ==     13      ; track density mismatch error
```



File Manager Definitions

File Manager Operation Codes

```
0000  ==        @FMCRE   ==     00H          ; create file
0001  ==        @FMDEL   ==     FMCRE+1      ; delete file
0002  ==        @FMOPN   ==     FMDEL+1      ; open file
0003  ==        @FMCLO   ==     FMOPN+1      ; close file
0004  ==        @FMTFR   ==     FMCLO+1      ; transfer sectors to/from file
0005  ==        @FMLOK   ==     FMTFR+1      ; lookup file information
0006  ==        @FMLAC   ==     FMLOK+1      ; lookup account information
0007  ==        @FMCTY   ==     FMLAC+1      ; change file type
0008  ==        @FMCAT   ==     FMCTY+1      ; change file attributes
0009  ==        @FMCFS   ==     FMCAT+1      ; change file size
000A  ==        @FMCAP   ==     FMCFS+1      ; change the auto-load-and-go pathname
000B  ==        @FMABT   ==     FMCAP+1      ; abort the calling program
000C  ==        @FMRST   ==     FMABT+1      ; reset the File Manager
000D  ==        @FMLX    ==     FMRST+1      ; load and execute a file of type FTMI
000E  ==        @FMFSZ   ==     FMLX+1       ; return a file's size to the caller
000F  ==        @FMCA    ==     FMFSZ+1      ; create account
0010  ==        @FMDA    ==     FMCA+1       ; delete account
0011  ==        @FMSDV   ==     FMDA+1       ; set/return default values
0012  ==        @FMCM    ==     FMSDV+1      ; compose a standard message code message
0013  ==        @FMFIN   ==     FMCM+1       ; finish executing the calling program
0014  ==        @FMRDS   ==     FMFIN+1      ; return disk statistics to the calling prog
0015  ==        @FMADE   ==     FMRDS+1      ; access directory entry
0016  ==        @FMRDV   ==     FMADE+1      ; return default values to the calling prog
0017  ==        @FMCAN   ==     FMRDV+1      ; change account name
0018  ==        @FMPD    ==     FMCAN+1      ; power down a specific unit
0019  ==        @FMPU    ==     FMPD+1       ; power up a specific unit
001A  ==        @FMFBU   ==     FMPU+1       ; flag buffer used
001B  ==        @FMINI   ==     FMFBU+1      ; initialize after bootstrap
001B  ==        @FMEOL   ==     FMINI        ; end of FM list
```

Message Codes

```
0000  ==        @MOK     ==     00H          ; ok, must be zero!
0014  ==        @MANE    ==     20           ; account not empty
0015  ==        @MDDUP   ==     MANE+1       ; directory: attempt to add duplicate symbol
0016  ==        @MDFUL   ==     MDDUP+1      ; directory: directory is full
0017  ==        @MDFND   ==     MDFUL+1      ; directory: matching symbol found
0018  ==        @MDBAD   ==     MDFND+1      ; directory: bad directory structure
0019  ==        @MDFNF   ==     MDBAD+1      ; directory: file name not found
001A  ==        @MDANF   ==     MDFNF+1      ; directory: account name not found
001B  ==        @MCADS   ==     MDANF+1      ; cant allocate disk space, disk possibly full
001C  ==        @MOFUL   ==     MCADS+1      ; cant open file, Open File Table full
001D  ==        @MOAVL   ==     MOFUL+1      ; OFT entry available
001E  ==        @MILDN   ==     MOAVL+1      ; illegal decimal number
001F  ==        @MILFN   ==     MILDN+1      ; illegal file name
0020  ==        @MILAN   ==     MILFN+1      ; illegal account name
0021  ==        @MILUN   ==     MILAN+1      ; illegal unit number
0022  ==        @MILID   ==     MILUN+1      ; illegal account ID
0023  ==        @MWRP    ==     MILID+1      ; write protect
0024  ==        @MDEP    ==     MWRP+1       ; delete protect
0025  ==        @MADEP   ==     MDEP+1       ; account delete protected
0026  ==        @MCCPF   ==     MADEP+1      ; can't change protected field
0027  ==        @MPARA   ==     MCCPF+1      ; pramater in error or out of range
0028  ==        @MFRT    ==     MPARA+1      ; file of wrong type specified
0029  ==        @MFNO    ==     MFRT+1       ; file not open
002A  ==        @MFOPN   ==     MFNO+1       ; file is open
002B  ==        @MFAIL   ==     MFOPN+1      ; general failur
002C  ==        @MEOLA   ==     MFAIL+1      ; end of list with available space
002D  ==        @MEOL    ==     MEOLA+1      ; end of list with no available space
002E  ==        @MIFMR   ==     MEOL+1       ; illegal File Manager request
002F  ==        @MFZNZ   ==     MIFMR+1      ; file size not zero
0030  ==        @MIFZ    ==     MFZNZ+1      ; illegal file size
0031  ==        @MEOF    ==     MIFZ+1       ; end of file reached during transfer
0032  ==        @MPEOF   ==     MEOF+1       ; transfer attempted past EOF
0033  ==        @MMEMP   ==     MPEOF+1      ; memory protect violation
0034  ==        @MUNPU   ==     MMEMP+1      ; unit not powered up
0035  ==        @MNYI    ==     MUNPU+1      ; function not yet implimented
0036  ==        @MFMO    ==     MNYI+1       ; file is multiply opened
0037  ==        @MDLRE   ==     MFMO+1       ; Disk level revision error
0038  ==        @MDNSL   ==     MDLRE+1      ; Drive label mismatch error
0039  ==        @MDNSS   ==     MDNSL+1      ; Drive size mismatch error
007F  ==        @MDNRDY  ==     127          ; drive not ready-out of speed
007E  ==        @MDNACC  ==     126          ; drive not ready for command
007D  ==        @MCNPR   ==     125          ; controller not there
```

```
007C  ==        @MDWNR   ==    124        ; drive went not ready after command started
007B  ==        @MCNFTS  ==    123        ; can not find target sector
007A  ==        @MDRDFL  ==    122        ; failure in drive read electronics
0079  ==        @MTSHDR  ==    121        ; targ sec has crc error in header
0078  ==        @MBADRV  ==    120        ; drive number too big
0077  ==        @MDERDS  ==    119        ; drive error during seek
0076  ==        @MHSLER  ==    118        ; head select error
0075  ==        @MCYLER  ==    117        ; drive on wrong cylinder
0074  ==        @MHDCRC  ==    116        ; crc error in header during position verify
0073  ==        @MILLDA  ==    115        ; illegal disk address used
0072  ==        @MCNTFL  ==    114        ; read/write ff will not set in controller

                   Message Codes (continued)

0071  ==        @MDWRTE  ==    113        ; write unsafe or attempt to wrt on prot cyl
0070  ==        @MFWSOR  ==    112        ; found wrong sector during read or verify
006F  ==        @MRDSHE  ==    111        ; crc or PLL sync error on read sector header
006E  ==        @MDCRCE  ==    110        ; data crc error
006D  ==        @MVDATE  ==    109        ; compare error in data during verify
006C  ==        @MVCRCE  ==    108        ; crc error on data during verify
006B  ==        @MRCER   ==    107        ; drive error during command execution
006A  ==        @MRDFL   ==    106        ; failure to be able to format drive
0069  ==        @MSHDR   ==    105        ; Pll sync error on read
0068  ==        @MMIPLS  ==    104        ; Missing index pulse
0067  ==        @MBUFRD  ==    103        ; Buffer error

                   Unit Structure

0080  ==        @SPD     ==    128    ; sectors per directory
0004  ==        @SPH     ==    4      ; sectors per DIB as a power of 2
                   HPM   ==    153*4  ; DIBs per unit
0000  ==        @DLDA    ==    0      ; disk label disk address
0001  ==        @HTDA    ==    1      ; DIB table disk address
0002  ==        @BHTDA   ==    2      ; bad DIB table disk address
0200  ==        @BPS     ==    512    ; bytes per sector

                   File Attributes

0001  ==        @ATBAK   ==    1      ; backupable flag, bit 0, backupable when 0
0080  ==        @ATDRT   ==    128    ; file dirty flag, bit 7, dirty when 1
0002  ==        @ATWRT   ==    2      ; write protect flag, bit 1, protected when 1
0004  ==        @ATDEL   ==    4      ; deleteable file flag, bit 2, not deleteable when 1

                   Directory Entry Type Codes

0000  ==        @ENTNU   ==    0      ; never used entry
0001  ==        @ENTIU   ==    1      ; entry in use (account or file)
0002  ==        @ENTDE   ==    2      ; deleted entry

                   File Types

0001  ==        @FTMI    ==    1      ; memory image file
0002  ==        @FTBP    ==    2      ; BASIC program file
0003  ==        @FTBD    ==    3      ; BASIC data file
0004  ==        @FTRD    ==    4      ; BACKUP/RECOVERY compressed directory
0005  ==        @FTRP    ==    5      ; BACKUP/RECOVERY packets
0006  ==        @FTCC    ==    6      ; CP/M work file
0007  ==        @FTCU    ==    7      ; CP/M unit
000A  ==        @FTPC    ==    10     ; PASCAL connection table
000B  ==        @FTPV    ==    11     ; PASCAL volume
000E  ==        @FTGB    ==    14     ; bar chart data
000F  ==        @FTGP    ==    15     ; pie chart data
0012  ==        @FTAS    ==    18     ; ASP sequential file
0013  ==        @FTAR    ==    19     ; ASP random file
0014  ==        @FTAI    ==    20     ; ASP index file

                   FMLX Option Codes

0000  ==        @LXRLE   ==    0      ; return load errors, execute loaded code
0001  ==        @LXALE   ==    1      ; use FMABT for load errors, execute code
0002  ==        @LXRET   ==    2      ; return on load errors, donot execute
0080  ==        @LXXPO   ==    080H   ; extended path option bit

                   FMTFR Operation Codes

0001  ==        @TCORD   ==    1      ; read
0000  ==        @TCOWR   ==    0      ; write
0002  ==        @TCOVF   ==    2      ; verify
0003  ==        @TCOWC   ==    3      ; write 'clean'
```

Disk Label Structure

```
0000  ==        @DLILL    ==    0               ; illegal directory address
0002  ==        @DLALX    ==    DLILL+2         ; auto-load-and-execute path name
0023  ==        @DLASZ    ==    35              ; auto load&execute pathname length
0025  ==        @DLMAJOR  ==    DLALX+DLASZ     ; major disk structure revision level
0026  ==        @DLMINOR  ==    DLMAJOR+1       ; minor disk structure revision level
0027  ==        @DLDSZE   ==    DLMINOR+1       ; disk size
0029  ==        @DLNSRT   ==    DLDSZE+2        ; number of sectors reserved for testing
002B  ==        @DLNHSZ   ==    DLNSRT+2        ; DIB size as a power of two
002C  ==        @DLDRSZ   ==    DLNHSZ+1        ; directory size (sectors)
002E  ==        @DLDIR    ==    DLDRSZ+2        ; base disk address for the unit's directory
0030  ==        @DLSST    ==    DLDIR+2         ; stepping delay in units of 12.5 microseconds
0031  ==        @DLMXH    ==    DLSST+1         ; maximum head number
0032  ==        @DLMXC    ==    DLMXH+1         ; maximum cylinder number
0034  ==        @DLPRC    ==    DLMXC+2         ; minimum cylinder with precomp on write
0036  ==        @DLLCC    ==    DLPRC+2         ; minimum cylinder with low current on write
0038  ==        @DLOFC    ==    DLLCC+2         ; cylinder to seek when sequencing down
003A  ==        @DLXXX    ==    DLOFC+2         ; reserved space
01C6  ==        @DLSIZ    ==    BPS-DLXXX       ; structure size
```

DIB Table Structure

```
0000  ==        @HTALC    ==    0               ; the count of allocated DIBs
0002  ==        @HTMAP    ==    HTALC+2         ; the bit map of DIBs, a 1 bit means inuse
```

Directory Entry Structure

```
0000  ==        @DETYP    ==    0               ; entry type
0001  ==        @DESYM    ==    DETYP+1         ; The entry's symbol
0001  ==        @DEACN    ==    DESYM           ; the account number
0003  ==        @DENAM    ==    DEACN+2         ; the name protion of the symbol
000E  ==        @DENMZ    ==    14              ; name length in symbol
0010  ==        @DESYZ    ==    DENAM+DENMZ-DESYM       ; symbol length
0011  ==        @DEBAS    ==    DESYM+DESYZ     ; base for account and file data
0020  ==        @DESIZ    ==    32              ; the size of an entry
0005  ==        @DESZ2    ==    5               ; the size of an entry as a power of 2
0004  ==        @DEPS     ==    4               ; directory entries per sector (2^DEPS)
0800  ==        @EPD      ==    2048            ; entries per directory
0007  ==        @DIRM     ==    07H             ; entry number upper mask
000F  ==        @DEOSM    ==    0FH             ; entry-of-sector lower mask
```

File Structure Discriptor

```
0000  ==        @FSDAT    ==    0               ; attributes
0001  ==        @FSDFZ    ==    FSDAT+1         ; file size in sectors
0003  ==        @FSDLB    ==    FSDFZ+2         ; bytes in use in last sector
0005  ==        @FSDHS    ==    FSDLB+2         ; nDIB size as a power of 2
0006  ==        @FSDMU    ==    FSDHS+1         ; unit number
0007  ==        @FSDH0    ==    FSDMU+1         ; DIB 0 address divided by SPH
0009  ==        @FSDSZ    ==    FSDH0+2         ; structure descriptor size
```

File Directory Entry

