LOTUS 700
DISC CONTROLLER UTILITIES AND DIAGNOSTICS MANUAL

POINT 4
DATA CORPORATION

P O I N T   4   D A T A   C O R P O R A T I O N
2569 McCabe Way, Irvine, California 92714
(714) 754-4114

```
  4444  4
 4444    444
444   4  4444
4   444  4444

4444444  4444
 444444  444
   4444  4
```

LOTUS 700 DISC CONTROLLER

UTILITIES AND DIAGNOSTICS

PRELIMINARY

NOTICE

Every attempt has been made to make this reference manual complete, accurate and up-to-date. However, all information herein is subject to change due to updates. All inquiries concerning this manual should be directed to POINT 4 Data Corporation.

P R E L I M I N A R Y

UTILITIES

1.0 GENERAL

DSKUT is a disk utility package that runs in a disk sub-system which equipped with POINT 4's 700 disk controller. The package contains three off-line utility programs used for initializing disk surfaces and maintenance of disk files, namely format, surface verify and copy. Each program is compatible with a wide variety of disk drives. The operation is performed on surface by surface of the disks on the basis of controlling parameters entered by user to the program.
While in operation, progress is reported frequently to the console to indicate the ongoing activity. In case of a hardware malfunction, the status of the controller will be displayed.
Once the program is initialized for an operation, it retains all the entered parameters after completion. Therefore, in the next operation, if any, default to the respective requests for parameters wherever they would be the same. Enter new if it is to be different from previous entry.

1.1 PARAMETER ENTRY

When doubt on entering parameters, type "H" for display of help messages.
All numeric entries are in decimal. Disk surface (Head) numbers are physical number starting from Ø, regardless whether the drive is SMD, CMD, or MMD type

All entries are terminated with carriage return (CR). Any entry error is correctable before CR. Use backspace or rubout key to erase the last character. Continuous erasure will delete entire input. Any unacceptable entry will cause a "WHAT?" response.

1.2 Numeric Display

All numeric displays are in decimal. Only the controller and drive status words are shown in octal.

1.3 Compatible Drives

A list of drives recognized by DSKUT is shown in Appendix 1. The list may be expanded in the future to include new types of drives. An expanded list will be reflected in the program version number.

1.4 Controller Device Code

DSKUT will only work with a controller of device code 27.

2.0 Operational Description

2.1 Format Program

Before data can be stored and retrieved from the disk, all disk surfaces must be formatted. Formatting is the process of identifying all disk sectors by writing to each sector its own address called a header, and clearing out the 256 word data area.

The format program carries out the process in two passes. In the first pass, disk surfaces are formatted one track at a time, surface by surface. The progress is reported on the console as each surface is being operated on:

```
FORMATTING Ø 1 . . .
```

When all surfaces are formatted, the second pass begins. Here the headers are verified be reading one track of headers into the program buffer, compare each header with the program generated reference to ensure it is correct. The verification process is done on surface by surface. The progress report appears on the console as each surface is operated on:

```
VERIFYING Ø 1 . . .
```

Any error found in header comparison will cause the program to re-write headers on that track up to 5 times. If error persists, the track and surface addresses are logged. Then upon completion of pass two, an error summary is shown.

2.2 Verify Program

To ensure the integrity of the disk surfaces and to isolate all bad spots, if any, each surface is thoroughly analyzed for its ability to retain the data under the worst case conditions. The Verify program performs the surface analysis and verification process in 8 write/read passes. In each pass a different worst case bit pattern is used, starting with 13427Ø. First, the pattern is written onto every sector one track at a time, surface by surface. When done, read data back into the program buffer one track at a time, verify the read back with that written to see if the bit pattern has been altered. If an error is discovered, re-read the sector up to 5 times. And if the error persists, log that sector as a hard error then proceed on to the next track. This verify process is also done on surface by surface. A pass is completed when the last track of the last surface has been verified. In the next pass, the previous worst case bit pattern is rotated right one position and the same WR cycle is repeated. This continues until the pattern has been rotated 7 times, or 8 total passes.

