

RABO

UDA & DISK DRV DIAG
CZUDCBO

AH-S831B-MC
FICHE 1 OF 2

SEP 1982
COPYRIGHT © 81-82
MADE IN USA



The main body of the document is a large grid of technical data. Each cell in the grid contains a small, dense block of text, likely representing a specific diagnostic procedure or component specification. The text is too small to be legible in this scan, but the layout is organized into a structured grid format.

RABO

UDA & DISK DRV DIAG
CZUDCBO

AH-S831B-MC
FICHE 2 OF 2

SEP 1982
COPYRIGHT © 81-82
MADE IN USA



The main body of the document is a large grid of technical data. Each cell in the grid contains a small, dense block of text, likely representing a specific diagnostic test result or configuration parameter. The text is too small to be legible in this scan, but the layout is organized into a regular grid pattern.

.REM *

IDENTIFICATION

PRODUCT CODE: AC-S830B-MC
PRODUCT NAME: CZUDCBO UDA50 & DISK DRV DIAG
PRODUCT DATE: 14-APR-82
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: DALE KECK

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1981, 1982 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

.REM *

TABLE OF CONTENTS

	Page	
1.0	GENERAL INFORMATION	3
1.1	PROGRAM ABSTRACT	3
1.2	SYSTEM REQUIREMENTS	4
2.0	OPERATING INSTRUCTIONS	4
2.1	COMMANDS	4
2.2	SWITCHES	5
2.3	FLAGS	6
2.4	HARDWARE QUESTIONS	6
2.5	SOFTWARE QUESTIONS	8
2.6	EXTENDED P-TABLE DIALOGUE	10
2.7	QUICK STARTUP PROCEDURE	12
3.0	ERROR INFORMATION	15
3.1	TYPES OF ERROR MESSAGES	15
3.2	SPECIFIC ERROR MESSAGES	17
3.2.1	HOST PROGRAM ERROR MESSAGES (00001 TO 00999)	17
3.2.2	TEST 1 ERROR MESSAGES (01000 TO 01999)	26
3.2.3	TEST 2 INFORMATIONAL MESSAGES	29
3.2.4	TEST 2 ERROR MESSAGES (02000 TO 02999)	30
3.2.5	TEST 3 INFORMATIONAL MESSAGES	40
3.2.6	TEST 3 ERROR MESSAGES (03000 TO 03999)	41
3.2.7	TEST 4 INFORMATIONAL MESSAGES	51
3.2.8	TEST 4 ERROR MESSAGES (04000 TO 04999)	52
3.2.9	SPECIAL DEVICE FATAL (05000)	76
3.3	TEST 4 RETRY/RECOVERY METHODS	78
3.4	DEC STANDARD 166 EXCERPTS	91
3.4.1	THE REPLACEMENT AND CACHING TABLES	91
3.4.2	FCT STRUCTURE	93
4.0	PERFORMANCE AND PROGRESS REPORTS	96
5.0	TEST SUMMARIES	98
5.1	TEST # 1 - UNIBUS ADDRESSING TEST	98
5.2	TEST # 2 - DISK RESIDENT DIAGNOSTIC TEST	100
5.3	TEST # 3 - DISK FUNCTION TEST	102
5.4	TEST # 4 - DISK EXERCISER	103

1.0 GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

This is the only diagnostic program provided for testing the UDA-50 Unibus Disk Controller and the disk drives connected to it. There are four tests within this diagnostic:

- Test # 1 - Unibus Addressing Test. Runs the UDA-50 ROM resident diagnostics, then further tests the Unibus address interface.
- Test # 2 - Disk Resident Diagnostic Test. Executes the diagnostics in each disk drive.
- Test # 3 - Disk Function Test. Functionally tests each disk drive to ensure the disk can seek, read, write and format.
- Test # 4 - Disk Exerciser. Exercises the disk drives in a manner similar to normal operating systems. This test should be used to gain confidence in the reliability of the disk drive.

This program is designed to handle all future disk drives that are attached to the UDA-50 without modifying or rereleasing. This is possible because the disk drives are programmed to tell this diagnostic about all their characteristics that make them different from other drives, such as number of cylinders, sectors per cylinder, etc.

Two other PDP-11 diagnostic programs are provided for the UDA-50 disk subsystem:

CZUDEBO - UDA-50 Disk Drive Formatter.

DEC/X11 - Unibus Exerciser can be run on the UDA-50 using the UDA-50 module DUBBO.

This diagnostic has been written for use with the Diagnostic Runtime Services Software (Supervisor). These services provide the interface to the operator and to the software environment. For a complete description of the Runtime Services, refer to the XXDP+ User's Manual. There is a brief description of the Runtime Services in section 2 of this document.

1.2 SYSTEM REQUIREMENTS

This program was designed using the PDP-11 Diagnostic Runtime Services revision C. Run time environments are determined by the Runtime Services and may change as new versions of the Services are developed. This program requires the following:

- PDP-11 Unibus processor
- 28K words of memory (minimum)
- Console terminal
- XXDP+ load media containing this program and the ZUDDBO.PAK data file
- One or more UDA-50 subsystems
- Line clock - either Type L or P

The line clock is used for all timed loops in the program. The diagnostic will run on a system with no clock but will hang whenever an event for which the program is waiting does not happen (i.e., a time-out error message will not result).

This diagnostic program requires that the data file ZUDDBO.PAK be on the XXDP+ system device. This data file is ordered under the name CZUDDBO.

2.0 OPERATING INSTRUCTIONS

This section contains a brief description of the Runtime Services. For detailed information, refer to the XXDP+ User's Manual (CHQUS).

2.1 COMMANDS

There are eleven legal commands for the Diagnostic Runtime Services (Supervisor). This section lists the commands and gives a very brief description of them. The XXDP+ User's Manual has more details.

COMMAND	EFFECT
START	Start the diagnostic from an initial state
RESTART	Start the diagnostic without initializing
CONTINUE	Continue at test that was interrupted (after ^C)
PROCEED	Continue from an error halt
EXIT	Return to XXDP+ Monitor (XXDP+ OPERATION ONLY!)
ADD	Activate a unit for testing (all units are considered to be active at start time)
DROP	Deactivate a unit
PRINT	Print statistical information (see section 4.0)
DISPLAY	Type a list of all device information
FLAGS	Type the state of all flags (see section 2.3)
ZFLAGS	Clear all flags (see section 2.3)

A command can be recognized by the first three characters. So you may, for example, type "STA" instead of "START".

2.2 SWITCHES

There are several switches which are used to modify supervisor operation. These switches are appended to the legal commands. All of the legal switches are tabulated below with a brief description of each. In the descriptions below, a decimal number is designated by "DDDD".

SWITCH -----	EFFECT -----
/TESTS:LIST	Execute only those tests specified in the list. List is a string of test numbers, for example - /TESTS:1:5:7-10. This list will cause tests 1,5,7,8,9,10 to be run. All other tests will not be run.
/PASS:DDDD	Execute DDDDD passes (DDDD = 1 to 64000)
/FLAGS:FLGS	Set specified flags. Flags are described in section 2.3.
/EOP:DDDD	Report end of pass message after every DDDDD passes only. (DDDD = 1 to 64000)
/UNITS:LIST	TEST/ADD/DROP only those units specified in the list. List example - /UNITS:0:5:10-12 use units 0,5,10,11,12 (unit numbers = 0-63).

Example of switch usage:

START/TESTS:1-5/PASS:1000/EOP:100

The effect of this command will be: 1) tests 1 through 5 will be executed, 2) all units will tested 1000 times and 3) the end of pass messages will be printed after each 100 passes only. A switch can be recognized by the first three characters. You may, for example, type "/TES:1-5" instead of "/TESTS:1-5".

Below is a table that specifies which switches can be used by each command.

	TESTS	PASS	FLAGS	EOP	UNITS
-----	-----	-----	-----	-----	-----
START	X	X	X	X	X
RESTART	X	X	X	X	X
CONTINUE		X	X	X	
PROCEED			X		
DROP					X
ADD					X
PRINT					
DISPLAY					X
FLAGS					
ZFLAGS					
EXIT					

2.3 FLAGS

Flags are used to set up certain operational parameters such as looping on error. All flags are cleared at startup and remain cleared until explicitly set using the flags switch. Flags are also cleared after a START or RESTART command unless set using the flag switch. The ZFLAGS command may also be used to clear all flags. With the exception of the START, the RESTART and ZFLAGS commands, no commands affect the state of the flags; they remain set or cleared as specified by the last flag switch.