```
0011  ==        @FDESD    ==    DEBAS           ; structure descriptor
0011  ==        @FDEAT    ==    FDESD+FSDAT     ; attributes
0012  ==        @FDEFZ    ==    FDESD+FSDFZ     ; file size in sectors
0014  ==        @FDELB    ==    FDESD+FSDLB     ; bytes in use in the last sector
0016  ==        @FDEHS    ==    FDESD+FSDHS     ; sectors per nDIB as a power of 2
0017  ==        @FDEX1    ==    FDESD+FSDMU     ; reserved space
0018  ==        @FDEH0    ==    FDESD+FSDH0     ; DIB 0 disk addr divided by SPH
001A  ==        @FDEFT    ==    FDEH0+2         ; file type
001B  ==        @FDETD    ==    FDEFT+1         ; type dependant data
001E  ==        @FDEXX    ==    FDETD+3         ; reserved space, 2 bytes
```

Account Directory Entry

```
0011  ==        @ADEAN    ==    DESYM+DESYZ     ; account number
0013  ==        @ADEXX    ==    ADEAN+2         ; reserved space, 13 bytes
```

Index block

```
0100  ==        @IXSIZ    ==    BPS/2           ; one half sector long
0081  ==        @EPI      ==    129             ; entries per index
```

Open FIle Block (OFB)

```
0000  ==        @OFBDE   ==    0              ; directory entry number
0002  ==        @OFBDB   ==    OFBDE+2        ; base disk address for directory
0004  ==        @OFBIL   ==    OFBDB+2        ; index length (1..EPI)
0005  ==        @OFBSD   ==    OFBIL+1        ; the file's structure descriptor
0005  ==        @OFBAT   ==    OFBSD+FSDAT    ; attributes
0006  ==        @OFBFZ   ==    OFBSD+FSDFZ    ; file size in sectors
0008  ==        @OFBLB   ==    OFBSD+FSDLB    ; bytes in last sector
000A  ==        @OFBHS   ==    OFBSD+FSDHS    ; nDIB size
000B  ==        @OFBMU   ==    OFBSD+FSDMU    ; unit number
000C  ==        @OFBH0   ==    OFBSD+FSDH0    ; DIB 0 address divided by SPH
000E  ==        @OFBI    ==    OFBSD+FSDSZ    ; index block
010E  ==        @OFBCB   ==    OFBI+IXSIZ     ; OFB check byte
010F  ==        @OFBSZ   ==    OFBCB+1        ; OFB size
000F  ==        @OFBUM   ==    0FH            ; unit mask in OFBMU
```

Create Information Block

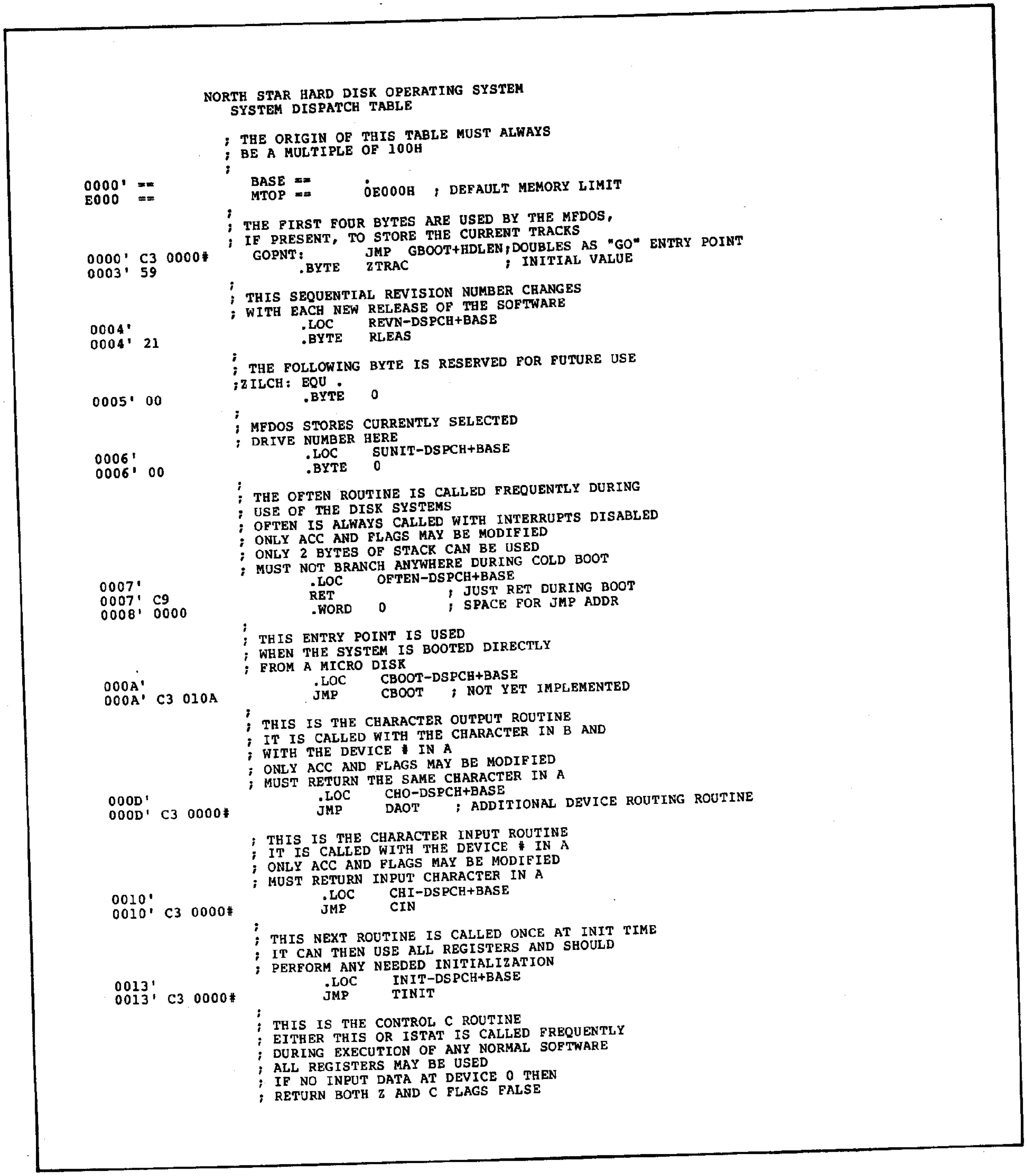
```
0000  ==        @CBTYP   ==    0              ; file type
0001  ==        @CBATR   ==    CBTYP+1        ; attributes
0002  ==        @CBNHZ   ==    CBATR+1        ; nDIB size as a power of 2
0003  ==        @CBPNA   ==    CBNHZ+1        ; pathname address
0005  ==        @CBTDD   ==    CBPNA+2        ; type dependant data
0008  ==        @CBSIZ   ==    CBTDD+3        ; create block size
```

Transfer Command Block

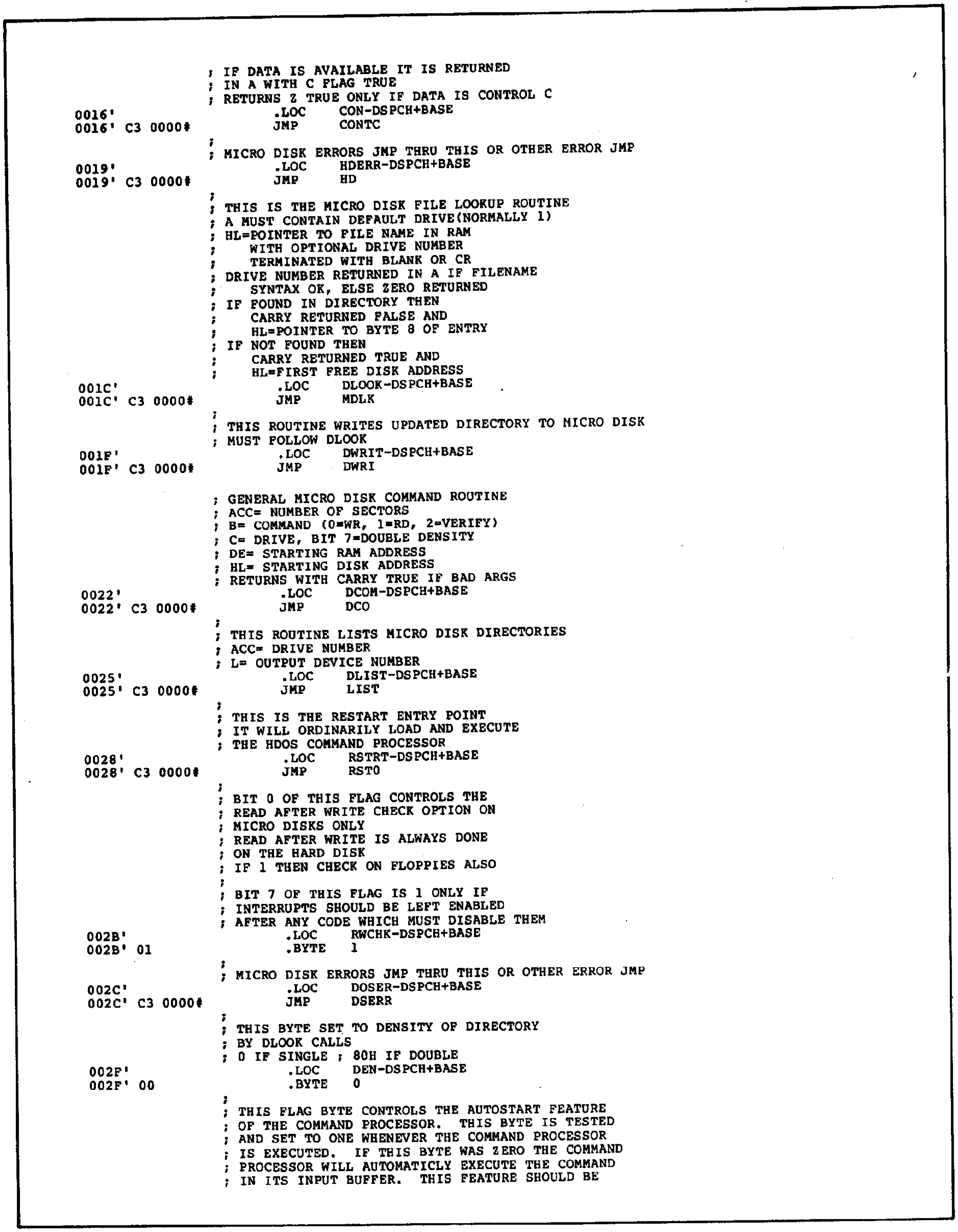
```
0000  ==        @TCOP    ==    0              ; operation code
0001  ==        @TCMEM   ==    TCOP+1         ; begining memory address
0003  ==        @TCSEC   ==    TCMEM+2        ; begining sector number
0005  ==        @TCLEN   ==    TCSEC+2        ; transfer length in sectors
0006  ==        @TCEMA   ==    TCLEN+1        ; ending memory address
0008  ==        @TCESN   ==    TCEMA+2        ; ending sector number
000A  ==        @TCSNT   ==    TCESN+2        ; sectors not transfered
000B  ==        @TCBLB   ==    TCSNT+1        ; bytes in use in the last sector
000D  ==        @TCDRT   ==    TCBLB+2        ; the 'or' of all dirty bits seen
000E  ==        @TCSIZ   ==    TCDRT+1        ; TC size
```

APPENDIX F

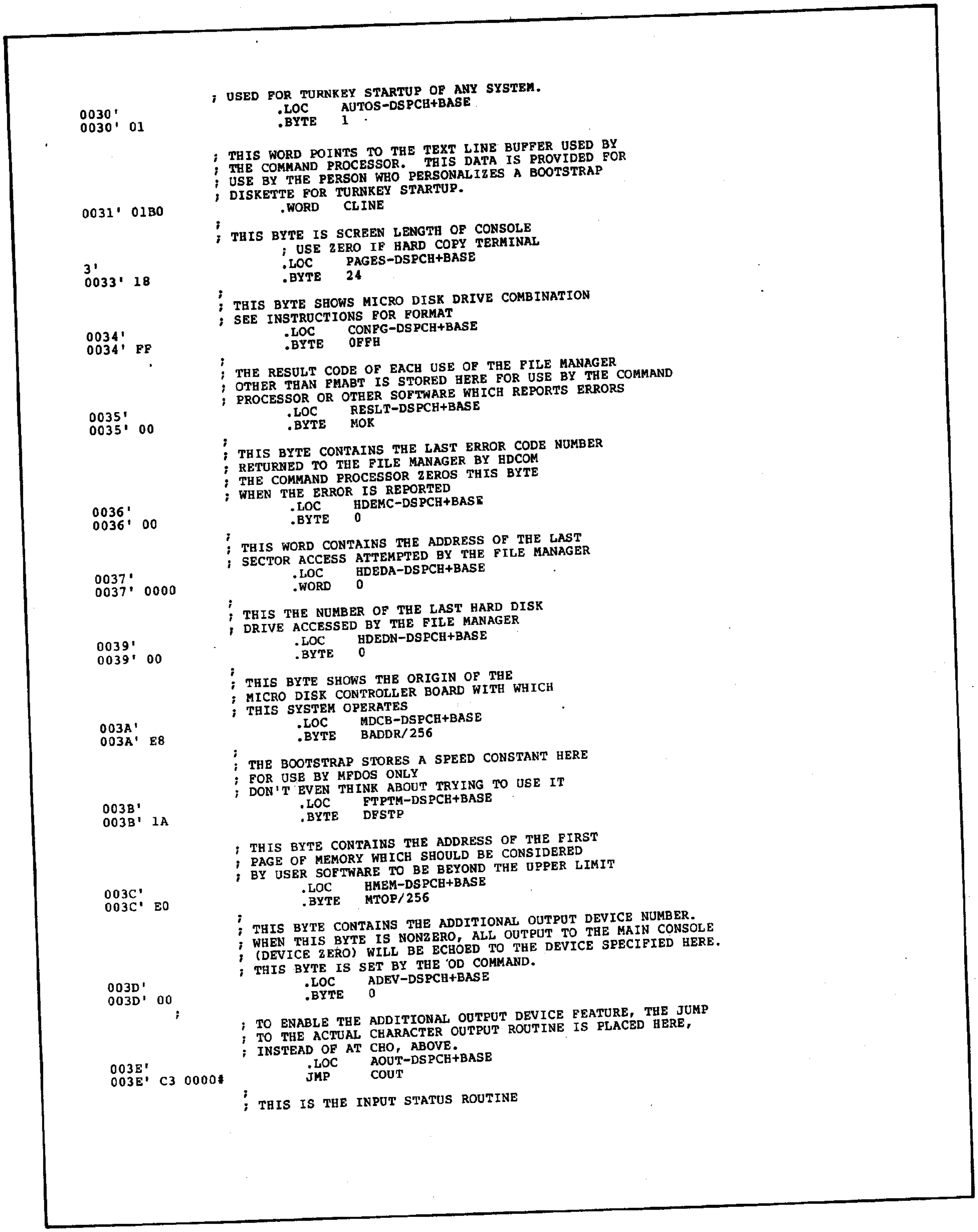
HDOS ENTRY POINTS AND I/O ROUTINES



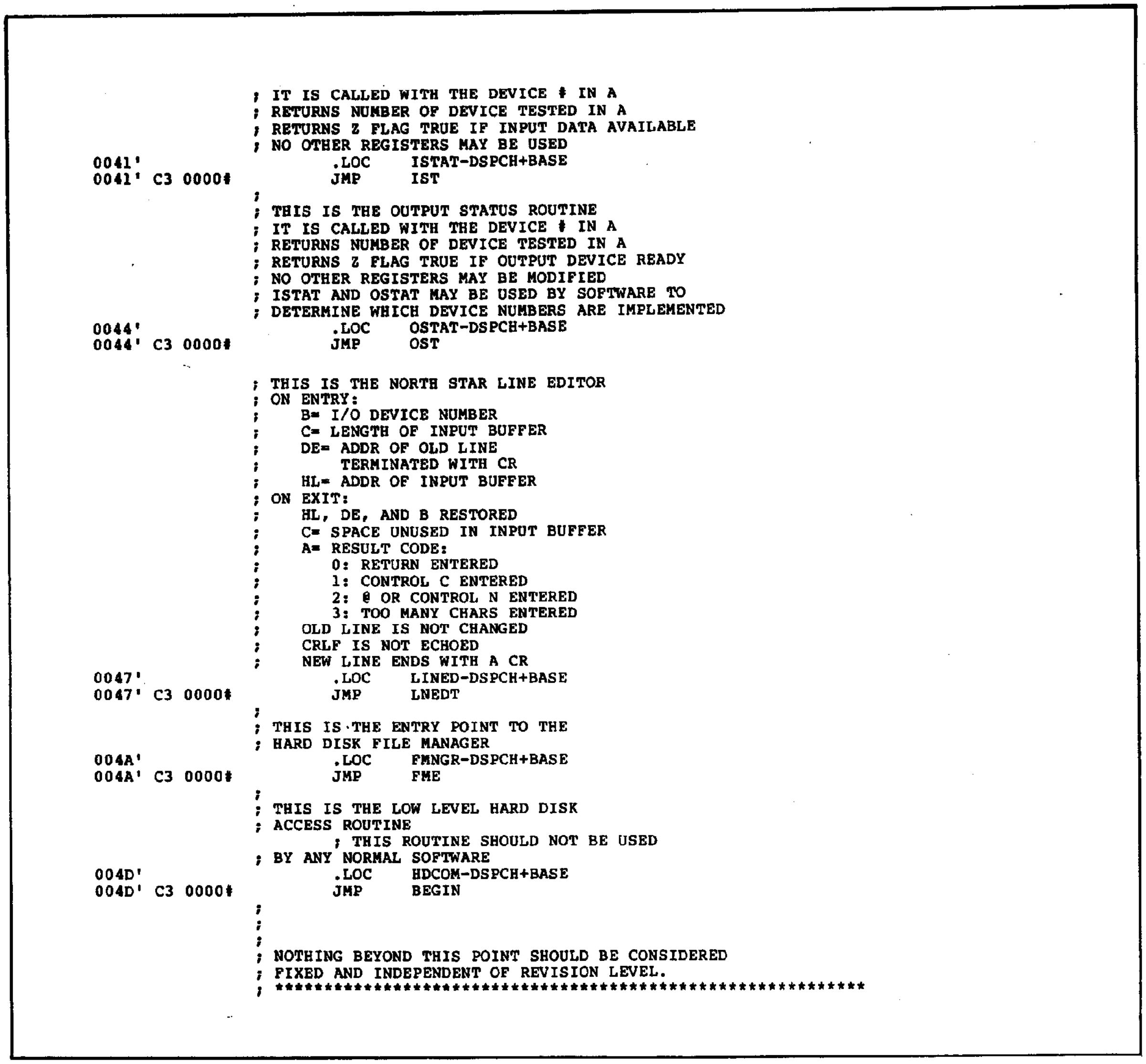
```
                    NORTH STAR HARD DISK OPERATING SYSTEM
                       SYSTEM DISPATCH TABLE