While the program is performing its function, the progress is reported on the console as it begins on a surface. The display appears as follows:

```
WRITING Ø 1 . . .
READING Ø 1 . . .
COMPLETED PASS 1

WRITING Ø 1 . . .
READING Ø 1 . . .
COMPLETED PASS 2
       .
       .
       .
WRITING Ø 1 . . .
READING Ø 1
COMPLETED PASS 8
```

When pass 8 is completed, an error summary is shown on the console. Any nonzero soft and hard errors will displayed in tabular form with their respective counts and addresses. For example:

```
TOTAL ERRORS:  HARD=8     SOFT=2
COUNT      SURFACE      TRACK      SECTOR
SOFT 2        Ø           2           1
HARD 8        Ø           2          31
```

Soft errors are recoverable errors, that is it succeeded after a number of retries under 5. Hard errors are irrecoverable errors, that is it still failed after 5 retries. The example above shows 8 hard errors all at the same spot on the disk, indicating it failed in each of the 8 passes.

2.2.1 How to Handle Bad Sectors

If there is any hard error shown in the table summary, the following prompt will appear:

FLAG OR CHAIN BAD SECTORS:

It suggests the option of either flag or chain the bad sectors. Flagging bad sectors is to take them out of operation so they can not be used again. This is done by re-writing the respective headers with the bad sector bit set. Thus when attempting to access a bad sector, the controller would abart the operation and report it as such in the status word.

Chainning bad sectors is to replace the bad sectors with good sectors located in an alternate track selected by the user. This done by re-writing the respective headers with chain bit set along with address of the alternate sector. Sector chainning is always on the same surface where each bad sector was found. Thus when attempting to access the bad sector, the controller is then directed to the alternate sector instead. This operation is transparent to the user.

2.3 Copy Program

The Copy program is used to duplicate data files from any disk surfaces to any other surfaces, within the same drive or to another drive.

The copy process is read one track of data from the source disk into the program buffer, write it out to the same track on the destination disk. Then read data from destination track and verify its content. This process is repeated for every track. Thus, copying proceeds on a surface by surface bases and reporting progress in the console as follows:

```
COPYING Ø 1 . . .
```

If an error is found in the data verification phase, the program retry write-read-verify cycle up to 5 times. If the error persists, then it is logged as a hard error.

2.4 Status Error Display

In all three programs, if an error (Bit 15) is detected in the controller status word, the operation is aborted immediately and display the status work in octal as:

```
CONTROL STATUS = XXXXXX
```

Followed by interpretation of error in English.

When the drive is write protected (write disabled), it will prevent the program from writing onto it. In fact, it will abort immediately. However, the program can read from it.

3.0 Procedure

3.1 Program Start

DSKUT is self starting when loaded into memory. Re-start is from location 2. When re-starting from 2, all previously entered parameters will be lost.

3.2 Program Greetings

After starting, the program prints the following greeting:

```
POINT 4 DISK SUBSYSTEM UTILITY
VERSION 1.0 FEB 03, 1980
FOR HELP, TYPE "H"
```

3.3 Select Program

Following the initial greetings, or after completing an operation, the prompt for selecting program is:

PROGRAM NAME:

The name to enter is one of following:

FORMAT for formatting only

VERIFY for verifying surfaces only

FORMAT, VERIFY for format followed by verify

COPY for duplicating files to other surface

3.4 Format and Verify Programs

Sections 3.4.1 through 3.4.4.1 outline the procedure for entering parameters for the Format and Verify programs.

3.4.1 Drive Type

The input prompt is

TYPE OF DRIVE:

Enter either the drive mnemonic or code number if known.
If not, consult the drive directory in Appendix 1 or type "H" for display of directory. Default uses the old entry if exists.

3.4.2 Drive Number

The input prompt is

DRIVE NUMBER:

Enter the number of controller port where the drive cable is connected to. This number should agree with the plug number in the drive. The range is Ø through 3. Default is Ø if initializing the first time. Else, it is no change from the old entry.

3.4.3 Surface Number

The input prompt is

SURFACE(S):

Enter the number of the surface to be operated on. Separate numbers with a comma if more than one surface are to be entered. The numbers may be entered in any order. The program will process each surface in the order entered. Default will activate all surfaces if entering the first time. Else no change from the last operation.