FLAG	EFFECT
HOE	Halt on error - control is returned to runtime services command mode
LOE	Loop on error
IER*	Inhibit all error reports
IBE*	Inhibit all error reports except first level (first level contains error type, number, PC, test and unit)
IXE*	Inhibit extended error reports (those called by PRINTX macro's)
PRI	Direct messages to line printer
PNT	Print test number as test executes
BOE	'BELL' on error
UAM	Unattended mode (no manual intervention)
ISR	Inhibit statistical reports
IDU	Inhibit program dropping of units
LOT	Loop on test

*Error messages are described in section 3.1

See the XXDP+ User's Manual for more details on flags. You may specify more than one flag with the FLAG switch. For example, to cause the program to loop on error, inhibit error reports and type a 'BELL' on error, you may use the following string:

```
/FLAGS:LOE:IER:BOE
```

2.4 HARDWARE QUESTIONS

When a diagnostic is STARTed, the Runtime Services will prompt the user for hardware information by typing "CHANGE HW (L) ?". When you answer this question with a "Y", the Runtime Services will ask for the number of units (in decimal). You will then be asked the following questions for each unit. When you answer this question with an "N", the Runtime Services will use the answers built into the program by the SETUP utility (see chapter 6 of the XXDP+ User's Manual). If you have never run the SETUP utility on this program file, the default values listed below (just before the question mark) will be used.

UNIBUS ADDRESS OF UDA (O) 172150 ?

Answer with the address of the UDAIP register of one UDA as addressed by the processor with memory management turned off (i.e., an even 16-bit address in the range of 160000 to 177774).

VECTOR (O) 154 ?

Answer with the interrupt vector address of the UDA. A vector address in the range of 4 to 774 may be specified. The UDA does not have a vector "hard wired" to it, so any vector not being used by this program and XXDP+ may be used.

BR LEVEL (D) 5 ?

Answer with the interrupt priority used by the UDA. Levels 4 to 7 are accepted. This level must match the level "hard wired" in the UDA by the priority plug.

UNIBUS BURST RATE (D) 63 ?

The UDA allows the ability to control the maximum number of words transferred across the UNIBUS each time the UDA becomes master. The default answer of 63 will allow for the fastest execution of this diagnostic program. You may answer with the value your operating system uses or use zero which will tell the UDA to supply a value that should work on any system. A decimal number in the range of 0 to 63 may be specified and all values should work on any system. A larger value will allow for a faster running program. The value will be passed directly to the UDA during initialization.

DRIVE NUMBER (D) 0 ?

Answer with the drive number of the drive you wish to test. This is the number which appears on the "unit plug" on the front of the disk drive. On a multi-unit drive, each sub-unit number on the drive must be tested as a separate unit to completely test the drive. A maximum of eight logical drives may be tested on one UDA at a time (UDA configuration limit).

EXERCISE ON CUSTOMER DATA AREA IN TEST 4 (L) N ?

Answer "N" to have test 4 (drive exerciser) run on the diagnostic area of the disk. Answer "Y" to run on the customer data area. A "Y" answer will destroy any customer data that may be on the disk. A warning message will be printed before testing begins if this question is answered "Y".

CUSTOMER DATA WILL BE DESTROYED ON:

UNIT	UDA AT	DRIVE
xx	xxxxxx	xxx

Unless the diagnostic is being run in unattended mode (i.e., START/FLAG:UAM command), a confirmation will also be required as follows:

ARE YOU SURE CUSTOMER DATA CAN BE DESTROYED (L) ?

If the above question is answered 'N', the entire diagnostic will stop and the Runtime Services prompt will be displayed. No default answer is provided for this question.

2.5 SOFTWARE QUESTIONS

After you have answered the hardware questions or after a RESTART or CONTINUE command, the Runtime Services will ask for software parameters. You will be prompted by "CHANGE SW (L) ?" If you wish to change any parameters, answer by typing 'Y'. The software questions and the default values are described in the next paragraphs.

ENTER MANUAL INTERVENTION MODE FOR SPECIAL DIAGNOSIS (L) N ?

Tests 2 and 4 have manual intervention modes which allow additional parameters to be input to alter the normal testing of a disk drive. This question should normally be answered 'N' when this diagnostic is first run. Then, depending on the errors detected, it may be desirable to change this answer to 'Y' and alter the testing to further isolate the problem. If this question is answered 'Y', and the UAM (unattended mode operation) flag is set, tests 2 and 4 will print a warning message that the mode cannot be entered and will proceed as if answered 'N'. See the description of the individual tests in section 5 for more information.

REMAINING SOFTWARE QUESTIONS APPLY TO TEST 4 ONLY

This informational message is printed to describe the use of the remaining questions. If test 4 is not being run, a 'CONTROL Z' can be typed to bypass them.

ERROR LIMIT (D) 32 ?

Enter the number of hard errors allowed before a drive is dropped from exercise by test #4. A number in the range of 1 to 5535 will be accepted.

READ TRANSFER LIMIT IN MEGABYTES - 0 FOR NO LIMIT (D) 0 ?

When the specified number of bytes have been read from a drive by test #4, the drive will be dropped from testing. When all drives are dropped, an end of pass will be indicated and the selected tests will be run again. This is the method used to determine how long test #4 is to run. Answer with a zero to prevent test from ending. The only other way test #4 can end is to have all drives dropped because the error limit on each is exceeded. Of course, the operator can always stop test #4 by typing a control-C.

SUPPRESS PRINTING SOFT ERRORS (L) Y?

When test #4 needs to perform retries, soft error reports will be printed to give as much information as possible. These actions are considered normal operation and are not error conditions until the retries fail. When the test is being run only to see how reliable the drive performs, this question should be answered 'Y' so they are not confused with hard errors. The number of these soft errors is always reported in the statistical report. Answer 'N' to see all the soft error reports.

DO INITIAL WRITE ON START (L) Y ?

If test #4 is to do data compares, the drive will need to be written with data patterns readable by the program.

If the diagnostic area is selected for testing, the initial write is always performed (regardless of how this question is answered).

If the customer data area is selected for testing, the initial write will be performed when all of the following are true:

1. This question is answered 'Y'.
2. This is the first time test #4 is being run after a START command.
3. The disk is write enabled.

Answering this question 'N' when testing on the customer data area will normally result in data comparison errors if the disk was not previously written by this diagnostic or the formatter.

Note that write checks are not performed during the initial write.

ENABLE ERROR LOG (L) N ?

A 'Y' answer will cause error messages in test #4 to be stored in a log buffer. Once the log buffer is full, additional error information is lost. The contents of the log buffer will be printed when test #4 is stopped and a statistical report requested. This log feature is intended to allow the Digital Diagnosis Center (DDC) to start test #4 then hang up from the system and let it run for some period of time. DDC can call the system back later, type control-C, then PRINT and see the errors that have occurred (up to the limit of the log buffer). A message will be printed to indicate no errors have occurred if the log buffer is empty. Test #4 will not be allowed to end while the error log is enabled until the error log is printed. The log buffer will hold 30 error messages when one disk unit is being tested. The log buffer will decrease in size as more units are tested.

2.6 EXTENDED P-TABLE DIALOGUE

When you answer the hardware questions, you are building entries in a table that describes the devices under test. The simplest way to build this table is to answer all questions for each unit to be tested. If you have a multiplexed device such as a mass storage controller with several drives or a communication device with several lines, this becomes tedious since most of the answers are repetitious.

To illustrate a more efficient method, suppose you are testing a fictional device, the XY11. Suppose this device consists of a control module with eight units (sub-devices) attached to it. These units are described by the octal numbers 0 through 7. There is one hardware parameter that can vary among units called the Q-factor. This Q-factor may be 0 or 1. Below is a simple way to build a table for one XY11 with eight units.

```
# UNITS (D) ? 8<CR>

UNIT 1
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 0<CR>
Q-FACTOR (O) 0 ? 1<CR>

UNIT 2
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 1<CR>
Q-FACTOR (O) 1 ? 0<CR>

UNIT 3
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 2<CR>
Q-FACTOR (O) 0 ? <CR>

UNIT 4
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 3<CR>
Q-FACTOR (O) 0 ? <CR>

UNIT 5
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 4<CR>
Q-FACTOR (O) 0 ? <CR>

UNIT 6
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 5<CR>
Q-FACTOR (O) 0 ? <CR>

UNIT 7
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 6<CR>
Q-FACTOR (O) 0 ? 1<CR>
```



```
UNIT 8  
CSR ADDRESS (0) 160000<CR>  
SUB-DEVICE # (0) ? 7<CR>  
Q-FACTOR (0) 1 ? <CR>
```

Notice that the default value for the Q-factor changes when a non-default response is given. Be careful when specifying multiple units!

As you can see from the above example, the hardware parameters do not vary significantly from unit to unit. The procedure shown is not very efficient.