                      ; THE ORIGIN OF THIS TABLE MUST ALWAYS
                      ; BE A MULTIPLE OF 100H
                      ;
0000' ==                 BASE ==        .
E000  ==                 MTOP ==        0E000H  ; DEFAULT MEMORY LIMIT
                      ;
                      ; THE FIRST FOUR BYTES ARE USED BY THE MFDOS,
                      ; IF PRESENT, TO STORE THE CURRENT TRACKS
0000' C3 0000#           GOPNT:         JMP  GBOOT+HDLEN;DOUBLES AS "GO" ENTRY POINT
0003' 59                       .BYTE    ZTRAC              ; INITIAL VALUE
                      ;
                      ; THIS SEQUENTIAL REVISION NUMBER CHANGES
                      ; WITH EACH NEW RELEASE OF THE SOFTWARE
0004'                          .LOC     REVN-DSPCH+BASE
0004' 21                       .BYTE    RLEAS
                      ;
                      ; THE FOLLOWING BYTE IS RESERVED FOR FUTURE USE
                      ;ZILCH: EQU .
0005' 00                       .BYTE    0
                      ;
                      ; MFDOS STORES CURRENTLY SELECTED
                      ; DRIVE NUMBER HERE
0006'                          .LOC     SUNIT-DSPCH+BASE
0006' 00                       .BYTE    0
                      ;
                      ; THE OFTEN ROUTINE IS CALLED FREQUENTLY DURING
                      ; USE OF THE DISK SYSTEMS
                      ; OFTEN IS ALWAYS CALLED WITH INTERRUPTS DISABLED
                      ; ONLY ACC AND FLAGS MAY BE MODIFIED
                      ; ONLY 2 BYTES OF STACK CAN BE USED
                      ; MUST NOT BRANCH ANYWHERE DURING COLD BOOT
0007'                          .LOC     OFTEN-DSPCH+BASE
0007' C9                       RET               ; JUST RET DURING BOOT
0008' 0000                     .WORD    0        ; SPACE FOR JMP ADDR
                      ;
                      ; THIS ENTRY POINT IS USED
                      ; WHEN THE SYSTEM IS BOOTED DIRECTLY
                      ; FROM A MICRO DISK
000A'                          .LOC     CBOOT-DSPCH+BASE
000A' C3 010A                  JMP      CBOOT    ; NOT YET IMPLEMENTED
                      ;
                      ; THIS IS THE CHARACTER OUTPUT ROUTINE
                      ; IT IS CALLED WITH THE CHARACTER IN B AND
                      ; WITH THE DEVICE # IN A
                      ; ONLY ACC AND FLAGS MAY BE MODIFIED
                      ; MUST RETURN THE SAME CHARACTER IN A
000D'                          .LOC     CHO-DSPCH+BASE
000D' C3 0000#                 JMP      DAOT     ; ADDITIONAL DEVICE ROUTING ROUTINE

                      ; THIS IS THE CHARACTER INPUT ROUTINE
                      ; IT IS CALLED WITH THE DEVICE # IN A
                      ; ONLY ACC AND FLAGS MAY BE MODIFIED
                      ; MUST RETURN INPUT CHARACTER IN A
0010'                          .LOC     CHI-DSPCH+BASE
0010' C3 0000#                 JMP      CIN
                      ;
                      ; THIS NEXT ROUTINE IS CALLED ONCE AT INIT TIME
                      ; IT CAN THEN USE ALL REGISTERS AND SHOULD
                      ; PERFORM ANY NEEDED INITIALIZATION
0013'                          .LOC     INIT-DSPCH+BASE
0013' C3 0000#                 JMP      TINIT
                      ;
                      ; THIS IS THE CONTROL C ROUTINE
                      ; EITHER THIS OR ISTAT IS CALLED FREQUENTLY
                      ; DURING EXECUTION OF ANY NORMAL SOFTWARE
                      ; ALL REGISTERS MAY BE USED
                      ; IF NO INPUT DATA AT DEVICE 0 THEN
                      ; RETURN BOTH Z AND C FLAGS FALSE
```



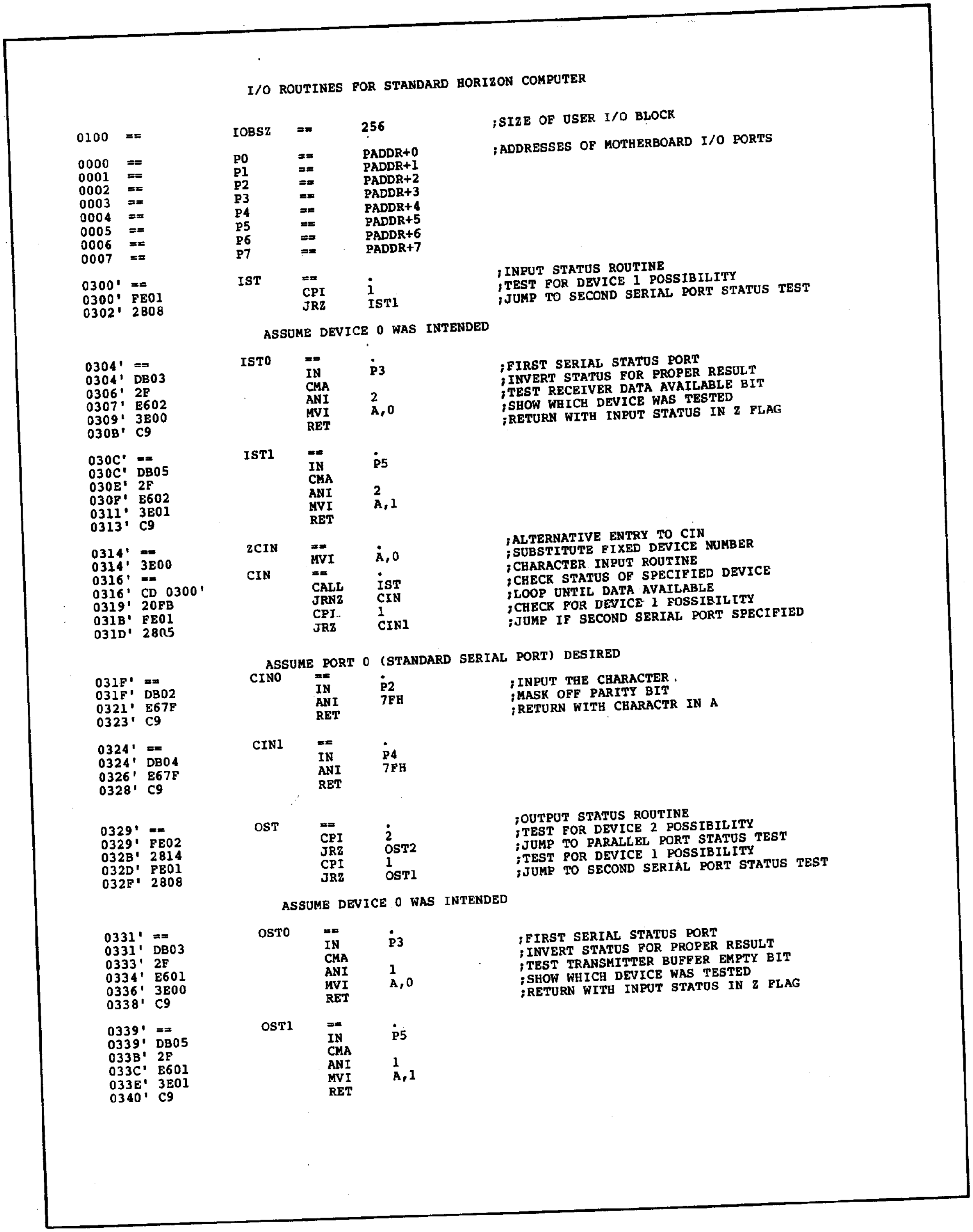
```
                        ; IF DATA IS AVAILABLE IT IS RETURNED
                        ; IN A WITH C FLAG TRUE
                        ; RETURNS Z TRUE ONLY IF DATA IS CONTROL C
0016'                           .LOC    CON-DSPCH+BASE
0016' C3 0000#                  JMP     CONTC
                        ;
                        ; MICRO DISK ERRORS JMP THRU THIS OR OTHER ERROR JMP
0019'                           .LOC    HDERR-DSPCH+BASE
0019' C3 0000#                  JMP     HD
                        ;
                        ; THIS IS THE MICRO DISK FILE LOOKUP ROUTINE
                        ; A MUST CONTAIN DEFAULT DRIVE(NORMALLY 1)
                        ; HL=POINTER TO FILE NAME IN RAM
                        ;    WITH OPTIONAL DRIVE NUMBER
                        ;    TERMINATED WITH BLANK OR CR
                        ; DRIVE NUMBER RETURNED IN A IF FILENAME
                        ;    SYNTAX OK, ELSE ZERO RETURNED
                        ; IF FOUND IN DIRECTORY THEN
                        ;    CARRY RETURNED FALSE AND
                        ;    HL=POINTER TO BYTE 8 OF ENTRY
                        ; IF NOT FOUND THEN
                        ;    CARRY RETURNED TRUE AND
                        ;    HL=FIRST FREE DISK ADDRESS
001C'                           .LOC    DLOOK-DSPCH+BASE
001C' C3 0000#                  JMP     MDLK
                        ;
                        ; THIS ROUTINE WRITES UPDATED DIRECTORY TO MICRO DISK
                        ; MUST FOLLOW DLOOK
001F'                           .LOC    DWRIT-DSPCH+BASE
001F' C3 0000#                  JMP     DWRI