3.4.3 Start Operation

After the surface number entry, the follow start prompt will appear

WHEN READY, STRIKE ANY KEY TO START

It means that proceed if the drive is on-line. If not, set it on-line, wait for ready light to come on then start.

3.4.4 Disposition of Bad Sectors

All bad sectors found by the Verify program will be listed in a summary displayed at the end of the verify operation. The request for direction for disposal of bad sectors appears as follows:

FLAG OR CHAIN BAD SECTORS:

Enter either FLAG or CHAIN. See Operational Description Sections 2.2.1 for explanation of flagging and chaining of bad sectors.

3.4.4.1 Alternate Track Selection

If the Chain option is chosen for disposing bad sectors, then the prompt

ALTERNATE TRACK:

will appear. Enter the selected track number in decimal. The program will verify the entry. If it exceeds the last track number of the drivé, it will be rejected with "WHAT?" response. If the selected track also contains bad sectors, the message

```
TRACK HAS BAD SECTORS
```

will appear and above prompt is repeated. Another track must be selected.

3.5 Copy Program

Sections 3.5.1 through 3.5 describe the procedure for entering parameters for the Copy program.

3.5.1 Source Drive Type

The input prompt for source drive is

```
FROM DRIVE-
TYPE:
```

Enter either the mnemonic or code number of the source drive if known. If not consult the drive directory in Appendix 1 or type "H" for its display. Default uses old entry if exists.

3.5.2 Source Drive Number

The input prompt is

```
NUMBER:
```

Enter the number of controller port where the source drive is connected to. Default uses old entry if exists.

3.5.3 Source Surface Number

The input prompt is:

```
SURFACE(S):
```

Enter the number of surface to copy from. Separate numbers with a comma if more than one surface are to be entered. There is a one to one correspondent to the destination surfaces. That is, the first source surface will be copied onto the first destination surface as entered. Default uses old entry if exists, also copy from all surfaces.

3.5.4 Copy Destination

The input prompt is:

TO THE SAME DRIVE?

Enter "Y" for yes, "N" for no.

3.5.5 Copy to Same Drive

When copy from surface to surface within the same drive, the next prompt is:

TO SURFACE(S):

Enter the number of surface receiving the copy. Separate numbers with a comma if more than one surface are to be entered. If a destination surface is same as the source, a message

COPY ITSELF?

will appear. The destination entry is then cancelled and must be re-entered.

3.5.6 Start Copying in the Same Drive

After entering the destination surface numbers, the prompt for start operation

WHEN READY, STRIKE ANY KEY TO START

It means that proceed if the drive is on-line. If not, set it to on-line, wait for ready light to come on then start.

3.5.7 Copy to Other Drive

For copying onto another drive, the following requests for parameters about the destination will appear.

3.5.7.1 Destination Drive Type

The destination drive may be of any type listed in the directory. Enter either the mnenonic or code number of the drive if known when the prompt

```
TO DRIVE-
TYPE:
```

is shown. Consult drive directory in Appendix 1 or type "H" for the delay if neither is known or in doubt.

3.5.7.2 Destination Drive Number

The input prompt is:

```
NUMBER:
```

Enter the number of controller port where the destination drive is connected to. It must be different from source port. Else, a message of

```
DRIVE NUMBER CONFLICT
```

will appear and the input prompt is repeated.

3.5.7.3 Destination Surface Numbers

The input prompt is:

```
SURFACE(S):
```

Enter the number of surface which is to receive the copy. If more than one number is to be entered, separate numbers with a comma in between.

3.5.7.4 Start Copying to Other Drive

See Section 3.5.6

How to Make a New Entry to the Drive Directory

The drive directory is used for locating the lookup table when the executive passes a mnemonic or code number of a selected drive in the utility or diagnostics program.

The directory is 79 entries long. Currently (May 5, 1980) there are 51 valid entries and leaves 28 available for future use. To make a new entry permanent it would require the text editor for data entry, then assemble all files to get the new object file. Figure 1 illustrates the organization of the directory to aid the description given below.

Main Directory

The main directory is called DDIR and contains 79 entries, one for each type of drive. Each entry consists of 3 words. A null entry (not used) is indicated by a -1 in the first word. For making a new entry, replace -1 with the character count of the assigned mnemonic. The second word points to the dummy mnemonic text location in the text file (UTTX for utility, DDTX for diagnostics). The text name is DNn, n is 1 thru 79. The third word points to the location of the lookup table and is labeled DTn, n is 1 thru 79. Both DN and DT have the same n number and correspond to the assigned drive code number, if the mnemonic is not used.