The Runtime Services can take multiple unit specifications however. Let's build the same table using the multiple specification feature.

```
# UNITS (0) ? 8<CR>
```

```
UNIT 1  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 0,1<CR>  
Q-FACTOR (0) 0 ? 1,0<CR>
```

```
UNIT 3  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 2-5<CR>  
Q-FACTOR (0) 0 ? 0<CR>
```

```
UNIT 7  
CSR ADDRESS (0) ? 160000<CR>  
SUB-DEVICE # (0) ? 6,7<CR>  
Q-FACTOR (0) 0 ? 1<CR>
```

As you can see in the above dialogue, the runtime services will build as many entries as it can with the information given in any one pass through the questions. In the first pass, two entries are built since two sub-devices and q-factors were specified. The Services assume that the CSR address is 160000 for both since it was specified only once. In the second pass, four entries were built. This is because four sub-devices were specified. The "-" construct tells the Runtime Services to increment the data from the first number to the second. In this case, sub-devices 2, 3, 4 and 5 were specified. (If the sub-device were specified by addresses, the increment would be by 2 since addresses must be on an even boundary.) The CSR addresses and Q-factors for the four entries are assumed to be 160000 and 0 respectively since they were only specified once. The last two units are specified in the third pass.

The whole process could have been accomplished in one pass as shown below.

```
# UNITS (D) ? 8<CR>
UNIT 1
CSR ADDRESS (O) ? 160000<CR>
SUB-DEVICE # (O) ? 0-7<CR>
Q-FACTOR (O) 0 ? 0,1,0,,,,1,1<CR>
```

As you can see from this example, null replies (commas enclosing a null field) tell the Runtime Services to repeat the last reply.

2.7 QUICK START-UP PROCEDURE

To start-up this program:

1. Boot XXDP+
2. Give the date and answer the LSI and 50HZ (if there is a clock) questions
3. Type 'R ZUDCBO'
4. Type 'START'
5. Answer the 'CHANGE HW' question with 'Y'
6. Answer all the hardware questions
7. Answer the 'CHANGE SW' question with 'N'

When you follow this procedure you will be using only the defaults for flags and software parameters. These defaults are described in sections 2.3 and 2.5.

Sample of terminal dialogue to test two disks on one UDA-50:

DR>STA/FLA:PNT

CHANGE HW (L) ? Y

UNITS (D) ? 2

UNIT 0

UNIBUS ADDRESS OF UDA (O) 172150 ?

VECTOR (O) 154 ?

BR LEVEL (D) 5 ?

UNIBUS BURST RATE (D) ?

DRIVE NUMBER (D) 0,1

EXERCISE ON CUSTOMER DATA AREA IN TEST 4 (L) N ?

CHANGE SW (L) ? N

TST: 001

TESTING INTERRUPT ABILITY OF UDA AT ADR 172150 VEC 154...COMPLETED

TST: 002

TST: 003

TST: 004

UNIT 0 UDA AT 172150 DRIVE 0 RUNTIME 0:02:43
INITIAL WRITE COMPLETE

UNIT 1 UDA AT 172150 DRIVE 1 RUNTIME 0:05:31
INITIAL WRITE COMPLETE

TEST 4 IN PROGRESS. RUNTIME 0:15:00

UNIT	DRIVE	SERIAL-NUMBER	SEEKS X1000	MBYTES READ	MBYTES WRITTEN	HARD ERRORS	SOFT ERRORS	ECC
0	0	0	3	9	6	0	0	0
1	1	1	3	8	6	0	0	0

Sample of terminal dialogue going through software questions to specify transfer limit (one disk being tested).

DR>STA/FLA:PNT

CHANGE HW (L) ? N

CHANGE SW (L) ? Y

ENTER MANUAL INTERVENTION MODE FOR SPECIAL DIAGNOSIS (L) N ?

REMAINING SOFTWARE QUESTIONS APPLY TO TEST 4 ONLY

ERROR LIMIT (D) 32 ?

READ TRANSFER LIMIT IN MEGABYTES - 0 FOR NO LIMIT (D) 0 ? 5

SUPPRESS PRINTING SOFT ERRORS (L) Y ?

DO INITIAL WRITE ON START (L) Y ?

ENABLE ERROR LOG (L) N ?

TST: 001

TESTING INTERRUPT ABILITY OF UDA AT ADR 172150 VEC 154...COMPLETED

TST: 002

TST: 003

TST: 004

UNIT 0 UDA AT 172150 DRIVE 0 RUNTIME 0:02:43

INITIAL WRITE COMPLETE

UNIT 0 UDA AT 172150 DRIVE 0 RUNTIME 0:09:41

REACHED TRANSFER LIMIT - TESTING STOPPED

TEST 4 IN PROGRESS. RUNTIME 0:09:41

UNIT	DRIVE	SERIAL-NUMBER	SEEKS X1000	MBYTES READ	MBYTES WRITTEN	HARD ERRORS	SOFT ERRORS	ECC
0	0	0	2	5	4	0	0	0

CZUDCB EOP 1

0 CUMULATIVE ERRORS

TST: 001

TESTING INTERRUPT ABILITY OF UDA AT ADR 172150 VEC 154...COMPLETED

TST: 002

.

.

.

3.0 ERROR INFORMATION

3.1 TYPES OF ERROR MESSAGES

There are three levels of error messages that may be issued by a diagnostic: general, basic and extended. General error messages are always printed unless the "IER" flag is set (section 2.3). The general error message is of the form:

```
NAME TYPE NUMBER ON UNIT NUMBER TST NUMBER PC:XXXXXX  
error message
```

where: NAME = diagnostic name
TYPE = error type (SYS FTL ERR, DEV FTL ERR, HRD ERR or SFT ERR)
NUMBER = error number
UNIT NUMBER = 0 - N (N is last unit in PTABLE)
TST NUMBER = test and subtest where error occurred
PC:XXXXXX = address of error message call

System fatal errors (SYS FTL ERR) are used to report errors that are fatal to the entire diagnostic program. The diagnostic stops and the Runtime Services prompt is printed.

Device fatal errors (DVC FTL ERR) are used to report errors that are fatal to the device (may be either a UDA-50 or disk drive). Testing stops on that device for the remainder of the current test.

Hard errors (HRD ERR) reports most of the errors detected. Testing will normally continue after the printing of the error.

Soft errors (SFT ERR) are used only in test 4. They present information about an error for which recovery will be attempted. These are printed only if the SUPPRESS PRINTING SOFT ERRORS software question is answered "N" and are used only to provide a greater detail of information. During the error recovery attempt, several soft errors may be printed. Unless the soft errors are followed by a hard error message, the error condition was corrected and testing proceeds.

Basic error messages are messages that contain some additional information about the error. These are always printed unless the "IER" or "IBE" flags are set (section 2.3). These messages are printed after the associated general message.

Extended error messages contain supplementary error information such as register contents or good/bad data. These are always printed unless the "IER", "IBE" or "IXE" flags are set (section 2.3). These messages are printed after the associated general error message and any associated basic error messages.

The general and basic error messages from this diagnostic are always one line each. The basic message defines what program detected the error, the drive being tested and the time of the error.

The PDP-11 program that is loaded into memory when you give the 'R ZUDCBO' command to the XXDP+ monitor is only a small part of this diagnostic. A data file called ZUDDBO.PAK on the system load device (the same device from which the 'R' command read the PDP-11 program) contains four programs which are read from the file and loaded into the UDA-50 for execution. These programs are called "diagnostic machine" or DM programs. The "diagnostic machine" is the facility in the UDA-50 which executes a PDP-11 like program. The large majority of the testing is done by these four "diagnostic machine" programs. Once the PDP-11 program has loaded and started the "diagnostic machine" program, all it does is respond to requests from that program. These requests include such things as telling the "diagnostic machine" which disks on that UDA-50 are to be tested, printing an error message and updating statistics which are printed in the statistical report (see section 4.0).

The basic message (the second line of every error message) will be one of the following:

HOST PROGRAM UDA AT xxxxxx RUNTIME hhh:mm:ss

The host program (PDP-11) detected the error. UDA AT xxxxxx identifies the address of the UDA-50 being tested. It may be omitted if the error is not specific to one UDA-50.

UNIBUS ADDRESSING DM PC:xxxx UDA AT xxxxxx RUNTIME hhh:mm:ss

The "diagnostic machine" program loaded in test 1 detected the error. DM PC xxxx identifies the address in the "diagnostic machine" program where the error message is reported.

DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss

The "diagnostic machine" program loaded in test 2 detected the error. DM PC xxxx identifies the address in the "diagnostic machine" program where the error message is reported. DRIVE xxx identifies the drive number.

DISK FUNCTIONAL DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss

The "diagnostic machine" program loaded in test 3 detected the error.

DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss

The "diagnostic machine" program loaded in test 4 detected the error.