                        ; GENERAL MICRO DISK COMMAND ROUTINE
                        ; ACC= NUMBER OF SECTORS
                        ; B= COMMAND (0=WR, 1=RD, 2=VERIFY)
                        ; C= DRIVE, BIT 7=DOUBLE DENSITY
                        ; DE= STARTING RAM ADDRESS
                        ; HL= STARTING DISK ADDRESS
                        ; RETURNS WITH CARRY TRUE IF BAD ARGS
0022'                           .LOC    DCOM-DSPCH+BASE
0022' C3 0000#                  JMP     DCO
                        ;
                        ; THIS ROUTINE LISTS MICRO DISK DIRECTORIES
                        ; ACC= DRIVE NUMBER
                        ; L= OUTPUT DEVICE NUMBER
0025'                           .LOC    DLIST-DSPCH+BASE
0025' C3 0000#                  JMP     LIST
                        ;
                        ; THIS IS THE RESTART ENTRY POINT
                        ; IT WILL ORDINARILY LOAD AND EXECUTE
                        ; THE HDOS COMMAND PROCESSOR
0028'                           .LOC    RSTRT-DSPCH+BASE
0028' C3 0000#                  JMP     RSTO
                        ;
                        ; BIT 0 OF THIS FLAG CONTROLS THE
                        ; READ AFTER WRITE CHECK OPTION ON
                        ; MICRO DISKS ONLY
                        ; READ AFTER WRITE IS ALWAYS DONE
                        ; ON THE HARD DISK
                        ; IF 1 THEN CHECK ON FLOPPIES ALSO
                        ;
                        ; BIT 7 OF THIS FLAG IS 1 ONLY IF
                        ; INTERRUPTS SHOULD BE LEFT ENABLED
                        ; AFTER ANY CODE WHICH MUST DISABLE THEM
002B'                           .LOC    RWCHK-DSPCH+BASE
002B' 01                        .BYTE   1
                        ;
                        ; MICRO DISK ERRORS JMP THRU THIS OR OTHER ERROR JMP
002C'                           .LOC    DOSER-DSPCH+BASE
002C' C3 0000#                  JMP     DSERR
                        ;
                        ; THIS BYTE SET TO DENSITY OF DIRECTORY
                        ; BY DLOOK CALLS
                        ; 0 IF SINGLE ; 80H IF DOUBLE
002F'                           .LOC    DEN-DSPCH+BASE
002F' 00                        .BYTE   0
                        ;
                        ; THIS FLAG BYTE CONTROLS THE AUTOSTART FEATURE
                        ; OF THE COMMAND PROCESSOR.  THIS BYTE IS TESTED
                        ; AND SET TO ONE WHENEVER THE COMMAND PROCESSOR
                        ; IS EXECUTED.  IF THIS BYTE WAS ZERO THE COMMAND
                        ; PROCESSOR WILL AUTOMATICLY EXECUTE THE COMMAND
                        ; IN ITS INPUT BUFFER.  THIS FEATURE SHOULD BE
```



```
                        ; USED FOR TURNKEY STARTUP OF ANY SYSTEM.
0030'                           .LOC    AUTOS-DSPCH+BASE
0030' 01                        .BYTE   1

                        ; THIS WORD POINTS TO THE TEXT LINE BUFFER USED BY
                        ; THE COMMAND PROCESSOR.  THIS DATA IS PROVIDED FOR
                        ; USE BY THE PERSON WHO PERSONALIZES A BOOTSTRAP
                        ; DISKETTE FOR TURNKEY STARTUP.
0031' 01B0                      .WORD   CLINE
                        ;
                        ; THIS BYTE IS SCREEN LENGTH OF CONSOLE
                                ; USE ZERO IF HARD COPY TERMINAL
3'                              .LOC    PAGES-DSPCH+BASE
0033' 18                        .BYTE   24
                        ;
                        ; THIS BYTE SHOWS MICRO DISK DRIVE COMBINATION
                        ; SEE INSTRUCTIONS FOR FORMAT
0034'                           .LOC    CONFG-DSPCH+BASE
0034' FF                        .BYTE   0FFH
                        ;
                        ; THE RESULT CODE OF EACH USE OF THE FILE MANAGER
                        ; OTHER THAN FMABT IS STORED HERE FOR USE BY THE COMMAND
                        ; PROCESSOR OR OTHER SOFTWARE WHICH REPORTS ERRORS
0035'                           .LOC    RESLT-DSPCH+BASE
0035' 00                        .BYTE   MOK
                        ;
                        ; THIS BYTE CONTAINS THE LAST ERROR CODE NUMBER
                        ; RETURNED TO THE FILE MANAGER BY HDCOM
                        ; THE COMMAND PROCESSOR ZEROS THIS BYTE
                        ; WHEN THE ERROR IS REPORTED
0036'                           .LOC    HDEMC-DSPCH+BASE
0036' 00                        .BYTE   0
                        ;
                        ; THIS WORD CONTAINS THE ADDRESS OF THE LAST
                        ; SECTOR ACCESS ATTEMPTED BY THE FILE MANAGER
0037'                           .LOC    HDEDA-DSPCH+BASE
0037' 0000                      .WORD   0
                        ;
                        ; THIS THE NUMBER OF THE LAST HARD DISK
                        ; DRIVE ACCESSED BY THE FILE MANAGER
0039'                           .LOC    HDEDN-DSPCH+BASE
0039' 00                        .BYTE   0
                        ;
                        ; THIS BYTE SHOWS THE ORIGIN OF THE
                        ; MICRO DISK CONTROLLER BOARD WITH WHICH
                        ; THIS SYSTEM OPERATES
003A'                           .LOC    MDCB-DSPCH+BASE
003A' E8                        .BYTE   BADDR/256
                        ;
                        ; THE BOOTSTRAP STORES A SPEED CONSTANT HERE
                        ; FOR USE BY MFDOS ONLY
                        ; DON'T EVEN THINK ABOUT TRYING TO USE IT
003B'                           .LOC    FTPTM-DSPCH+BASE
003B' 1A                        .BYTE   DFSTP

                        ; THIS BYTE CONTAINS THE ADDRESS OF THE FIRST
                        ; PAGE OF MEMORY WHICH SHOULD BE CONSIDERED
                        ; BY USER SOFTWARE TO BE BEYOND THE UPPER LIMIT
003C'                           .LOC    HMEM-DSPCH+BASE
003C' E0                        .BYTE   MTOP/256
                        ;
                        ; THIS BYTE CONTAINS THE ADDITIONAL OUTPUT DEVICE NUMBER.
                        ; WHEN THIS BYTE IS NONZERO, ALL OUTPUT TO THE MAIN CONSOLE
                        ; (DEVICE ZERO) WILL BE ECHOED TO THE DEVICE SPECIFIED HERE.
                        ; THIS BYTE IS SET BY THE OD COMMAND.
003D'                           .LOC    ADEV-DSPCH+BASE
003D' 00                        .BYTE   0
                  ;
                        ; TO ENABLE THE ADDITIONAL OUTPUT DEVICE FEATURE, THE JUMP
                        ; TO THE ACTUAL CHARACTER OUTPUT ROUTINE IS PLACED HERE,
                        ; INSTEAD OF AT CHO, ABOVE.
003E'                           .LOC    AOUT-DSPCH+BASE
003E' C3 0000#                  JMP     COUT
                        ;
                        ; THIS IS THE INPUT STATUS ROUTINE
```

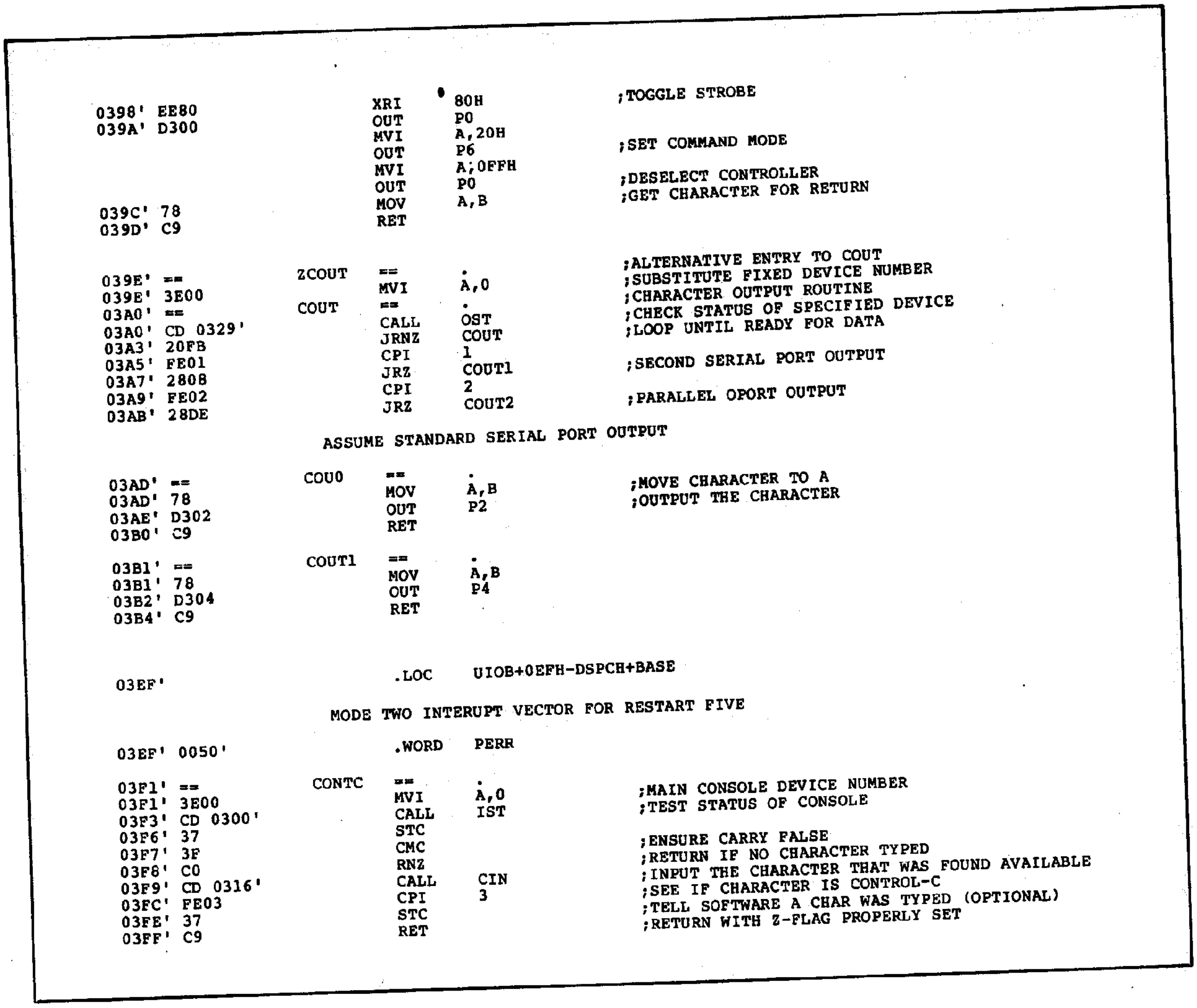