Lookup Table - DTn

Each table consists of 4 words: Word Ø signifies whether the drive is a removable storage module type or a cartridge and fixed media (CMD) combination. It is set to Ø for SMD and -1 for CMD. Word 1 is the last head number of the drive, e.g. head 1 would be the last of 2 heads. Word 2 is sectors per track. Word 3 is the last cylinder number. It is equal to total cylinder minus 1. A skeleton table must null out all 4 words. The unusual table is also reflected in the directory, first word is -1.

Mnemonic Text -DNn

Replace the null character (   ) in the dummy DNn text with the character string of the assigned mnemonic.

Directory Display

The directory is displayed on the console when user asks for it. It may display whole directory or in part.

The display routine works from a list of texts which consists of labels DTX1 thru DTX79 with some -1s in between for non-existent texts (drives). Each number in DTX corresponds to DN and DT, and each DTX is one line of display describing the characteristics of the drive. The drives are grouped under the manufacturer's name. To add a new display, replace the proper -1 slot with a new DTX number and add the descriptive text to the text file.

Drive Code Number

The code numbers are implied in the main directory. There are no actual numbers exist. It is the order of directory entry called code number. That is, the drive in the first entry has a code number 1, second entry has 2 and so on in ascending order. The executive merely passes the directory index number to locate the drive table.

DIAGNOSTICS

1.0 Introduction

DC700 is a versatile software package which provides fast accurate and complete testing of POINT 4's 700 disk controller hardware in a production environment. With the in-depth fault isolation and user interactive capabilities, the software is also a powerful tool for troubleshooting malfunctioning units.

The package contains three separate programs selectable to run under three different circumstances. First is the local logic test. A thorough check of the accessible logic circuits and CPU interface. Secondly is the drive test. Every function and capability of the controller is exercised and verified with a disk drive. Third and finally the tester. A multi-drive configuration is electronically simulated by the test board in the CPU mainframe and thus tests with four drives.

The philosophy of the test program is to begin test with the most basic operation or function, then progressively spreading outward to overlap other related function. The overlapping continues until all complex functions are covered. At anytime a malfunction or an out of spec condition is detected, a detailed description of the test and what has occurred is displayed in the console. After analyzing the information, the user may proceed with one of many options offered by the program.

1.1 Personnel

DC700 diagnostic programs are intended to be used by qualified persons who are technically knowledgeable of POINT 4 CPU assembly programming language and the theory of disk controller operation.

1.2 Compatible Drives

A list of disk drives that are compatible with the drive test is in Appendix 1. The list is also callable for display in the console in the program.

2.0 Operational Description

Figure 1 illustrates the overall logic flow of DC700. The operations of some of the boxes are described in the following sections.

2.1 Program Initialization

Program initialization is the process of pre-setting the operating limits and conditions of the program prior to the start of actual operation. It is done by entering the proper parameter by the user as each request appears. Once initialized, the program retains the information until they are changed by the user as different situation requires. For convenience, there are help messages for every entry when needed.

2.1.1 Device Code

The device code of a controller may be non-standard. The program has the ability to address a controller of any device code. Whatever, it must be entered to the program. Default is standard 27.

2.1.2 Stop Option

Whether to stop or proceed after an error display is an option set during initialization. This option is reversible anytime by a command when the prompt ( ) is shown.

2.2 Test Mode Selection

A test mode is a method in which the eontroller is being tested. There are three modes to choose from: Logic, Drive, and Tester.

The selection is depending upon the degree of testing and the availability of a drive. A test mode is selected by entering the name when the request appears.