Sample error message:

CZUDC DVC FTL ERR 00021 ON UNIT 00 TST 001 SUB 003 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME 0:00:12
UDA RESIDENT DIAGNOSTICS DETECTED FAILURE
UDASA CONTAINS 104041
REPLACE UDA MODULE M7161

- general message
- basic message
 \ - extended message
 /

Informational messages are also printed by this program. They are usually one or two lines in length. They are printed as extended messages and are always printed unless the "IER", "IBE" or "IXE" flags are set.

Sample informational message:

```
UNIT 0 UDA AT 172150 DRIVE 0 RUNTIME 0:02:43  
INITIAL WRITE COMPLETE
```

3.2 SPECIFIC ERROR MESSAGES

Following is a list of the error messages that may be printed by the diagnostic program. In the list, some of the numbers that may vary with execution or program version are shown as "xxx". These include program counters and runtime. Other numbers, such as unit number, drive number, UDA-50 address and data in registers are filled with sample numbers. Additional information about the error may follow the error message.

3.2.1 HOST PROGRAM ERROR MESSAGES (00001 to 00999)

```
00001 CZUDC SYS FTL ERR 00001 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx  
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx  
I DON'T LIKE THE ANSWERS YOU GAVE TO THE HARDWARE QUESTIONS  
UDA HAS MORE THAN ONE VECTOR, BR LEVEL OR BURST RATE
```

When the hardware questions were answered, two units were selected with the same UNIBUS address but with a different vector, BR level or burst rate. A single UDA-50 can have only one vector, BR level or burst rate. The program is aborted and returns to the Runtime Services prompt so that you can change the hardware questions.

```
00002 CZUDC SYS FTL ERR 00002 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx  
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx  
I DON'T LIKE THE ANSWERS YOU GAVE TO THE HARDWARE QUESTIONS  
TWO UNITS SELECT THE SAME DRIVE
```

The hardware questions for two units were exactly the same. The program is aborted and returns to the Runtime Services prompt so that you can change the hardware questions.

00003 CZUDC SYS FTL ERR 00003 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
I DON'T LIKE THE ANSWERS YOU GAVE TO THE HARDWARE QUESTIONS
MORE THAN EIGHT DRIVES SELECTED ON THIS UDA

Up to four physical disk drives can be attached to a UDA-50 at one time. A physical disk drive may be from one to four logical disk drives. Each logical disk drive is considered one unit to the diagnostic program. Even though more than eight logical disk drives can be attached to one UDA-50, the UDA-50 only supports eight. The program is aborted and returns to the Runtime Services prompt so that you can change the hardware questions.

00004 CZUDC SYS FTL ERR 00004 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM RUNTIME x:xx:xx
NOT ENOUGH ROOM IN MEMORY TO TEST THE UNITS SELECTED
PLEASE START PROGRAM OVER AND TEST FEWER UNITS AT A TIME

This program does not limit the number of units that can be tested by specifying a maximum number. What limits the number is the amount of memory used to store data on each unit. You have exceeded the number of units that are testable at one time. Start program over and select fewer units.

00005 CZUDC SYS FTL ERR 00005 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM RUNTIME x:xx:xx
CHECKSUM ERROR IN DM PROGRAM FILE

As a DM program is read from the load media, a checksum is calculated. If the checksum contained in the file does not match what is calculated, an error reading the data file is declared. Restore the data file ZUDB0.PAK to your load media.

00006 CZUDC SYS FTL ERR 00006 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM RUNTIME x:xx:xx
TABLE INCONSISTANCY ERROR. PLEASE RE-LOAD PROGRAM

When the host program is started, controller tables are set according to the P-tables. Error 00006 will occur if the tables were corrupted after restarting the diagnostic. Load and start your program again.

00007 CZUDC SYS FTL ERR 00007 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM RUNTIME x:xx:xx
ERROR IN DM PROGRAM FILE. DM PROGRAM NOT FOUND

The host program was not able to read the DM program from the load media properly. Restore the data file ZUDB0.PAK to your load media.

00008 CZUDC SYS FTL ERR 00008 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
I DON'T LIKE THE ANSWERS YOU GAVE TO THE HARDWARE QUESTIONS
TWO UDA'S USE THE SAME VECTOR

The hardware questions for two units specified different UDA-50 Unibus addresses but identical vector addresses. The program is aborted and returns to the Runtime Services prompt so tha you can change the hardware questions.

00020 CZUDC DVC FTL ERR 00020 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
MEMORY ERROR TRYING TO READ UDA REGISTERS
CHECK UNIBUS SELECTION SWITCHES ON UDA MODULE M7161
OR UNIBUS
OR REPLACE UDA MODULE M7161

A non-existent memory error occurred when the host program tried to access the UDAIP and UDASA registers. The UDA is at another address (check the UNIBUS selection switches) or module M7161 is broken or the UNIBUS is broken.

00021 CZUDC DVC FTL ERR 00021 ON UNIT 00 TST 001 SUB 003 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
UDA RESIDENT DIAGNOSTICS DETECTED FAILURE
UDASA CONTAINS 105154
REPLACE UDA MODULE M7162

The UDA Resident diagnostic detected a failure. The error is displayed in the UDASA. Here are the possible error values and their meaning:

- 104000 - Fatal sequencer error
- 104040 - D processor ALU error
- 104041 - D proc ROM parity error
- 105102 - D proc with no Board #2 or RAM parity error
- 105105 - D proc RAM huffer error
- 105152 - D proc SDI error
- 105153 - D proc write mode wrap SERDES error
- 105154 - D proc read mode SERDES, RSGEN, and ECC error
- 106040 - U proc ALU error
- 106041 - U proc Control Register error
- 106042 - U proc DFAIL/ROM parity error/Board #1 test count is wrong
- 106047 - U proc Constant ROM error with D proc running SDI test
- 106055 - Unexpected trap found, aborted diagnostic
- 106071 - U proc ROM error
- 106072 - U proc ROM parity error
- 106200 - Step 1 data error (MSB not set)
- 107103 - U proc RAM parity error
- 107107 - U proc RAM buffer error

107115 - Board #2 test count was wrong
112300 - Step 2 error
122240 - NPR error
122300 - Step 3 error
142300 - Step 4 error

Replace the board specified. M7161 is the Unibus interface board.
M7162 is the SDI interface board.

00022 CZUDC DVC FTL ERR 00022 ON UNIT 00 TST 001 SUB 003 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
STEP BIT DID NOT SET IN UDASA REGISTER DURING INITIALIZATION
STEP BIT EXPECTED 004000
UDASA CONTAINS 000000
REPLACE UDA MODULE M7161

The UDA did not respond as expected during the initialization sequence which communicates using data in the UDASA register. A normal response from the UDA contains either a STEP bit or an ERROR bit defined as follows:

Bit 15 (100000)	Error bit
Bit 14 (040000)	Step 4 bit
Bit 13 (020000)	Step 3 bit
Bit 12 (010000)	Step 2 bit
bit 11 (004000)	Step 1 bit

The expected step bit nor the error bit set within the expected time.

00023 CZUDC DVC FTL ERR 00023 ON UNIT 00 TST 001 SUB 005 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
UDA DID NOT CLEAR RING STRUCTURE IN HOST MEMORY DURING INITIALIZATION
6 WORDS WERE TO BE CLEARED STARTING AT ADDRESS 040644
FIRST SEVERAL WORDS NOT CLEARED (UP TO 6):

ADDRESS	CONTENTS
040644	000010
040650	000010
040652	000010

REPLACE UDA MODULE M7161

The UDA is to clear the ring structure (a communications area used by the UDA to talk to the host) in host memory before Step 4 of initialization. If the UDA diagnostics did not clear memory and did not flag an error, then error message 00023 is displayed. The contents of each word in memory is set to 177777 before the test. Failure of the UDA to clear each word indicates a fault in the address interface to the Unibus.

00024 CZUDC DVC FTL ERR 00024 ON UNIT 00 TST 001 SUB 006 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
UDASA REGISTER DID NOT GO TO ZERO AFTER STEP 3 WRITE OF INITIALIZATION
PURGE/POLE DIAGNOSTICS WERE REQUESTED
UDASA CONTENTS 004400

For better testing, the host can test the PURGE and POLE mechanism of the UDA. To do so the host sets bit15 of the step 3 data and sends the data to the UDA. The UDA must go to zero and wait for the purge and pole. If the UDA never went to zero, then error message 00024 is displayed. The UDA may have a bad M7161 module or the UNIBUS may be broken.