```
                  ; IT IS CALLED WITH THE DEVICE # IN A
                  ; RETURNS NUMBER OF DEVICE TESTED IN A
                  ; RETURNS Z FLAG TRUE IF INPUT DATA AVAILABLE
                  ; NO OTHER REGISTERS MAY BE USED
0041'                     .LOC    ISTAT-DSPCH+BASE
0041' C3 0000#            JMP     IST
                  ;
                  ; THIS IS THE OUTPUT STATUS ROUTINE
                  ; IT IS CALLED WITH THE DEVICE # IN A
                  ; RETURNS NUMBER OF DEVICE TESTED IN A
                  ; RETURNS Z FLAG TRUE IF OUTPUT DEVICE READY
                  ; NO OTHER REGISTERS MAY BE MODIFIED
                  ; ISTAT AND OSTAT MAY BE USED BY SOFTWARE TO
                  ; DETERMINE WHICH DEVICE NUMBERS ARE IMPLEMENTED
0044'                     .LOC    OSTAT-DSPCH+BASE
0044' C3 0000#            JMP     OST

                  ; THIS IS THE NORTH STAR LINE EDITOR
                  ; ON ENTRY:
                  ;    B= I/O DEVICE NUMBER
                  ;    C= LENGTH OF INPUT BUFFER
                  ;    DE= ADDR OF OLD LINE
                  ;        TERMINATED WITH CR
                  ;    HL= ADDR OF INPUT BUFFER
                  ; ON EXIT:
                  ;    HL, DE, AND B RESTORED
                  ;    C= SPACE UNUSED IN INPUT BUFFER
                  ;    A= RESULT CODE:
                  ;       0: RETURN ENTERED
                  ;       1: CONTROL C ENTERED
                  ;       2: @ OR CONTROL N ENTERED
                  ;       3: TOO MANY CHARS ENTERED
                  ;    OLD LINE IS NOT CHANGED
                  ;    CRLF IS NOT ECHOED
                  ;    NEW LINE ENDS WITH A CR
0047'                     .LOC    LINED-DSPCH+BASE
0047' C3 0000#            JMP     LNEDT
                  ;
                  ; THIS IS THE ENTRY POINT TO THE
                  ; HARD DISK FILE MANAGER
004A'                     .LOC    FMNGR-DSPCH+BASE
004A' C3 0000#            JMP     FME
                  ;
                  ; THIS IS THE LOW LEVEL HARD DISK
                  ; ACCESS ROUTINE
                          ; THIS ROUTINE SHOULD NOT BE USED
                  ; BY ANY NORMAL SOFTWARE
004D'                     .LOC    HDCOM-DSPCH+BASE
004D' C3 0000#            JMP     BEGIN
                  ;
                  ;
                  ;
                  ; NOTHING BEYOND THIS POINT SHOULD BE CONSIDERED
                  ; FIXED AND INDEPENDENT OF REVISION LEVEL.
                  ; **************************************************************
```

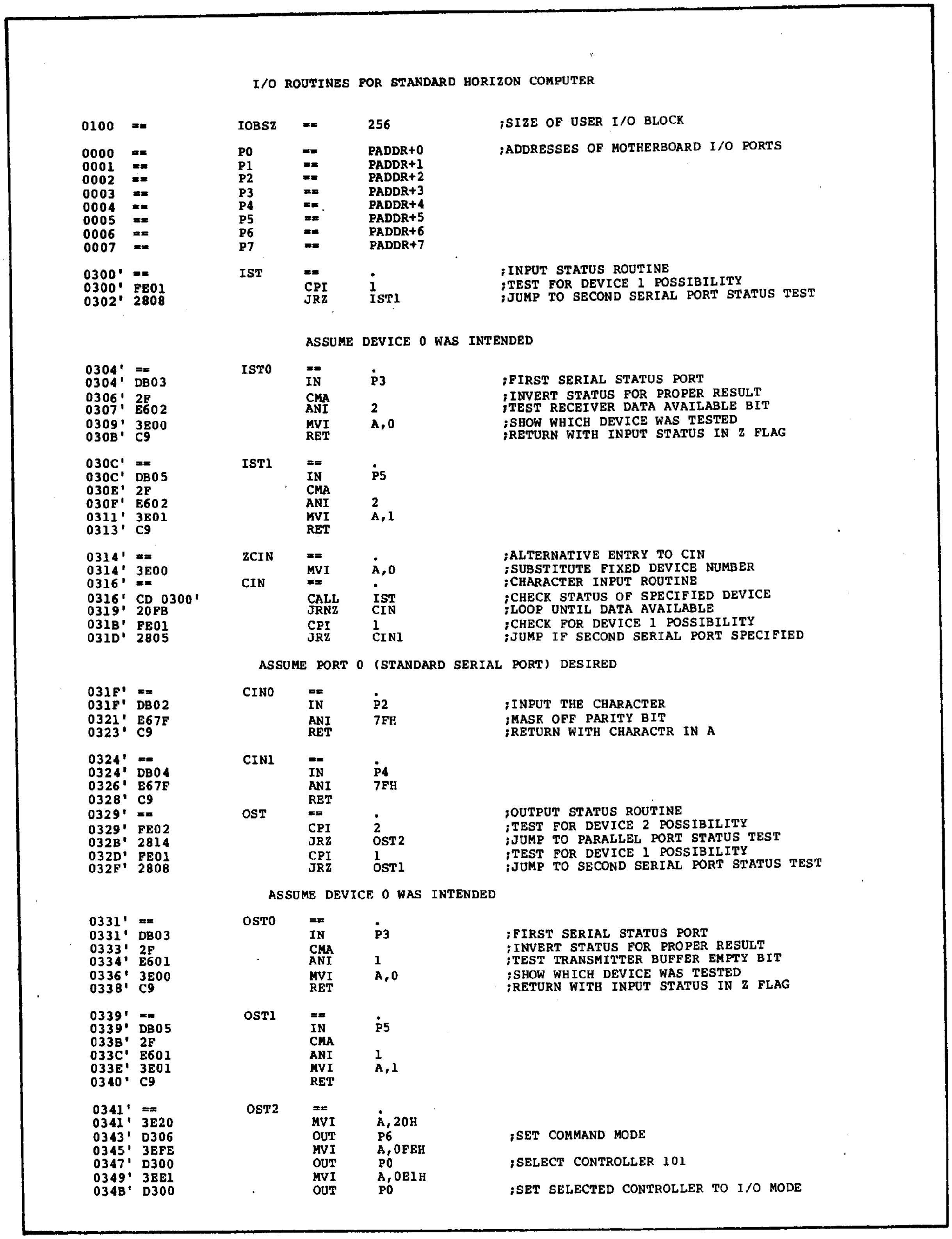


```
                    I/O ROUTINES FOR STANDARD HORIZON COMPUTER

0100   ==          IOBSZ   ==      256          ;SIZE OF USER I/O BLOCK

                                                ;ADDRESSES OF MOTHERBOARD I/O PORTS
0000   ==          P0      ==      PADDR+0
0001   ==          P1      ==      PADDR+1
0002   ==          P2      ==      PADDR+2
0003   ==          P3      ==      PADDR+3
0004   ==          P4      ==      PADDR+4
0005   ==          P5      ==      PADDR+5
0006   ==          P6      ==      PADDR+6
0007   ==          P7      ==      PADDR+7

                                                ;INPUT STATUS ROUTINE
0300' ==           IST     ==      .
0300' FE01                 CPI     1            ;TEST FOR DEVICE 1 POSSIBILITY
0302' 2808                 JRZ     IST1         ;JUMP TO SECOND SERIAL PORT STATUS TEST

                     ASSUME DEVICE 0 WAS INTENDED

0304' ==           IST0    ==      .            ;FIRST SERIAL STATUS PORT
0304' DB03                 IN      P3
0306' 2F                   CMA                  ;INVERT STATUS FOR PROPER RESULT
0307' E602                 ANI     2            ;TEST RECEIVER DATA AVAILABLE BIT
0309' 3E00                 MVI     A,0          ;SHOW WHICH DEVICE WAS TESTED
030B' C9                   RET                  ;RETURN WITH INPUT STATUS IN Z FLAG

030C' ==           IST1    ==      .
030C' DB05                 IN      P5
030E' 2F                   CMA
030F' E602                 ANI     2
0311' 3E01                 MVI     A,1
0313' C9                   RET

                                                ;ALTERNATIVE ENTRY TO CIN
0314' ==           ZCIN    ==      .
0314' 3E00                 MVI     A,0          ;SUBSTITUTE FIXED DEVICE NUMBER
0316' ==           CIN     ==      .            ;CHARACTER INPUT ROUTINE
0316' CD 0300'             CALL    IST          ;CHECK STATUS OF SPECIFIED DEVICE
0319' 20FB                 JRNZ    CIN          ;LOOP UNTIL DATA AVAILABLE
031B' FE01                 CPI     1            ;CHECK FOR DEVICE 1 POSSIBILITY
031D' 2805                 JRZ     CIN1         ;JUMP IF SECOND SERIAL PORT SPECIFIED

                     ASSUME PORT 0 (STANDARD SERIAL PORT) DESIRED
031F' ==           CIN0    ==      .
031F' DB02                 IN      P2           ;INPUT THE CHARACTER
0321' E67F                 ANI     7FH          ;MASK OFF PARITY BIT
0323' C9                   RET                  ;RETURN WITH CHARACTR IN A

0324' ==           CIN1    ==      .
0324' DB04                 IN      P4
0326' E67F                 ANI     7FH
0328' C9                   RET

                                                ;OUTPUT STATUS ROUTINE
0329' ==           OST     ==      .
0329' FE02                 CPI     2            ;TEST FOR DEVICE 2 POSSIBILITY
032B' 2814                 JRZ     OST2         ;JUMP TO PARALLEL PORT STATUS TEST
032D' FE01                 CPI     1            ;TEST FOR DEVICE 1 POSSIBILITY
032F' 2808                 JRZ     OST1         ;JUMP TO SECOND SERIAL PORT STATUS TEST

                     ASSUME DEVICE 0 WAS INTENDED

0331' ==           OST0    ==      .            ;FIRST SERIAL STATUS PORT
0331' DB03                 IN      P3
0333' 2F                   CMA                  ;INVERT STATUS FOR PROPER RESULT
0334' E601                 ANI     1            ;TEST TRANSMITTER BUFFER EMPTY BIT
0336' 3E00                 MVI     A,0          ;SHOW WHICH DEVICE WAS TESTED
0338' C9                   RET                  ;RETURN WITH INPUT STATUS IN Z FLAG

0339' ==           OST1    ==      .
0339' DB05                 IN      P5
033B' 2F                   CMA
033C' E601                 ANI     1
033E' 3E01                 MVI     A,1
0340' C9                   RET
```

```
0341' ==        OST2    ==      .
                        MVI     A,20H
                        OUT     P6              ;SET COMMAND MODE
                        MVI     A,0FEH
                        OUT     P0              ;SELECT CONTROLLER 101
                        MVI     A,0E1H
                        OUT     P0              ;SET SELECTED CONTROLLER TO I/O MODE
                        MVI     A,60H
                        OUT     P6              ;SET DATA MODE
0341' DB06              IN      P6              ;MOTHERBOARD STATUS BYTE
0343' 2F                CMA
                        BIT     3,A             ;TEST CONTROLLER'S PO FLAG
0344' CB47              BIT     0,A             ;*** TEST MOTHERBOARD'S PO FLAG ***
                        MVI     A,20H
                        OUT     P6              ;SET COMMAND MODE
                        MVI     A,0FFH
                        OUT     P0              ;DESELECT CONTROLLER
0346' 3E02              MVI     A,2             ;DEVICE NUMBER
0348' C9                RET

             TINIT FIRST REWRITES ALL RAM TO SET PARITY CORRECT

0349' ==        TINIT   ==      .
0349' 3E40              MVI     A,40H           ;DISABLE PARITY LOGIC
034B' D3C0              OUT     0C0H            ;BEFORE READING UNWRITTEN RAM
034D' 21 EC00           LXI     H,BADDR+1024    ;FIRST BYTE TO CLEAR
0350' 54                MOV     D,H
0351' 5D                MOV     E,L
0352' 01 FC00           LXI     B,-1024         ;NUMBER OF BYTES TO CLEAR
0355' EDB0              LDIR                    ;SET PARITY ON ALL RAM
0357' 3C                INR     A               ;TO 41H, PARITY ENABLE CODE
0358' D3C0              OUT     0C0H            ;REARM PARITY LOGIC

              NOW INITIALIZE MOTHERBOARD AND SET UP BOTH SERIAL PORTS

035A' AF                XRA     A               ;ZERO ACC
035B' D306              OUT     P6              ;INITIALIZE MOTHERBOARD
035D' D306              OUT     P6              ;EXTRA
035F' D306              OUT     P6              ;EXTRA
0361' D306              OUT     P6              ;EXTRA
0363' 3ECE              MVI     A,0CEH          ;2 STOPS, 16xCLOCK, 8 BITS, NO PARITY
0365' D303              OUT     P3              ;SEND TO FIRST SERIAL PORT
0367' 3ECE              MVI     A,0CEH          ;SAME CODE AS FIRST PORT
0369' D305              OUT     P5              ;SECOND PORT
036B' 3E37              MVI     A,37H           ;CMD: RTS, ER, RXF, DTR, TXEN
036D' D303              OUT     P3              ;FIRST PORT
036F' 3E37              MVI     A,37H           ;SAME CODE AS FIRST PORT
0371' D305              OUT     P5              ;SECOND PORT