2.2.1 Logic Mode

The logic mode tests only the logic circuits local to the controller and the interface with CPU. No other external interface is required. The purpose of logic test is to verify that all the basic control logic, registers, IO flags, PROM programming and CPU interface are functioning properly. The following tests are performed:

1. Register Reset
2. Device Addressing
3. Control Full Status Set/Reset
4. Drive Done Status Set/Reset
5. Clear Done Status
6. Extended Memory Address Register W/R
7. Memory Address Register W/R
8. Surface, Sector Address Register W/R
9. Busy Flag Set/Reset
10. Read Buffers All Ones Data and Interrupt Net
11. Read Buffers All Zeros Data
12. Head and Sector PROM Programming

Once entered the logic mode, tests are cycled 1 through 12 continuously until manually stopped. At the end of the first cycle, the program will display the results of PROM programming test which shows the upper limits that were programmed into the PROMs for addressing the disk. The display will appear only once each time logic mode is entered.

2.2.2 Drive Mode

The drive mode tests all the disk I/O and other control capabilities of the controller. The drive interface circuits, DMA channels, disk I/O timing and other status bits which were not testable in logic mode are covered in this mode.

The program can work with many different types of drives (See Section 1.2). All the program needs to know are the name of drive and port number of the controller where the drive is connected.

The following test steps are performed in this mode:

1. Recalibrate Drive
2. Format One Sector
3. Read One Sector Format
4. Write One Sector Header
5. Format 32 Sectors
6. Read 32 Sector Data
7. Read One Sector Data
8. Memory Address Register Increment
9. RW Timeout
10. Surf/Sect Address Error Status
11. Cylinder Address Error Status
12. Header Word Unused Bits
13. Format Cross Cylinder
14. Format A Cylinder
15. Format Last Removable Surface
16. Read Format from Last Sectors
17. Data Verify Error Status
18. Sector Check
19. Write Header
20. Overlap Seek
21. Bad Sector Status
22. Sector Chaining
23. Double Chain Sector
24. Bad Alternate Sector
25. Read FIFO Buffer

Once started test in this mode, tests 1 through 25 will be recycled in an endless loop until manually stopped. At the end of each cycle, a pass count is printed to show the number of cycles it has been through.

2.2.3 Tester Mode

The tester mode is an extension of drive mode except that an actual drive is not required. The tester board in the CPU in effect emulates a multi-drive hookup which enables testing in this configuration.

The following test steps are executed in this mode:

1. Release Drive
2. Drive Done Status, Drives Ø - 3
3. Read Sepcial Header, Drives Ø - 3
4. Trespass and Release Drive
5. Read Drive Status, Drive Ø - 3
6. Check Drive Status Bits 1, 3, 4, & 6
7. Check Drive Status Bits Ø, 9, & 10 - 12
8. Format and Read Format, 1 Sector
9. Write and Read Header
10. Bad Sector Status
11. 1-Sector Data Write, Read and Verify
12. 2-Sector Data Write and Read
13. Sector Chainning
14. Overlap Seek
15. Read Offset

2.3 Test Steps

In any test mode, the controller is stepped through a series of tests. Each step is designed exclusively to test a particular logic or function. To achieve the lowest level of fault isolation, a step is further divided into numerous stages. For example, to test the write-read desk operation it would consist of the following four states:

1. SEEK to the desired cylinder. Check drive status when done.
2. WRITE a test pattern to disk sector. Check status when done.
3. READ that sector into the program buffer. Check status when done.
4. COMPARE read-back data with test pattern.

If, for instance the result of stage 4 shows compare error, the user may command the program to loop back around stages 3 and 4 while trying to track down the source of error, or re-start from stage 1.

2.4 Error Display

When the program detects an error in a test, it will display up to 7 lines information describing the process and occurrence. The display includes details about what item is being tested, reference to program listing, input data or signal used, output result and the normal criterion. More detail description of each line is given below in the order of appearance.

2.4.1 REFERENCE

Reference is a 6 digit octal number pointing to the location in program listing where the error was found. It is provided as an aid to users who wish to consult the listing.

2.4.2 TEST STEP

The name of the test step which is currently under test. The name will appear only once if more than one error was found under the same step.

2.4.3 TEST STAGE

The name of test stage within a test step where the failure was occurred.

2.4.4 DRIVE NUMBER

The drive number is the number of the controller port where the drive is connected to, not the number plug in the drive. The drive number will not appear where it is not applicable, such as in most logic tests.

2.4.5 INPUT

The input is the parameter(s) used for stimulating a test in a test stage. There may be more than one parameter and it may be a data type, control signal or command.