00025 CZUDC DVC FTL ERR 00025 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
UDA DID NOT RETURN CORRECT DATA IN UDASA REGISTER DURING INITIALIZATION
UDASA EXPECTED 004400
UDASA CONTAINS 004000
REPLACE UDA MODULE M7161

For each step of initialization, specific data is expected to be displayed in the UDASA. If the UDASA does not match the expected data, then error message 00025 is displayed. Replace UDA module M7161.

00026 CZUDC DVC FTL ERR 00026 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
DATA COMPARISON ERROR DURING DIAGNOSTIC PORT LOOP TEST
DATA SENT TO UDASA 000001
RECEIVED FROM UDASA 000000
REPLACE UDA MODULE M7161

The UDA can be put into a mode where the UDASA acts as a wrap port. While the UDA is in this mode, any data being sent to the UDASA will be displayed in the UDASA within a small period of time. If the data in the UDASA does not match the data that was sent to the UDASA, then error message 00026 is displayed. Replace UDA module M7161.

00027 CZUDC DVC FTL ERR 00027 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
UDASA REGISTER DID NOT CHANGE AFTER WRITING TO IT
IN PORT LOOP DIAGNOSTIC
UDASA CONTAINS 004400
REPLACE UDA MODULE M7161

The UDA can be put into a mode where the UDASA acts as a wrap port. While the UDA is in this mode, any data being sent to the UDASA will be displayed in the UDASA within a small period of time. After the host program sent data to it while it was in diagnostic wrap mode, the UDA did not change the contents of the UDASA. Error message 00027 is displayed. Replace UDA module M7161.

00028 CZUDC DVC FTL ERR 00028 ON UNIT 00 TST 001 SUB 004 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
UDA DID NOT INTERRUPT THE PDP-11
REPLACE UDA MODULE M7161

The host program timed out while waiting for an interrupt that had to occur. The UDA was told to use interrupts during the initialization process. The UDA then waited for the interrupt but it did not occur. Replace the UDA module M7161.

00029 CZUDC DVC FTL ERR 00029 ON UNIT 00 TST 001 SUB 004 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
UDA INTERRUPTED AT DIFFERENT BR LEVEL THAN SPECIFIED IN HARDWARE
QUESTIONS. INTERRUPT WAS AT BR LEVEL 5
CHECK PRIORITY PLUG ON UDA MODULE M7161
OR CHANGE HARDWARE QUESTIONS

The priority plug on the UDA and the BR LEVEL specified during the hardware questions do not match. Either change the plug number or reanswer the hardware question. If all these have been done and there is still a problem replace UDA module M7161.

00030 CZUDC DVC FTL ERR 00030 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
UDA REPORTED FATAL ERROR IN UDASA REGISTER WHILE RUNNING DM PROGRAM
UDASA CONTAINS 100004

A message from the UDA firmware reports an unexpected failure. An error code is presented in the UDASA.
Here is a list of the codes and their meanings:

- 004400 - UDA has been initied by either a bus init or by writing into the UDAIP.
- 100001 - UNIBUS envelope/packet read error (parity or timeout)
- 100002 - UNIBUS envelope/packet write error (parity or timeout)
- 100003 - UDA ROM and RAM parity error
- 100004 - UDA RAM parity error
- 100005 - UDA ROM parity error
- 100006 - UNIBUS ring read error
- 100007 - UNIBUS ring write error
- 100010 - UNIBUS interrupt master failure
- 100011 - Host access timeout error
- 100012 - Host exceeded credit limit
- 100013 - UDA SDI hardware fatal error
- 100014 - DM XFC fatal error
- 100015 - Hardware timeout of instruction loop
- 100016 - Invalid virtual circuit identifier
- 100017 - Interrupt write error on UNIBUS

00031 CZUDC DVC FTL ERR 00031 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
NO INTERRUPT RECEIVED FROM DM PROGRAM FOR 3 MINUTES
ASSUME PROGRAM IS HUNG

All DM programs are required to communicate with the host program; so as to assure the host program that the DM program is not hung up or in an endless loop. If the DM program has not done so, the host program assumes the DM is hung and this message appears.

00032 CZUDC DVC FTL ERR 00032 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
MESSAGE BUFFER RECEIVED FROM DM PROGRAM WITH UNKNOWN REQUEST NUMBER
MESSAGE BUFFER CONTAINS:
000001 000002 000003 000004 000005 000006 000007
000008 000009 000010 000011 000012 000013 000014
000015 000016 000017 000018 000019 000020 000021
000022 000023 000024 000025 000026 000027 000028
000029 000030 000031 000032 000033 000034 000035

The DM program and the host program communicate with each other using packets. Each packet must have a request number set up by the DM program and interpreted by the host program. This request number is not a known request number. The problem may be the UNIBUS or either one of the UDA modules or a corrupted DM program. Word 1 contains the DM request number, and word 2 typically contains the drive number. The rest of the buffer contains information specific to a DM request. The numbers in the example show the order in which words are displayed.

00033 CZUDC DVC FTL ERR 00033 ON UNIT 00 TSi xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
RESPONSE PACKET FROM UDA DOES NOT CONTAIN EXPECTED DATA
EITHER UDA RETURNED ERROR STATUS OR PACKET WAS NOT RECEIVED CORRECTLY

COMMAND PACKET SENT	RESPONSE PACKET RECEIVED
000000 000020	000000 000020
000000 000000	000000 000000
000000 000002	000000 000202
000000 014336	000000 014336
000000 034674	000000 034674
000000 000000	000000 000000
000000 000000	000000 000000
000000 051232	000000 051232
000000 000000	000000 000000
000000 000000	000000 000000
000000 000000	000000 000000
000000 000000	000000 000000

The host program inspected the response packet which was given by to UDA. The response packet may have been in error with one of the following points:

- 1) The end code was not as expected.
- 2) The status code showed an error occurred with the last command.
- 3) The command reference numbers (the first word) did not match.

If 1 or 3 occurred, there may have been a transmission problem between the UDA and the host program. If 2 occurred, check the error code in the MSCP specification for further information. The packets are displayed two words per line, low order word and byte to the right (corresponding to the MSCP long-word entity).

00036 CZUDC DVC FTL ERR 00036 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
NO INTERRUPT RECEIVED FROM UDA FOR 30 SECONDS
WHILE LOADING DM PROGRAM

After a DM program has been sent to the UDA, the host program expects an interrupt within 30 seconds. The interrupt is used to assure the host program that the DM program is sane. If no interrupt occurred, then error message 00036 is displayed and the DM program is assumed to be hung.

00037 CZUDC DVC FTL ERR 00037 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
UDA REPORTED FATAL ERROR IN UDASA REGISTER WHILE LOADING DM PROGRAM
UDASA CONTAINS 100004
REPLACE UDA MODULE M7161

While loading the DM program to the UDA, the UDASA became non-zero. When this occurs, it signifies that the UDA microcode has run across a fatal error. The displayed value is in octal. Check the error code with the list included with error number 00030.

00038 CZUDC DVC FTL ERR 00038 ON UNIT 00 TST 001 SUB 002 PC: xxxxxx
HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
MEMORY ERROR TRYING TO READ UDA REGISTERS
CHECK UNIBUS SELECTION SWITCHES ON UDA MODULE M7161
OR UNIBUS
OR REPLACE UDA MODULE M7161

A non-existent memory error occurred when the host program tried to access the UDAIP and UDASA registers while in subtest 2 of test 1. The UDA is at another address (check the UNIBUS selection switches) or module M7161 is broken or the UNIBUS is broken.

3.2.2 TEST 1 ERROR MESSAGES (01000 TO 01999)

01000 CZUDC HRD ERR 01000 ON UNIT 00 TST 001 SUB 007 PC: xxxxxx
UNIBUS ADDRESSING DM PC:xxxx UDA AT xxxxxx RUNTIME hhh:mm:ss
NON-EXISTANT MEMORY ERROR TRYING TO READ FROM UNIBUS.

	OCTAL	HEX
ADDRESS	000000	00000

The host has given the DM routine the range of accessible host memory. While reading one location within the range, it appeared non-existent to the UDA. Since everything within the bounds were believed to be accessible this error message will be printed. The message prints the address in octal and hex.

01001 CZUDC HRD ERR 01001 ON UNIT 00 TST 001 SUB 007 PC: xxxxxx
UNIBUS ADDRESSING DM PC:xxxx UDA AT xxxxxx RUNTIME hhh:mm:ss
PARITY ERROR ON READ FROM UNIBUS.

	OCTAL	HEX
ADDRESS	000000	00000
DATA READ	000000	0000
DATA EXPECTED	000000	0000

The host has given the DM routine the range of accessible host memory. While reading one location within the range, the DM routine has found a location with bad parity. Every location was accessed by the host program. The host program filled a location with its address. The message prints the address, the data it actually received, and the expected data it should have received in octal and hex.