0373' DB02              IN      P2              ;CLEAR STANDARD SERIAL PORT INPUT BUFFER
0375' DB02              IN      P2
0377' DB04              IN      P4              ;CLEAR SECOND SERIAL PORT INPUT BUFFER
0379' DB04              IN      P4


037B' 060D              MVI     B,13            ;CARRIAGE RETURN TO INIT PRINTER
037D' 21 012B           LXI     H,RWCHK
0380' CB7E              BIT     7,M             ;TEST INTERUPT ENABLE FLAG
0382' 2807              JRZ     COUT2
0384' 3E04              MVI     A,UIOB/256              ;PAGE ADDRESS OF I/O BLOCK
0386' ED47              STAI
0388' ED5E              IM2                     ;SET INTERUPT MODE TWO
038A' FB                EI

             PRINTER PARALLEL OUTPUT ROUTINE

038B' ==        COUT2   ==      .
038B' 3E20              MVI     A,20H
                        OUT     P6              ;SET COMMAND MODE
038D' D306              OUT     P6              ;*** CLEAR MOTHERBOARD'S PO FLAG ***
                        MVI     A,0FEH
                        OUT     P0              ;SELECT CONTROLLER 101 AGAIN
                        MVI     A,0E1H
                        OUT     P0              ;SET SELECTED CONTROLLER TO I/O MODE
                        MVI     A,60H
                        OUT     P6              ;SET DATA MODE
                        IN      P0              ;CLEAR CONTROLLER'S PO FLAG
038F' 78                MOV     A,B             ;CHARACTER TO SEND
0390' F680              ORI     80H             ;SET STROBE FALSE
0392' D300              OUT     P0              ;SEND CHARACTER
0394' EE80              XRI     80H             ;TOGGLE STROBE
0396' D300              OUT     P0
```



```
0398' EE80                XRI     80H            ;TOGGLE STROBE
039A' D300                OUT     P0
                          MVI     A,20H
                          OUT     P6             ;SET COMMAND MODE
                          MVI     A;0FFH
                          OUT     P0             ;DESELECT CONTROLLER
039C' 78                  MOV     A,B            ;GET CHARACTER FOR RETURN
039D' C9                  RET

039E' ==          ZCOUT   ==      .              ;ALTERNATIVE ENTRY TO COUT
039E' 3E00                MVI     A,0            ;SUBSTITUTE FIXED DEVICE NUMBER
03A0' ==          COUT    ==      .              ;CHARACTER OUTPUT ROUTINE
03A0' CD 0329'            CALL    OST            ;CHECK STATUS OF SPECIFIED DEVICE
03A3' 20FB                JRNZ    COUT           ;LOOP UNTIL READY FOR DATA
03A5' FE01                CPI     1
03A7' 2808                JRZ     COUT1          ;SECOND SERIAL PORT OUTPUT
03A9' FE02                CPI     2
03AB' 28DE                JRZ     COUT2          ;PARALLEL OPORT OUTPUT

                  ASSUME STANDARD SERIAL PORT OUTPUT

03AD' ==          COU0    ==      .
03AD' 78                  MOV     A,B            ;MOVE CHARACTER TO A
03AE' D302                OUT     P2             ;OUTPUT THE CHARACTER
03B0' C9                  RET

03B1' ==          COUT1   ==      .
03B1' 78                  MOV     A,B
03B2' D304                OUT     P4
03B4' C9                  RET


03EF'                     .LOC    UIOB+0EFH-DSPCH+BASE

                  MODE TWO INTERUPT VECTOR FOR RESTART FIVE

03EF' 0050'               .WORD   PERR

03F1' ==          CONTC   ==      .              ;MAIN CONSOLE DEVICE NUMBER
03F1' 3E00                MVI     A,0            ;TEST STATUS OF CONSOLE
03F3' CD 0300'            CALL    IST
03F6' 37                  STC                    ;ENSURE CARRY FALSE
03F7' 3F                  CMC                    ;RETURN IF NO CHARACTER TYPED
03F8' C0                  RNZ                    ;INPUT THE CHARACTER THAT WAS FOUND AVAILABLE
03F9' CD 0316'            CALL    CIN            ;SEE IF CHARACTER IS CONTROL-C
03FC' FE03                CPI     3              ;TELL SOFTWARE A CHAR WAS TYPED (OPTIONAL)
03FE' 37                  STC                    ;RETURN WITH Z-FLAG PROPERLY SET
03FF' C9                  RET
```



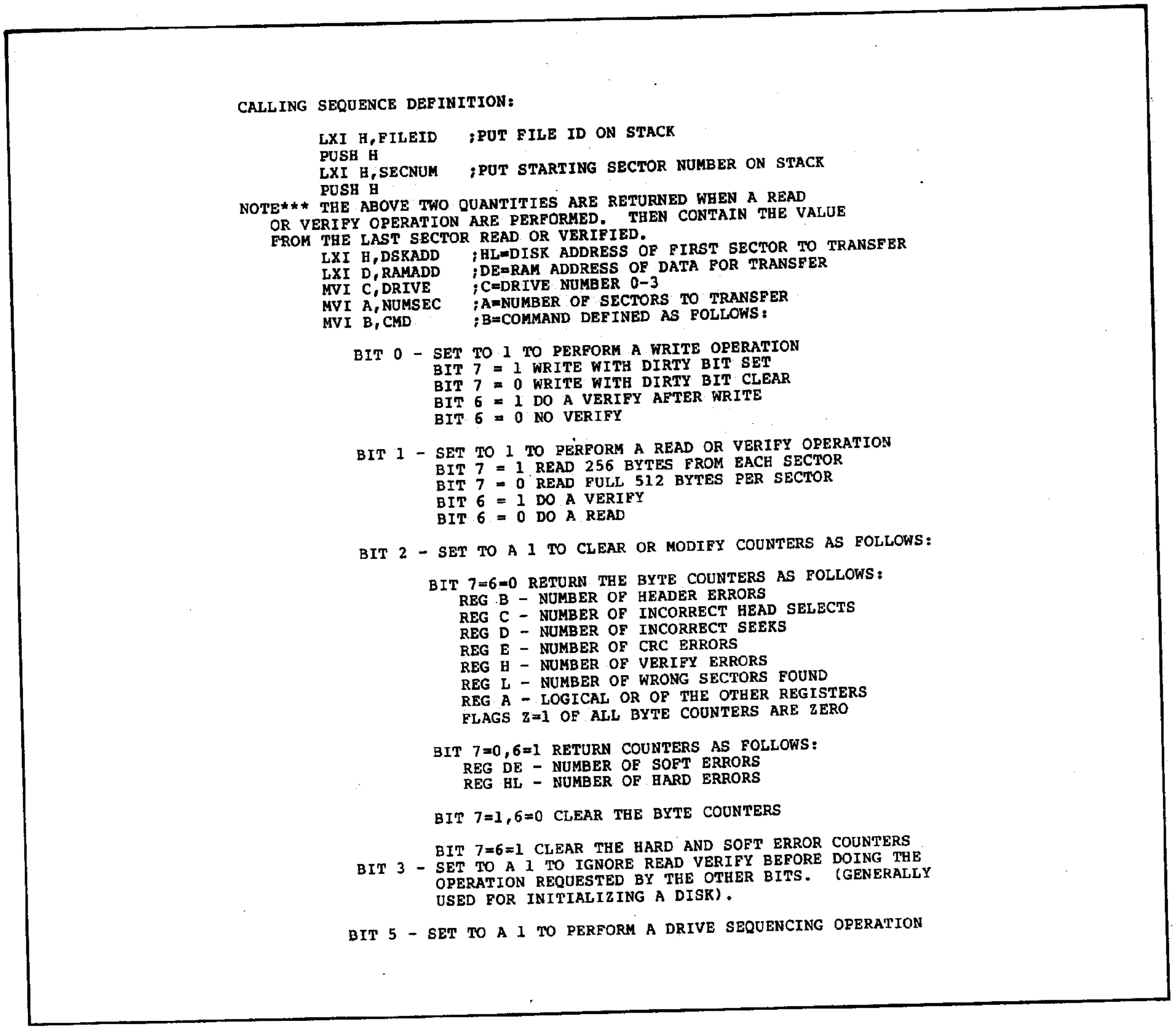
```
                    I/O ROUTINES FOR STANDARD HORIZON COMPUTER

0100    ==          IOBSZ     ==      256             ;SIZE OF USER I/O BLOCK

0000    ==          P0        ==      PADDR+0         ;ADDRESSES OF MOTHERBOARD I/O PORTS
0001    ==          P1        ==      PADDR+1
0002    ==          P2        ==      PADDR+2
0003    ==          P3        ==      PADDR+3
0004    ==          P4        ==      PADDR+4
0005    ==          P5        ==      PADDR+5
0006    ==          P6        ==      PADDR+6
0007    ==          P7        ==      PADDR+7

0300' ==            IST       ==      .               ;INPUT STATUS ROUTINE
0300' FE01                    CPI     1               ;TEST FOR DEVICE 1 POSSIBILITY
0302' 2808                    JRZ     IST1            ;JUMP TO SECOND SERIAL PORT STATUS TEST

                              ASSUME DEVICE 0 WAS INTENDED

0304' ==            IST0      ==      .               ;FIRST SERIAL STATUS PORT
0304' DB03                    IN      P3              ;INVERT STATUS FOR PROPER RESULT
0306' 2F                      CMA                     ;TEST RECEIVER DATA AVAILABLE BIT
0307' E602                    ANI     2               ;SHOW WHICH DEVICE WAS TESTED
0309' 3E00                    MVI     A,0             ;RETURN WITH INPUT STATUS IN Z FLAG
030B' C9                      RET

030C' ==            IST1      ==      .
030C' DB05                    IN      P5
030E' 2F                      CMA
030F' E602                    ANI     2
0311' 3E01                    MVI     A,1
0313' C9                      RET

0314' ==            ZCIN      ==      .               ;ALTERNATIVE ENTRY TO CIN
0314' 3E00                    MVI     A,0             ;SUBSTITUTE FIXED DEVICE NUMBER
0316' ==            CIN       ==      .               ;CHARACTER INPUT ROUTINE
0316' CD 0300'                CALL    IST             ;CHECK STATUS OF SPECIFIED DEVICE
0319' 20FB                    JRNZ    CIN             ;LOOP UNTIL DATA AVAILABLE
031B' FE01                    CPI     1               ;CHECK FOR DEVICE 1 POSSIBILITY
031D' 2805                    JRZ     CIN1            ;JUMP IF SECOND SERIAL PORT SPECIFIED
                    ASSUME PORT 0 (STANDARD SERIAL PORT) DESIRED

031F' ==            CIN0      ==      .
031F' DB02                    IN      P2              ;INPUT THE CHARACTER
0321' E67F                    ANI     7FH             ;MASK OFF PARITY BIT
0323' C9                      RET                     ;RETURN WITH CHARACTR IN A

0324' ==            CIN1      ==      .
0324' DB04                    IN      P4
0326' E67F                    ANI     7FH
0328' C9                      RET
0329' ==            OST       ==      .               ;OUTPUT STATUS ROUTINE
0329' FE02                    CPI     2               ;TEST FOR DEVICE 2 POSSIBILITY
032B' 2814                    JRZ     OST2            ;JUMP TO PARALLEL PORT STATUS TEST
032D' FE01                    CPI     1               ;TEST FOR DEVICE 1 POSSIBILITY
032F' 2808                    JRZ     OST1            ;JUMP TO SECOND SERIAL PORT STATUS TEST

                      ASSUME DEVICE 0 WAS INTENDED

0331' ==            OST0      ==      .
0331' DB03                    IN      P3              ;FIRST SERIAL STATUS PORT
0333' 2F                      CMA                     ;INVERT STATUS FOR PROPER RESULT
0334' E601                    ANI     1               ;TEST TRANSMITTER BUFFER EMPTY BIT
0336' 3E00                    MVI     A,0             ;SHOW WHICH DEVICE WAS TESTED
0338' C9                      RET                     ;RETURN WITH INPUT STATUS IN Z FLAG

0339' ==            OST1      ==      .
0339' DB05                    IN      P5
033B' 2F                      CMA
033C' E601                    ANI     1
033E' 3E01                    MVI     A,1
0340' C9                      RET

0341' ==            OST2      ==      .
0341' 3E20                    MVI     A,20H
0343' D306                    OUT     P6              ;SET COMMAND MODE
0345' 3EFE                    MVI     A,0FEH
0347' D300                    OUT     P0              ;SELECT CONTROLLER 101
0349' 3EE1                    MVI     A,0E1H
034B' D300                    OUT     P0              ;SET SELECTED CONTROLLER TO I/O MODE
```