2.4.6 OUTPUT

The output is the test result of what has occurred from the above input. The result may be actual data or a statement of true-false.

2.4.7 NORMAL

Normal is what was expected for a test result under an ideal condition. That is, error free. Compare NORMAL with OUTPUT to see the difference.

2.5 Test Software

This section discusses the software aspect of a typical test routine. All test routines are organized in the same format. The only difference is the length of test.

2.5.1 Modules

The test routines are divided into three modules according the test mode as follows:

Test Mode	Module Name	Starting Location
Logic + Tester	DDLG	33ØØ
Drive	DDRV	64ØØ
Tester	DDTS	

Starting location is the address of the very first test in that module. The names of test steps in each module are listed in sections 2.2.1, 2.2.2 and 2.2.3

2.5.2 Program Listing

Each test step always begins at top of page in the listing like a chapter in a book. It starts with the name of test step followed by a short description about the purpose of the step. Comments and explanation are interspersed between test stages so a reader can follow the logic and have a better understanding of the test.

2.5.3 Structure

All test routines are identically structured regardless of their length. A step consists of three parts: Step initialization, test stages and error message blocks.

2.5.3.1 Step Initialization

Certain parameters of a test step must be initialized prior to entering the test stages. This is done by the very first instruction of a step; a subroutine with 4 calling arguments as follows:

```
JSR   @SET
Name of test step            TSTX
Test step start vector       TSTV
Next test step vector        NTST
Print drive number flag      DRVN
```

The 4 arguments are copied into the respective locations as pointed by the arrows so they can be accessed by the error routine when needed:

TSTX - A pointer to the ASCII text of name of test step.

TSTV - Starting address of the first test stage. When SET is all done, it exit by JMP @ TSTV to start test.

NTST - A pointer to the start of next test step. It is used by the "S" command and at the end of current test by JMP @ NTST to begin the next step.

DRVN - A flage to the error routine whether to print drive number or not. Print when DRVN = Ø. Else don't.

2.5.3.2 Test Stages

Test stages are the main body of a test step. The number of stages varies from 2 on up to 12. A test stage is defined as one of a series of actions taken to produce the desired result (See Section 2.3).

In other words, it is the input data or command to a controller circuit to do something. When done, the result is checked to see if it is correct. Proceed to the next stage if so, else call the error routine. At the end of the last stage, end of a test step, the exit is made by JMP @ NTST into the next test. See Figure 2.

2.5.3.3 Error Message Blocks

The error message blocks appear at the end of the "chapter" in the listing. Usually there are as many blocks as test stages.

An error message block contains 6 words. Each block is associated with certain test stages. A block is referenced by the error routine as follows:

```
JSR     @ERROR
Name of error message block
```

From the error message block, combining with those initialized by SET, the ERROR routine displays up to 7 lines of information on the console. The block format is fixed. This is, each word must appear in the correct sequence as shown below:

```
Block Name:  Address of routine which found the error
             Loop back address for "L" command
             Pointer to text of test stage
             Test input:  Number=Print contents of input registers
                index @ XXXX=Print statement text
             Test output: Number=Print contents of output registers
                index @ XXXX=Print statement text
             Normal:  Number=print data,@XXXX=statement text
```

Following the error display is the command prompt, awaiting user's command. Therefore, the exit from ERROR routine is depending upon the command, see Section 2.5 and Figure 2.

3.0 Procedure

3.1 Program Restart

The program is self-starting when first loaded into memory. Restart is from location 2.

3.2 Data Entry and Display

All data entries to the program are terminated with CR. Any entry error is erasable with either the backspace or rubout key before hitting CR. Continuous erasing will delete entire entry. Any unacceptable entry will be rejected with a "?" response. For assistance type "H". The display of data and status values are in octal. Only the test pass count is shown in decimal.

3.3 Device Code Entry

A controller of device code other than 27 must be entered. Else, defaults (CR only) to 27.

3.4 Test Mode Entry

Set test mode by entering the name of the program under which the controller to be tested: LOGIC, DRIVE, OR TESTER. After exiting test from a mode and to re-enter the same mode, just hit CR to proceed. The program retains the original input. To change test mode, key in the new mode name.