~~01002~~ CZUDC HRD ERR 01002 ON UNIT 00 TST 001 SUB 007 PC: xxxxxx
UNIBUS ADDRESSING DM PC:xxxx UDA AT xxxxxx RUNTIME hhh:mm:ss
UNIBUS ADDRESSING ERROR - INCORRECT DATA READ.
MEMORY LOCATION SHOULD CONTAIN OWN ADDRESS.

	OCTAL	HEX
DATA READ	000000	0000
DATA EXPECTED	000000	0000

The host has given the DM routine the locations of accessible host memory. Every location was accessed by the host program. The host program filled a location with its address. The DM program read from one location and found that the data it read was not equal to its address. The message prints the address, the data it actually received, and the expected data it should have received in octal and hex.

01003 CZUDC HRD ERR 01003 ON UNIT 00 TST 001 SUB 007 PC: xxxxxx
UNIBUS ADDRESSING DM PC:xxxx UDA AT xxxxxx RUNTIME hhh:mm:ss
NON-EXISTANT MEMORY ERROR TRYING TO READ FROM UNIBUS WITHIN BUFFER.

	OCTAL	HEX
STARTING ADDESS OF BUFFER	123456	0A72E
BUFFER SIZE	001234	029C

After reading every accessible location of host memory, the DM routine breaks up memory into buffers. The DM routine writes and reads data patterns from each host buffer into its DM buffer. While reading one of these buffers, a non-existent memory error occurred. The message prints out the starting address of the buffer and the size of the buffer in octal(for PDP-11 users) and in hex(for VAX users) so the user can determine about where the non-existent memory location occurred.

01004 CZUDC HRD ERR 01004 ON UNIT 00 TST 001 SUB 007 PC: xxxxxx
UNIBUS ADDRESSING DM PC:xxxx UDA AT xxxxxx RUNTIME hhh:mm:ss
PARITY ERROR ON READ FROM UNIBUS WITHIN BUFFER.

	OCTAL	HEX
STARTING ADDESS OF BUFFER	123456	0A72E
BUFFER SIZE	001234	029C

After reading every accessible location of host memory, the DM routine breaks up memory into buffers. The DM routine writes and reads data patterns from each host buffer into its DM buffer. While reading one of these buffers, a parity error occurred. The message prints out the starting address of the buffer and the size of the buffer in octal(for PDP-11 users) and in hex(for VAX users) so the user can determine about where the non-existent memory location occurred.

```

01005 CZUDC HRD ERR 01005 ON UNIT 00 TST 001 SUB 007 PC: xxxxxx
UNIBUS ADDRESSING DM PC:xxxx UDA AT xxxxxx RUNTIME hhh:mm:ss
DATA COMPARE FAILED AFTER WRITE THEN READ FROM UNIBUS.
BUFFER SIZE = 005302(O) 0AC2(X) 2754.(D)
STARTING ADDRESSES OF BUFFERS
OCTAL HEX
044232 0489A
057056 05E2E
071676 073BE
104512 0894A
CURRENT DATA PATTERN READ 0
LAST PATTERN WRITTEN 0
STARTING ADDRESS OF LAST BUFFER WRITTEN 104512(O) 0894A(X)
NUMBER OF ERRORS FOUND 2754.(D)
LOCATION DATA EXPECTED DATA RECEIVED
OCTAL HEX OCTAL HEX OCTAL HEX
057056 05E2E 111111 9249 002472 053A
057060 05E30 044444 4924 005302 0AC2
057062 05E32 022222 2492 000000 0000
  
```

After reading an entire buffer, the DM program checks each location. If any or all of the locations did not contain the expected data, this message appears. It contains the buffer size in octal, hex and decimal. The reason it appears in decimal is so the user can correlate this value with the number of errors which is printed in decimal. The starting addresses of the buffers are printed in octal and hex. There will always be at least two buffers and up to four buffers printed. The current data pattern read is printed. DM program will be testing the buffer with this data pattern. The last data pattern written by the DM program is printed. The address of the last buffer written is printed in octal and hex. As many as three errors are presented in the message. This portion presents the location of the error, the expected data and the actual data all in octal and hex.

01006 CZUDC HRD ERR 01006 ON UNIT 00 TST 001 SUB 007 PC: xxxxxx
UNIBUS ADDRESSING DM PC:xxxx UDA AT xxxxxx RUNTIME hhh:mm:ss
UNIBUS ADDRESSING ERROR. TWO ADDRESSES READ SAME LOCATION.

	OCTAL	HEX
KNOWN GOOD ADDRESS	625252	32AAA
ERROR ADDRESS	425252	22AAA
ADDRESS BIT IN ERROR	200000	10000

The UDA can only write to a small portion of memory because there is a PDP-11 program running in the memory. To verify it can address all of memory, it uses one location that it is permitted to write which it calls a "known good address". By changing only one bit in the address of this location it selects a "test address". Different patterns are written to the "known good address", each followed by a read of the "test address". If the data read from the "test address" matches the data written to the "known good address" each time, the address line is determined to be stuck. The "test address" is printed as the error address.

3.2.3 TEST 2 INFORMATIONAL MESSAGES

UNIT x UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
INFORMATION SENT BACK FROM THE DRIVE IS BEING PRESENTED.

TEST NUMBER 0000
DRIVE TYPE 00
ERROR NUMBER 0000
data

There is not error, but it is a message. The disk drive wanted the let the host know what had happened when the drive's internal diagnostic was run. The format follows that of hard error 2021.

UNIT x UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
FOLLOWING REPORT HAS BEEN TRUNCATED DUE TO SIZE

This is a message that may appear if the disk drive gave too much data for the DM program to handle. This message may precede the previous message and hard error 2021.

3.2.4 TEST 2 ERROR MESSAGES (02000 TO 02999)

02000 CZUDC HRD ERR 02000 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
HOST SPECIFIED UNIT #0 THAT CAN'T BE FOUND.
TEST2 RESTARTING

When test 2 starts executing out of the DM, it doesn't know if it had been started to execute drive diagnostics or restarted to down line load a diagnostic into the drive. If it had been restarted for the latter reason, the host must tell Test 2 which drive was to receive the diagnostic. If the drive specified by the host is not attached to the UDA or could not be located by Test 2, this error message will be printed.

02001 CZUDC HRD ERR 02001 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
CANNOT RECEIVE VALID DRIVE STATE FROM DRIVE AFTER DRIVE WAS INITED
CHECK IF DRIVE IS POWERED ON.

This error message is presented if valid drive state was not received from the drive after the drive was initied. There are two types of invalid states: no clocks or 'hard' errors. If after getting state and no clocks occur, error 2001 is reported. There may be a bad transmitter on the drive side or a bad receiver on the UDA side or the SDI cable may have taken a hit.

02002 CZUDC HRD ERR 02002 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DRIVE STATE RECEIVED HAS BAD PARITY AFTER DRIVE WAS INITED

This error message is presented if bad parity was received from the drive after the drive was initied. There may be a bad transmitter on the drive side or a bad receiver on the UDA side or the SDI cable may have taken a hit.

02003 CZUDC HRD ERR 02003 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DRIVE IS NOT ASSERTING RECEIVER READY IN DRIVE STATE AFTER DRIVE WAS INITED

This error message is presented if receiver ready was not received from the drive after the drive was initied. There may be a bad transmitter on the drive side or a bad receiver on the UDA side or the SDI cable may have taken a hit.

02004 CZUDC HRD ERR 02004 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF ECHO COMMAND TO DRIVE
ECHO DATA FF

This error message is presented if a send of the ECHO command timed out. This may be caused by receiver ready being deasserted. The echo data is presented in hex.

02005 CZUDC HRD ERR 02005 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF ECHO RESPONSE FROM DRIVE
ECHO DATA FF

This error message is presented if a receive of an ECHO command was in error. The echo data is presented in hex. There may be a bad transmitter on the drive side or a bad receiver on the UDA side or the SDI cable may have taken a hit.

02006 CZUDC HRD ERR 02006 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ECHO COMMAND RESPONDED WITH DIFFERENT DATA
ECHO DATA SENT 00FE
ECHO DATA RECEIVED 00FF

This error message is presented if the data returning from an ECHO command did not match the data it was suppose to. The data presented is in hex.

02007 CZUDC HRD ERR 02007 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR BIT SET IN GET STATUS RESPONSE AFTER DRIVE CLEAR COMMAND
GET STATUS RESPONSE
REAL TIME STATE state
STATUS (FROM R TO L): word6 word5 word4 word3 word2 word1 word0:

This error message is presented when an error bit is set in the status of a drive after the drive was cleared of all errors. The data displayed is the response from a GET STATUS command. The error bits in the response are in bit position 3, 5 and 6 of word2. For further description of the GET STATUS response, refer to the SDI Functional Spec v3.6 and the drive's functional spec.