```
034D' 3E60              MVI     A,60H           ;SET DATA MODE
034F' D306              OUT     P6              ;MOTHERBOARD STATUS BYTE
0351' DB06              IN      P6
0353' 2F                CMA
0354' CB5F              BIT     3,A             ;TEST CONTROLLER'S PO FLAG
0356' 3E20              MVI     A,20H           ;SET COMMAND MODE
0358' D306              OUT     P6
035A' 3EFF              MVI     A,0FFH          ;DESELECT CONTROLLER
035C' D300              OUT     P0              ;DEVICE NUMBER
035E' 3E02              MVI     A,2
0360' C9                RET

          TINIT FIRST REWRITES ALL RAM TO SET PARITY CORRECT

0361' ==        TINIT   ==      .
0361' 3E40              MVI     A,40H           ;DISABLE PARITY LOGIC
0363' D3C0              OUT     0C0H            ;BEFORE READING UNWRITTEN RAM
0365' 21 EC00           LXI     H,BADDR+1024    ;FIRST BYTE TO CLEAR
0368' 54                MOV     D,H
0369' 5D                MOV     E,L
036A' 01 FC00           LXI     B,-1024         ;NUMBER OF BYTES TO CLEAR
036D' EDB0              LDIR                    ;SET PARITY ON ALL RAM
036F' 3C                INR     A               ;TO 41H, PARITY ENABLE CODE
0370' D3C0              OUT     0C0H            ;REARM PARITY LOGIC

          NOW INITIALIZE MOTHERBOARD AND SET UP BOTH SERIAL PORTS

0372' AF                XRA     A               ;ZERO ACC
0373' D306              OUT     P6              ;INITIALIZE MOTHERBOARD
0375' D306              OUT     P6              ;EXTRA
0377' D306              OUT     P6              ;EXTRA
0379' D306              OUT     P6              ;EXTRA
037B' 3ECE              MVI     A,0CEH          ;2 STOPS, 16xCLOCK, 8 BITS, NO PARITY
037D' D303              OUT     P3              ;SEND TO FIRST SERIAL PORT
037F' 3ECE              MVI     A,0CEH          ;SAME CODE AS FIRST PORT
0381' D305              OUT     P5              ;SECOND PORT
0383' 3E37              MVI     A,37H           ;CMD: RTS, ER, RXF, DTR, TXEN
0385' D303              OUT     P3              ;FIRST PORT
0387' 3E37              MVI     A,37H           ;SAME CODE AS FIRST PORT
0389' D305              OUT     P5              ;SECOND PORT

                                                ;CLEAR STANDARD SERIAL PORT INPUT BUFFER
038B' DB02              IN      P2
038D' DB02              IN      P2              ;CLEAR SECOND SERIAL PORT INPUT BUFFER
038F' DB04              IN      P4
0391' DB04              IN      P4

                                                ;CARRIAGE RETURN TO INIT PRINTER
0393' 060D              MVI     B,13
0395' 21 012B           LXI     H,RWCHK         ;TEST INTERUPT ENABLE FLAG
0398' CB7E              BIT     7,M
039A' 2807              JRZ     COUT2                   ;PAGE ADDRESS OF I/O BLOCK
039C' 3E04              MVI     A,UIOB/256
039E' ED47              STAI                    ;SET INTERUPT MODE TWO
03A0' ED5E              IM2
03A2' FB                EI

          PRINTER PARALLEL OUTPUT ROUTINE

03A3' ==        COUT2   ==      .
03A3' 3E20              MVI     A,20H           ;SET COMMAND MODE
03A5' D306              OUT     P6
03A7' 3EFE              MVI     A,0FEH          ;SELECT CONTROLLER 101 AGAIN
03A9' D300              OUT     P0
03AB' 3EE1              MVI     A,0E1H          ;SET SELECTED CONTROLLER TO I/O MODE
03AD' D300              OUT     P0
03AF' 3E60              MVI     A,60H           ;SET DATA MODE
03B1' D306              OUT     P6              ;CLEAR CONTROLLER'S PO FLAG
03B3' DB00              IN      P0              ;CHARACTER TO SEND
03B5' 78                MOV     A,B             ;SET STROBE FALSE
03B6' F680              ORI     80H             ;SEND CHARACTER
03B8' D300              OUT     P0              ;TOGGLE STROBE
03BA' EE80              XRI     80H
03BC' D300              OUT     P0              ;TOGGLE STROBE
03BE' EE80              XRI     80H
03C0' D300              OUT     P0
03C2' 3E20              MVI     A,20H           ;SET COMMAND MODE
03C4' D306              OUT     P6
03C6' 3EFF              MVI     A,0FFH          ;DESELECT CONTROLLER
03C8' D300              OUT     P0              ;GET CHARACTER FOR RETURN
03CA' 78                MOV     A,B
03CB' C9                RET
```

```
03CC' ==          ZCOUT   ==       .                 ;ALTERNATIVE ENTRY TO COUT
03CC' 3E00                MVI      A,0               ;SUBSTITUTE FIXED DEVICE NUMBER
03CE' ==          COUT    ==       .                 ;CHARACTER OUTPUT ROUTINE
03CE' CD 0329'            CALL     OST               ;CHECK STATUS OF SPECIFIED DEVICE
03D1' 20FB                JRNZ     COUT              ;LOOP UNTIL READY FOR DATA
03D3' FE01                CPI      1
03D5' 2808                JRZ      COUT1             ;SECOND SERIAL PORT OUTPUT
03D7' FE02                CPI      2
03D9' 28C8                JRZ      COUT2             ;PARALLEL OPORT OUTPUT

                   ASSUME STANDARD SERIAL PORT OUTPUT

03DB' ==          COU0    ==       .
03DB' 78                  MOV      A,B               ;MOVE CHARACTER TO A
03DC' D302                OUT      P2                ;OUTPUT THE CHARACTER
03DE' C9                  RET

03DF' ==          COUT1   ==       .
03DF' 78                  MOV      A,B
03E0' D304                OUT      P4
03E2' C9                  RET


03EF'                     .LOC     UIOB+0EFH-DSPCH+BASE

                   MODE TWO INTERUPT VECTOR FOR RESTART FIVE

03EF' 0050'               .WORD    PERR

03F1' ==          CONTC   ==       .
03F1' 3E00                MVI      A,0               ;MAIN CONSOLE DEVICE NUMBER
03F3' CD 0300'            CALL     IST               ;TEST STATUS OF CONSOLE
03F6' 37                  STC
03F7' 3F                  CMC                        ;ENSURE CARRY FALSE
03F8' C0                  RNZ                        ;RETURN IF NO CHARACTER TYPED
03F9' CD 0316'            CALL     CIN               ;INPUT THE CHARACTER THAT WAS FOUND AVAILABLE
03FC' FE03                CPI      3                 ;SEE IF CHARACTER IS CONTROL-C
03FE' 37                  STC                        ;TELL SOFTWARE A CHAR WAS TYPED (OPTIONAL)
03FF' C9                  RET                        ;RETURN WITH Z-FLAG PROPERLY SET
```

APPENDIX G

DISK HANDLER FOR THE HARD DISK-HD18 (HDCOM)

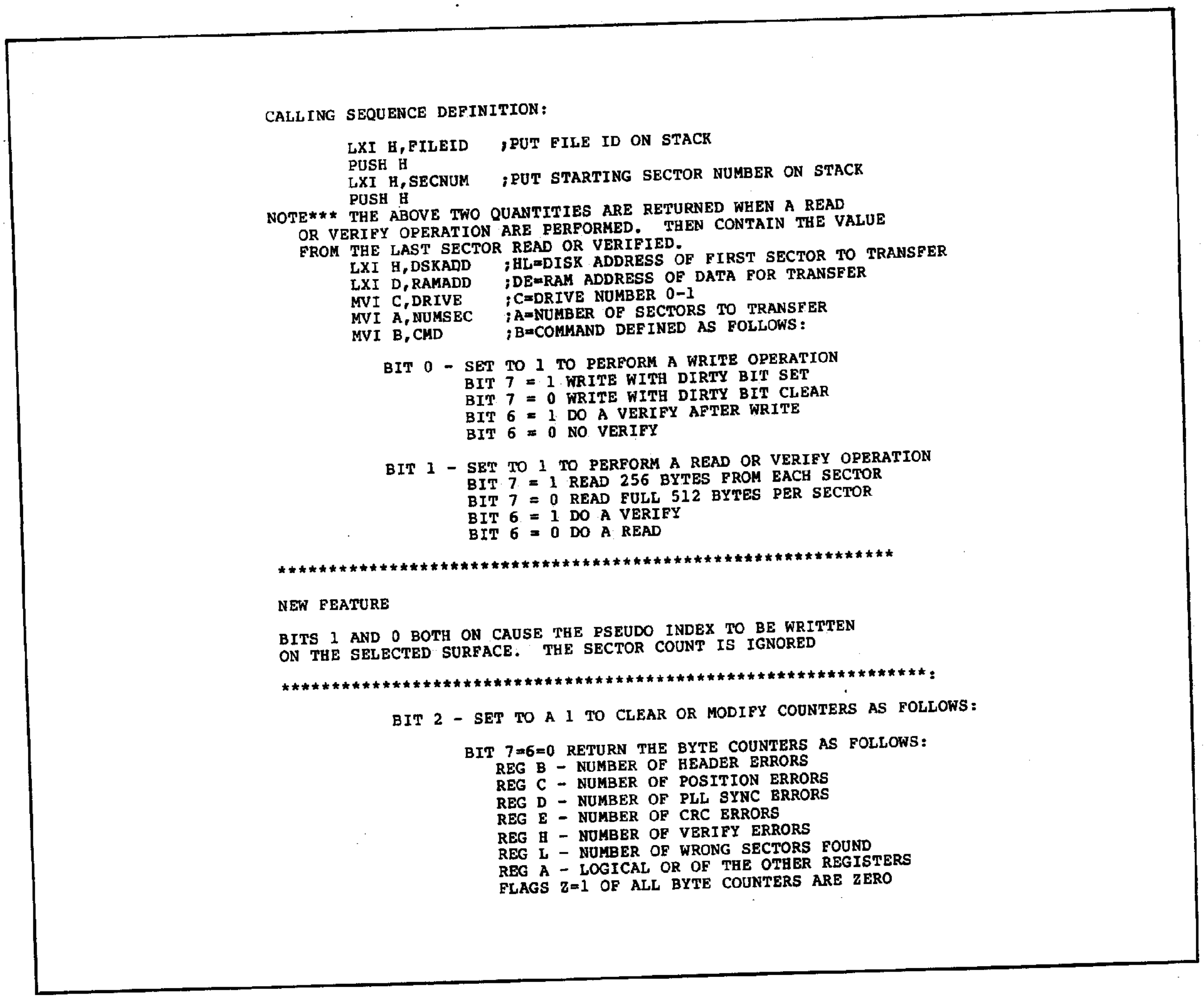


```
CALLING SEQUENCE DEFINITION:

        LXI H,FILEID    ;PUT FILE ID ON STACK
        PUSH H
        LXI H,SECNUM    ;PUT STARTING SECTOR NUMBER ON STACK
        PUSH H
NOTE*** THE ABOVE TWO QUANTITIES ARE RETURNED WHEN A READ
   OR VERIFY OPERATION ARE PERFORMED.  THEN CONTAIN THE VALUE
   FROM THE LAST SECTOR READ OR VERIFIED.
        LXI H,DSKADD    ;HL=DISK ADDRESS OF FIRST SECTOR TO TRANSFER
        LXI D,RAMADD    ;DE=RAM ADDRESS OF DATA FOR TRANSFER
        MVI C,DRIVE     ;C=DRIVE NUMBER 0-3
        MVI A,NUMSEC    ;A=NUMBER OF SECTORS TO TRANSFER
        MVI B,CMD       ;B=COMMAND DEFINED AS FOLLOWS:

           BIT 0 - SET TO 1 TO PERFORM A WRITE OPERATION
                   BIT 7 = 1 WRITE WITH DIRTY BIT SET
                   BIT 7 = 0 WRITE WITH DIRTY BIT CLEAR
                   BIT 6 = 1 DO A VERIFY AFTER WRITE
                   BIT 6 = 0 NO VERIFY

           BIT 1 - SET TO 1 TO PERFORM A READ OR VERIFY OPERATION
                   BIT 7 = 1 READ 256 BYTES FROM EACH SECTOR
                   BIT 7 = 0 READ FULL 512 BYTES PER SECTOR
                   BIT 6 = 1 DO A VERIFY
                   BIT 6 = 0 DO A READ

           BIT 2 - SET TO A 1 TO CLEAR OR MODIFY COUNTERS AS FOLLOWS:

                  BIT 7=6=0 RETURN THE BYTE COUNTERS AS FOLLOWS:
                     REG B - NUMBER OF HEADER ERRORS
                     REG C - NUMBER OF INCORRECT HEAD SELECTS
                     REG D - NUMBER OF INCORRECT SEEKS
                     REG E - NUMBER OF CRC ERRORS
                     REG H - NUMBER OF VERIFY ERRORS
                     REG L - NUMBER OF WRONG SECTORS FOUND
                     REG A - LOGICAL OR OF THE OTHER REGISTERS
                     FLAGS Z=1 OF ALL BYTE COUNTERS ARE ZERO

                  BIT 7=0,6=1 RETURN COUNTERS AS FOLLOWS:
                     REG DE - NUMBER OF SOFT ERRORS
                     REG HL - NUMBER OF HARD ERRORS

                  BIT 7=1,6=0 CLEAR THE BYTE COUNTERS

                  BIT 7=6=1 CLEAR THE HARD AND SOFT ERROR COUNTERS
           BIT 3 - SET TO A 1 TO IGNORE READ VERIFY BEFORE DOING THE
                   OPERATION REQUESTED BY THE OTHER BITS.  (GENERALLY
                   USED FOR INITIALIZING A DISK).