3.5 Stop On Error Entry

Set the option to stop the program from executing the next test or to proceed when detecting an error. Enter "Y" for yes to stop or "N" for no stopping. The option is reversible by entering "C" command when the command prompt ( ) appears.

3.6 Logić Test

After the stop option entry and before starting the logic test, the program checks to see if any disk drive is on-line. If so, a message will appear to ask the user to set it off-line so it will not interfere with testing. Test will begin as soon as the drive goes off-line or no drive was connected to the controller. Once started, the test sequence repeats in an endless loop until manually stopped by hitting any key. Then a count of total passes is shown and the program returns to initialization mode.

3.7 Drive Test

3.7.1 Drive Name Entry

If the drive test mode was selected, then following the stop option input is the drive type input. Enter either the drive code number or mnemonic (CR if same as set previously). If neither is known, type "H" to list the crive directory for consultation. The directory contains the following information in one line for each drive, by column:

1. Drive code number
2. Drive nmemonic
3. Manufacturers name
4. Model or type
5. Storage capacity in MB
6. Number of cylinders
7. Number of heads
8. Number of sectors track

The directory display may be temporarily stopped and resume by hitting any key except CR, which is a quit display command.

3.7.2 Drive Number Entry

The number of the controller port where the drive cable is connected is entered. This number must also match the number plug in the drive. Default is drive Ø if it has not been set previously, or remains as previously set. The drive test will begin after checking for drive on-line. An action message will appear if drive is off-line. Then hit any key to start when ready.

3.7.3 Test Progress

The drive test requires minutes seconds to sequence through all steps. The sequence is repeated continuously until stopped by the user. At the end of each sequence (pass), the pass number is shown. If there were no errors at all, only the pass numbers in ascending order are shown. To exit from test, strike any key. If it happens to be doing a lengthy test during a key stroke, the user should wait until it finished its current test to let it quit.

3.8 Test Control Commands

Test control commands are used to set next course of action after an error display. A command is entered whenever the command prompt, a greater than sign ( ), appears. Type "H" to list the commands and their functions on console. The detailed description of each command is given below.

3.8.1 Loop on Error - Command "L"

Loop on the erring test stage as shown in the error display. Error displays are suppressed. Strike any key except "X" to terminate looping, the command prompt will re-appear afterwards.

3.8.2 Proceed from Error - Command "P"

Proceed test onto the next immediate stage or step.

3.8.3 Repéat Test - Command "R"

Re-start current test step once from the beginning.

3.8.4 Skip Test Step - Command "S"

Skip over the entire test step onto the next step in sequence.

3.8.5 Change Stop on Error - Command "C"

Change the stop on error option to the opposite sense. E.G. if stop was set previously, it will be set to no stop, vice versa. The setting will be shown on the console.

3.8.6 Abort Test - Command "X"

Quit current operation and restart program.

3.8.7 Dump Write Buffer - Command "DWB"

Dump contents of write buffer (WBUFF) in the program out to the console. Dumping is shown in 9 columns with the first column showing the buffer starting address of the following 8 data words in the same row. Hit any key to suspend dumping temporarily or resume dumping. Hit CR to exit from dump.

3.8.8 Dump Read Buffer - Command "DRB"

Dump contents of read buffer (RBUFF) in the program out to the console. The display and dump commands are same as in DWB.

3.8.9 Decode DOA Word - Command "DDOA"

Decode the most recent DOA word and display each active field of the work in interpretive English. The display may include clear RW and drive attention, controller command and volume select fields.

3.8.10 Decode DDC Word - Command "DDOC"

Decode the most recent DOC word and display the addressed surface and sector numbers and sector count. The display legends H for head (surface), S for sector and SC for sector count (negated).

3.8.11 Decode DIA Word - Command "DDIA"

Decode the most recent DIA controller status word and display the meaning of each active (True) bit in interpretive English.

3.8.12 Decode DIB Word - Command "DDIB"

Decode the most recent DIB drive status word and display the meaning of each active bit in interpretive English.

3.8.13 Jump to Location X - Command "JX"

Jumps to memory location X. X is in octal. This command must be used with extreme caution. Jumping to the wrong address may destroy the program. Its main purpose is to go to debug from the program if debug is also loaded elsewhere in memory.