REAL TIME STATE state: REAL TIME STATE 0003

The real time state is the real time drive state <<AFTER>> Test 2 detected the error. <<THIS VALUE IS DISPLAYED IN HEX>>. In this example, receiver ready and attention are both asserted.

The bit positions are defined as follows:

- 0001 - Receiver ready (Test 2 able to transmit to drive)
- 0002 - Attention (error occurred or online timeout expired)
- 0040 - Available (drive offline and usable)
- 1000 - Read/Write ready

The complete meaning of these bits is beyond the scope of this text, please refer to the operator documentation for the drive you are working on.

STATUS (R TO L): word6 word5 word4 word3 word2 word1 word0:
The status is the response to the SDI GET STATUS command. These words are printed in HEX. <<NOTE THAT THE STATUS IS PRINTED OUT FROM RIGHT TO LEFT!!>>. The status' meaning is beyond the scope of this text, please refer to the operator documentation for the drive you are working on.

02008 CZUDC HRD ERR 02008 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF ONLINE COMMAND TO DRIVE

The ONLINE command timed out while it was sent to the drive. The drive did not assert the RECEIVER READY signal over the SDI.

02009 CZUDC HRD ERR 02009 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF ONLINE RESPONSE FROM DRIVE
explanation

This error message is presented if a receive of an ONLINE command was in error. An explanation of what the error was is also presented. These explanations are:

- TIMEOUT ERROR OCCURED DURING RECEIVE XFC
- This error is a failure of the drive to respond to an SDI level 2 command (see the SDI specification) before the drive-supplied command timeout expires.
- 1ST WORD NOT START FRAME DURING RECEIVE XFC
- The first word received by the UDA from the drive was not a valid message start frame.
- FRAMING ERROR OCCURED ON SDI LEVEL 0 READ DURING RECEIVE XFC
- This is caused by one of the following conditions:
1) Illegal frame code -- the frame is not a message start, continue, or end frame. 2) Illegal sequence of frames -- such as a message start frame without ever receiving a message end frame. This can be caused by the drive sending a response before the UDA asserts receiver ready, or a random hit on the SDI cable that garbles a frame or a bad drive transmitter or UDA receiver.
- CHECKSUM ERROR OCCURED ON SDI LEVEL 0 READ DURING RECEIVE XFC
- The checksum attached to a message end frame did not match the checksum computed over the level 2 command. This could be caused by a bad drive transmitter, bad UDA receiver, incorrectly computed checksum by the drive (unlikely) or a random hit on the SDI cable.

BUFFER SIZE SMALLER THEN RESPONSE DURING RECEIVE XFC

- A buffer size size set aside for the response was not large enough for the response received. This is caused by the drive sending a response that is incorrect for the request sent to the drive, or the drive sending some garbage with the response.

CODE FROM RECEIVE XFC WAS UNINTELLIGIBLE FROM SUBSYSTEM 0000

- The response from the drive was not anything that was expected. Possible UDA microcode change without test 2 update.

02010 CZUDC HRD ERR 02010 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ONLINE COMMAND WAS UNSUCCESSFUL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The ONLINE command was not successful. The drive's status is displayed. See hard error 2007 for further information on the format of the status. The drive did not assert the RECEIVER READY signal over the SDI.

02011 CZUDC HRD ERR 02011 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ONLINE COMMAND DID NOT RETURN EXPECTED RESPONSE CODE
EXPECTED RESPONSE 7E
ACTUAL RESPONSE 00

The ONLINE command did not return an expected response code. If there were at least an UNSUCCESSFUL response, test 2 will report the drive state and status. The expected response and actual response are in hex.

02012 CZUDC HRD ERR 02012 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF GET UNIT CHARACTERISTICS COMMAND TO DRIVE

The GET UNIT CHARACTERISTICS command timed out while it was sent to the drive. The drive did not assert the RECEIVER READY signal over the SDI.

02013 CZUDC HRD ERR 02013 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF GET UNIT CHARACTERISTICS COMMAND FROM DRIVE
explanation

This error message is presented if a receive of a GET UNIT CHARACTERISTICS command was in error. An explanation of what the error was is also presented. These explanations are described in hard error 2009.

02014 CZUDC HRD ERR 02014 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
GET UNIT CHARACTERISTICS COMMAND WAS UNSUCCESSFUL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The GET UNIT CHARACTERISTICS command was not successful.
The drive's status is displayed. See hard error 2007 for
further information on the format of the status.

02015 CZUDC HRD ERR 02015 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
GET UNIT CHARACTERISTICS COMMAND DID NOT RETURN EXPECTED RESPONSE CODE'\
EXPECTED RESPONSE 78
ACTUAL RESPONSE 00

The GET UNIT CHARACTERISTICS command did not return an expected
response code. The expected response and actual response
are in hex.

02016 CZUDC HRD ERR 02016 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
HOST PROGRAM GAVE DM CODE IMPROPER DATA
EXPECTED VALUE SHOULD BE BETWEEN 0 AND 3
ACTUAL VALUE WAS xx

The host tells the DM program what to do after the DM
program is done testing the drive's diagnostic. If
the value is not within the expected range, this error
message is printed. There is no drive problem. The
problem is between the host and the UDA.

02017 CZUDC HRD ERR 02017 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF DIAGNOSE COMMAND TO DRIVE

The DIAGNOSE command timed out while it was sent
to the drive. The drive did not assert
the RECEIVER READY signal over the SDI.

02018 CZUDC HRD ERR 02018 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF DIAGNOSE RESPONSE FROM DRIVE
explanation

This error message is presented if a receive of a DIAGNOSE
command was in error. An explanation of what the error was
is also presented. These explanations are described in
hard error 2009.

02019 CZUDC HRD ERR 02019 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DIAGNOSE COMMAND WAS UNSUCCESSFUL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The DIAGNOSE command was not successful. The drive's status is displayed. See hard error 2007 for further information on the format of the status.

02020 CZUDC HRD ERR 02020 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DIAGNOSE COMMAND DID NOT RETURN EXPECTED RESPONSE CODE
EXPECTED RESPONSE FC
ACTUAL RESPONSE 00

The DIAGNOSE command did not return an expected response code. The expected response and actual response are in hex.

02021 CZUDC HRD ERR 02021 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DRIVE DIAGNOSTIC REPORTS A HARD ERROR
TEST NUMBER 0000
DRIVE TYPE 00
ERROR NUMBER 0000
data

The drive diagnostic found an error and is reporting the error back to the host. All values are in hex. TEST NUMBER shows what test was run. DRIVE TYPE shows what type of drive was being tested. ERROR NUMBER shows the result of the test. The drive may pass back data to the host. This data will be presented in a 32 bit hex format following the error message. More data may follow the 32 bit hex values. This data is printed in ascii format. For definitions of what these values mean, refer to the drive functional spec.

02022 CZUDC HRD ERR 02022 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
HOST PROGRAM DOWN LINE LOADED A DIAGNOSTIC WITH A ZERO BYTE COUNT

The host program was attempting to down line load a diagnostic of zero length. The DM program must have the byte count specified by the host.

02023 CZUDC HRD ERR 02023 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DIAGNOSTIC filnam REQUESTED BY THE DRIVE COULD NOT BE SUPPLIED BY HOST.

The host program could not supply the diagnostic 'filnam' to down line load to the drive.

02024 CZUDC HRD ERR 02024 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF MEMORY READ COMMAND TO DRIVE

The MEMORY READ command timed out while it was sent to the drive. The drive did not assert the RECEIVER READY signal over the SDI.

02025 CZUDC HRD ERR 02025 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF MEMORY READ RESPONSE FROM DRIVE
explanation

This error message is presented if a receive of a MEMORY READ command was in error. An explanation of what the error was is also presented. These explanations are described in hard error 2009.

02026 CZUDC HRD ERR 02026 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
MEMORY READ COMMAND WAS UNSUCCESSFUL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The MEMORY READ command was not successful. The drive's status is displayed. See hard error 2007 for further information on the format of the status.

02027 CZUDC HRD ERR 02027 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
MEMORY READ COMMAND DID NOT RETURN EXPECTED RESPONSE CODE
EXPECTED RESPONSE 72
ACTUAL RESPONSE 00

The MEMORY READ command did not return an expected response code. The expected response and actual response are in hex.

02028 CZUDC HRD ERR 02028 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF MEMORY WRITE COMMAND TO DRIVE

The MEMORY WRITE command timed out while it was sent to the drive. The drive did not assert the RECEIVER READY signal over the SDI.

02029 CZUDC HRD ERR 02029 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF MEMORY WRITE RESPONSE FROM DRIVE
explanation

This error message is presented if a receive of a MEMORY WRITE command was in error. An explanation of what the error was is also presented. These explanations are described in hard error 2009.