           BIT 5 - SET TO A 1 TO PERFORM A DRIVE SEQUENCING OPERATION
```

```
                  BIT 4 = 1 PERFORM A HARDWARE RESET ON THE SELECTED DRIVE

                  BIT 4 = 0 PERFORM THE SEQUENCE UP OR DOWN AS CONTROLLED
                            BY BIT 7

                  BIT 7 = 1 PERFORM A SEQUENCE UP OPERATION
                            NOTE:IF DRIVE IS ALREADY SEQUENCED UP
                                 THIS WILL PERFORM A RESET TO DRIVE
                                 ****************************************
                                 THE ROUTINE WILL RETURN A DRIVE NOT READY
                                 FOR COMMAND ERROR (126), UNTIL THE DRIVE IS
                                 UP TO SPEED AND READY.  WHEN FIRST POWERING
                                 A DRIVE UP, IT IS REQUIRED THAT HDCOM BE
                                 CALLED WITH A SEQUENCE UP COMMAND UNTIL IT
                                 RETURNS WITH OUT ERROR.  THE LENGTH OF ANY
                                 SINGLE CALL WILL BE LESS THAN 5 MILLISECONDS.;
                  BIT 7 = 0 PERFORM A SEQUENCE DOWN OPERATION

        CALL HDCOM      ;PERFORM THE CALL TO HDCOM
                        ;THE DISK ADDRESS (HL), RAM ADDRESS(DE)
                        ;SECTOR NUMBER (STACK) AND NUMBER OF SECTORS (A)
                        ;ARE ALL UPDATED BY HDCOM.  AT THE END OF AN OPERATION
                        ;THEY CONTAIN THE VALUES OF THE NEXT CONSECUTIVE
                        ;SECTOR IF NO ERROR OCCURRED, OR POINT TO THE
                        ;SECTOR IN ERROR IF AN ERROR DID OCCUR.
        JNZ ERROR       ;THE Z FLAG IS SET TO 0 ON AN ERROR
                        ;THE B REGISTER CONTAINS AN ERROR NUMBER IN
                        ;THE 7 LSB AND THE MSB = 1 IF ANY SECTOR READ OR
                        ;VERIFIED WAS DIRTY.
```

APPENDIX H

DISK HANDLER FOR THE FIVE INCH HARD DISK (MWCOM)



```
CALLING SEQUENCE DEFINITION:

        LXI H,FILEID    ;PUT FILE ID ON STACK
        PUSH H
        LXI H,SECNUM    ;PUT STARTING SECTOR NUMBER ON STACK
        PUSH H
NOTE*** THE ABOVE TWO QUANTITIES ARE RETURNED WHEN A READ
   OR VERIFY OPERATION ARE PERFORMED.  THEN CONTAIN THE VALUE
   FROM THE LAST SECTOR READ OR VERIFIED.
        LXI H,DSKADD    ;HL=DISK ADDRESS OF FIRST SECTOR TO TRANSFER
        LXI D,RAMADD    ;DE=RAM ADDRESS OF DATA FOR TRANSFER
        MVI C,DRIVE     ;C=DRIVE NUMBER 0-1
        MVI A,NUMSEC    ;A=NUMBER OF SECTORS TO TRANSFER
        MVI B,CMD       ;B=COMMAND DEFINED AS FOLLOWS:

            BIT 0 - SET TO 1 TO PERFORM A WRITE OPERATION
                    BIT 7 = 1 WRITE WITH DIRTY BIT SET
                    BIT 7 = 0 WRITE WITH DIRTY BIT CLEAR
                    BIT 6 = 1 DO A VERIFY AFTER WRITE
                    BIT 6 = 0 NO VERIFY

            BIT 1 - SET TO 1 TO PERFORM A READ OR VERIFY OPERATION
                    BIT 7 = 1 READ 256 BYTES FROM EACH SECTOR
                    BIT 7 = 0 READ FULL 512 BYTES PER SECTOR
                    BIT 6 = 1 DO A VERIFY
                    BIT 6 = 0 DO A READ
 *************************************************************

 NEW FEATURE

 BITS 1 AND 0 BOTH ON CAUSE THE PSEUDO INDEX TO BE WRITTEN
 ON THE SELECTED SURFACE.  THE SECTOR COUNT IS IGNORED
 ****************************************************************;

             BIT 2 - SET TO A 1 TO CLEAR OR MODIFY COUNTERS AS FOLLOWS:

                    BIT 7=6=0 RETURN THE BYTE COUNTERS AS FOLLOWS:
                       REG B - NUMBER OF HEADER ERRORS
                       REG C - NUMBER OF POSITION ERRORS
                       REG D - NUMBER OF PLL SYNC ERRORS
                       REG E - NUMBER OF CRC ERRORS
                       REG H - NUMBER OF VERIFY ERRORS
                       REG L - NUMBER OF WRONG SECTORS FOUND
                       REG A - LOGICAL OR OF THE OTHER REGISTERS
                       FLAGS Z=1 OF ALL BYTE COUNTERS ARE ZERO
```

```
              BIT 7=0,6=1 RETURN COUNTERS AS FOLLOWS:
                 REG BC - NUMBER OF NONDATA ERRORS
                 REG DE - NUMBER OF SOFT ERRORS
                 REG HL - NUMBER OF HARD ERRORS

              BIT 7=1,6=0 CLEAR THE BYTE COUNTERS

              BIT 7=6=1 CLEAR THE HARD AND SOFT ERROR COUNTERS
         BIT 3 - SET TO A 1 TO IGNORE READ VERIFY BEFORE DOING THE
              OPERATION REQUESTED BY THE OTHER BITS.  (GENERALLY
              USED FOR INITIALIZING A DISK).  CAUSES MWCOM TO USE INDEX
               PULSE AND SECTOR PULSES TO LOCATE SECTORS.

        BIT 5 - SET TO A 1 TO PERFORM A DRIVE SEQUENCING OPERATION

               BIT 4 = 1 POSITION DRIVE OVER TRACK 0

               BIT 4 = 0 PERFORM THE SEQUENCE UP OR DOWN AS CONTROLLED
                         BY BIT 7

               BIT 7 = 1 PERFORM A SEQUENCE UP OPERATION
                         NOTE: THIS MEANS TO POSITION OVER TRACK 0
               BIT 7 = 0 DESELECT DRIVE

CALL MWCOM      ;PERFORM THE CALL TO MWCOM
                ;THE DISK ADDRESS (HL), RAM ADDRESS(DE)
                ;SECTOR NUMBER (STACK) AND NUMBER OF SECTORS (A)
                ;ARE ALL UPDATED BY MWCOM.  AT THE END OF AN OPERATION
                ;THEY CONTAIN THE VALUES OF THE NEXT CONSECUTIVE
                ;SECTOR IF NO ERROR OCCURRED, OR POINT TO THE
                ;SECTOR IN ERROR IF AN ERROR DID OCCUR.
JNZ ERROR       ;THE Z FLAG IS SET TO 0 ON AN ERROR
                ;THE B REGISTER CONTAINS AN ERROR NUMBER IN
                ;THE 7 LSB AND THE MSB = 1 IF ANY SECTOR READ OR
                ;VERIFIED WAS DIRTY.
```

APPENDIX I

GLOSSARY

The following are basic terms used in this manual.

ACCESS

The process of obtaining data from a diskette or Hard disk.

ACCOUNT

A grouping of files on hard disk. An account can be considered logically equivalent to the group of files on a single diskette.

APPLICATION PROGRAM

A program written to perform a specific task such as word processing or maintaining a general ledger.

BACKUP

Additional copies of system or data files that protect you against informatin loss from power transients, equipment malfunction, or operator error.

BOOTSTRAP

The process of initializing an operating system on your computer.

CHARACTER

Any letter, number or special symbol that is displaced on the screen, placed in memory or stored on a diskette or hard disk.

DATA

Any information that can be processed by computer.

DATA DISKETTE

A diskette used to store data generated by the user. For example, a word processing data diskette stores documents, and a mailing list data diskette stores mailing list information.

DIB

Data Incremental Block (previously called hunk.) The smallest unit of storage that can be allocated/flagged on the hard disk. A DIB is composed of 16 sectors.

DIRECTORY

A table of information about individual files.

DISK

See HARD DISK.

DISKETTE

The flexible magnetic media on which programs and data are stored. North Star diskettes are 5-1/4" in diameter.

DISKETTE DRIVE

The unit that comprises the spindle, recording/playback heads, drive actuators, etc. This unit contains the openings into which your diskettes are inserted.

DUAL

A dual capacity diskette is single-sided, with double-density. Both dual and quad capacity data and program diskettes can be used with HDOS.

FILE

A unit of storage on a diskette or hard disk, that is grouped, and accessed, under one name. A file is a logical subdivision while a SECTOR is a physical portion of the diskette or hard disk.

FILE BLOCK

A unit of information equal to 256 bytes.

FLOPPY DISK

See DISKETTE.

FORMATTING A DATA DISKETTE

The process that creates the file structures on a diskette that must be present before a blank diskette can be used for storing data.

HARD COPY

The printed output of stored or processed data.

HARD DISK

A storage medium offering greater storage capacity, and considerably shorter access time than a diskette. Backup is performed via diskette or tape.

HUNK

Synonymous with DIB.

INITIAL RECOVERY DISKETTE

Contains prerecorded North Star Application Software.

I/O

Abbreviation of input/output, meaning either or both operations.

MEMORY

The part of the computer that can store information. Because the program for any function being performed must be in main memory during operation, the size of the computer memory (measured in bytes) is a good indication of a computer's potential. A byte can store one character; so, for example, 64K bytes of memory represents storage for approximately 64,000 characters.

The two most common types of main memory are "Read-Only Memory" (ROM) and "Random Access Memory" (RAM), also called Read/Write Memory.

Other types of memory are Programmable Read-Only Memory (PROM), which is a ROM which may be altered, and ERasable Programmable Read-Only Memory (EPROM), which is a PROM that can be reused several times.

The contents of main memory can be permanently stored on media such as diskettes, hard disks, tape cartridges, reel to reel tape, and punched or encoded cards.

MENU

A list of possible activities a program can perform. This list is presented on the video screen so the user can choose from its alternatives.

NUMERIC

Means 'pertaining to numbers.' A numeric field is one where only numbers, blanks, and certain symbols such as commas, periods, dollar signs, percent signs, etc., can be entered.

OPERATING SYSTEM

The programs designed to monitor and coordinate tasks created by application programs. The operating system controls input and output of data between peripherals and memory, governs file management on hard disk by performing utlity functions such as copy, delete and create, and loads and executes application programs.

PROGRAM

A set of logically ordered instructions designed to direct the computer through a particular operation or set of operations. Also referred to as "software."

QUAD

A quad capacity diskette is double-sided, with double density. It can store twice as much information as a dual density diskette.

RAM

Random Access Memory, also known as Read/Write Memory.

READ

The process of picking up stored data and transfering it to the internal memory. Reading always occurs from a peripheral unit to the internal memory.

RECORD

Basically a group of fields. For example, a list containing the name, address and phone number of everyone at a party of 20 people is a list of 20 records, with each record containing three fields.

RETURN

This key has many uses, depending on the program. Generally the RETURN key indicates the end of a data input operation.

ROM

Read-Only Memory.

SCRATCH DISKETTE

A new blank diskette or one that contains material you don't wish to keep permanently.

SECTOR

A contiguous 512 byte section of a hard disk or diskette track.

SOFTWARE

The computer instructions that direct computer hardware to perform tasks. There are different categories of software: application software, operatig systems, language compilers, etc.

SYSTEM DISKETTE

Contains the prerecorded programs that make up the North Star System or Application Software you purchased.

WRITE

The process of recording information in internal memory; the transfer of information from internal memory to an external storage or output medium, such as a diskette, hard disk, or printer.

INDEX

Reader Comments

Please use this postpaid mailer to make us aware of the strengths and weaknesses of this North Star manual. Specific errors or deficiencies should be referenced by page numbers and paragraph headings. Attach additional sheets if you require more space for your comments.

We will carefully consider your suggestions for incorporation in future versions of this manual. Thank you.

Do you find the manual easy to use and understand? □ YES □ NO

Do you think certain aspects should be organized differently? If so, please explain below.

Are there specific points or issues in the manual that need clarification or correction? Please give reference page numbers and paragraph/table/figure headings and describe the problem below.

Additional comments:

NAME OF MANUAL

PART NUMBER

NAME

DATE

COMPANY NAME

STREET ADDRESS

CITY

STATE

ZIP CODE

FOLD FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 2106 SAN LEANDRO, CA

POSTAGE WILL BE PAID BY ADDRESSEE

North Star Computers, Inc.
ATTENTION: Technical Publications
14440 Catalina Street
San Leandro, CA 94577 USA

NorthStar™
North Star Computers, Inc.
14440 Catalina St., San Leandro, CA 94577 USA
(415) 357-8500 TWX/Telex (910) 366-7001