02030 CZUDC HRD ERR 02030 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
MEMORY WRITE COMMAND WAS UNSUCCESSFUL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The MEMORY WRITE command was not successful. The drive's status is displayed. See hard error 2007 for further information on the format of the status.

02031 CZUDC HRD ERR 02031 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
MEMORY WRITE COMMAND DID NOT RETURN EXPECTED RESPONSE CODE
EXPECTED RESPONSE 7E
ACTUAL RESPONSE 00

The MEMORY WRITE command did not return an expected response code. The expected response and actual response are in hex.

02032 CZUDC HRD ERR 02032 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF RUN COMMAND TO DRIVE

The RUN command timed out while it was sent to the drive. The drive did not assert the RECEIVER READY signal over the SDI.

02033 CZUDC HRD ERR 02033 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF RUN RESPONSE FROM DRIVE
explanation

This error message is presented if a receive of a RUN command was in error. An explanation of what the error was is also presented. These explanations are described in hard error 2009.

02034 CZUDC HRD ERR 02034 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
RUN COMMAND WAS UNSUCCESSFUL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The RUN command was not successful. The drive's status is displayed. See hard error 2007 for further information on the format of the status.

02035 CZUDC HRD ERR 02035 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
RUN COMMAND DID NOT RETURN EXPECTED RESPONSE CODE
EXPECTED RESPONSE 7E
ACTUAL RESPONSE 00

The RUN command did not return an expected response code. The expected response and actual response are in hex.

02036 CZUDC HRD ERR 02036 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF RECALIBRATE COMMAND TO DRIVE

The RECALIBRATE command timed out while it was sent to the drive. The drive did not assert the RECEIVER READY signal over the SDI.

02037 CZUDC HRD ERR 02037 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF RECALIBRATE RESPONSE FROM DRIVE
explanation

This error message is presented if a receive of a RECALIBRATE command was in error. An explanation of what the error was is also presented. These explanations are described in hard error 2009.

02038 CZUDC HRD ERR 02038 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
RECALIBRATE COMMAND WAS UNSUCCESSFUL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The RECALIBRATE command was not successful. The drive's status is displayed. See hard error 2007 for further information on the format of the status.

02039 CZUDC HRD ERR 02039 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
RECALIBRATE COMMAND DID NOT RETURN EXPECTED RESPONSE CODE
EXPECTED RESPONSE 7E
ACTUAL RESPONSE 00

The RECALIBRATE command did not return an expected response code. The expected response and actual response are in hex.

02040 CZUDC HRD ERR 02040 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF GET STATUS COMMAND TO DRIVE

The GET STATUS command timed out while it was sent to the drive. The drive did not assert the RECEIVER READY signal over the SDI.

02041 CZUDC HRD ERR 02041 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF GET STATUS RESPONSE FROM DRIVE
explanation

This error message is presented if a receive of a GET STATUS command was in error. An explanation of what the error was is also presented. These explanations are described in hard error 2009.

02042 CZUDC HRD ERR 02042 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
GET STATUS COMMAND WAS UNSUCCESSFUL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The GET STATUS command was not successful. The drive's status is displayed. See hard error 2007 for further information on the format of the status.

02043 CZUDC HRD ERR 02043 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
GET STATUS COMMAND DID NOT RETURN EXPECTED RESPONSE CODE
EXPECTED RESPONSE F6
ACTUAL RESPONSE 00

The GET STATUS command did not return an expected response code. The expected response and actual response are in hex.

02044 CZUDC HRD ERR 02044 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND OF DRIVE CLEAR COMMAND TO DRIVE

The DRIVE CLEAR command timed out while it was sent to the drive. The drive did not assert the RECEIVER READY signal over the SDI.

02045 CZUDC HRD ERR 02045 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
ERROR DURING RECEIVE OF DRIVE CLEAR RESPONSE FROM DRIVE
explanation

This error message is presented if a receive of a DRIVE CLEAR command was in error. An explanation of what the error was is also presented. These explanations are described in hard error 2009.

02046 CZUDC HRD ERR 02046 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DRIVE CLEAR COMMAND WAS UNSUCCESSFUL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The DRIVE CLEAR command was not successful. The drive's status is displayed. See hard error 2007 for further information on the format of the status.

02047 CZUDC HRD ERR 02047 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK RESIDENT DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DRIVE CLEAR COMMAND DID NOT RETURN EXPECTED RESPONSE CODE
EXPECTED RESPONSE 7E
ACTUAL RESPONSE 00

The DRIVE CLEAR command did not return an expected response code. The expected response and actual response are in hex.

3.2.5 TEST 3 INFORMATIONAL MESSAGES

UNIT xx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
LOGGABLE INFORMATION AFTER RECAL
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

After sending a RECALIBRATE command, the ATTENTION bit was set. Test 3 then sent a GET STATUS command and found the LOGGABLE INFORMATION bit was set. This is not an error, it is only some information being sent from the drive. Normal operation continues.

Check 03001 for explanation of 'REAL TIME STATE' and 'STATUS'

3.2.6 TEST 3 ERROR MESSAGES (03000 TO 03999)

03001 CZUDC HRD ERR 03001 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT ON SEND
COMMAND WAS command
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

If test 3 tries to send a level 2 command to the drive, and receiver ready is deasserted, error 3001 occurs. Where command is one of the following:

GET COMMON CHARACTERISTICS
ONLINE
DRIVE CLEAR
DISCONNECT
GET SUBUNIT CHARACTERISTICS
GET STATUS
CHANGE MODE
INITIATE RECLIBRATE
SPIN UP

REAL TIME STATE state: REAL TIME STATE 0003

The real time state is the real time drive state <<AFTER>> Test 3 detected the error. <<THIS VALUE IS DISPLAYED IN HEX>>. In this example, receiver ready and attention are both asserted.

The bit positions are defined as follows:

0001 - Receiver ready (Test 3 able to transmit to drive)
0002 - Attention (error occurred or online timeout expired)
0040 - Available (drive offline and usable)
1000 - Read/Write ready

The complete meaning of these bits is beyond the scope of this text, please refer to the operator documentation for the drive you are working on.

STATUS (R TO L): word6 word5 word4 word3 word2 word1 word0:

The status is the response to the SDI GET STATUS command. These words are printed in HEX. <<NOTE THAT THE STATUS IS PRINTED OUT FROM RIGHT TO LEFT!!>>. The status' meaning is beyond the scope of this text, please refer to the operator documentation for the drive you are working on.

03002 CZUDC HRD ERR 03002 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
TIME-OUT OF RECEIVE
COMMAND WAS GET COMMON CHARACTERISTICS
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

This error is a failure of the drive to respond to an SDI level 2 command (see the SDI specification) before the drive-supplied command timeout expires.

Check 03001 for explanation of 'REAL TIME STATE' and 'STATUS'

03003 CZUDC HRD ERR 03003 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
FIRST WORD RECEIVED WAS NOT A START FRAME
COMMAND WAS GET COMMON CHARACTERISTICS
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The first word received by the UDA from the drive was not a valid message start frame.

03004 CZUDC HRD ERR 03004 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
FRAMING ERROR ON LEVEL 0 RESPONSE
COMMAND WAS GET COMMON CHARACTERISTICS
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

Error 3004 is caused by one or more of the following conditions: 1) Illegal frame code -- the frame is not a message start, continue, or end frame. 2) Illegal sequence of frames -- such as a message start frame without ever receiving a message end frame. This can be caused by the drive sending a response before the UDA asserts receiver ready, or a random hit on the SDI cable that garbles a frame or a bad drive transmitter or UDA receiver.

Check 03001 for explanation of 'REAL TIME STATE' and 'STATUS'

03005 CZUDC HRD ERR 03005 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
CHECKSUM ERROR ON LEVEL 0 RESPONSE
COMMAND WAS GET COMMON CHARACTERISTICS
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The checksum attached to a message end frame did not match the checksum computed over the level 2 command. This could be caused by a bad drive transmitter, bad UDA receiver, incorrectly computed checksum by the drive (unlikely) or a random hit on the SDI cable.

Check 03001 for explanation of 'REAL TIME STATE' and 'STATUS'

03006 CZUDC HRD ERR 03006 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
RESPONSE LONGER THAN EXPECTED
COMMAND WAS GET COMMON CHARACTERISTICS
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The buffer size set aside for the response was not large enough for the response received. This is caused by the drive sending a response that is incorrect for the request sent to the drive, or the drive sending some garbage with the response.

Check 03001 for explanation of 'REAL TIME STATE' and 'STATUS'

03007 CZUDC HRD ERR 03007 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
CODE FROM RECEIVE WAS UNINELLIGIBLE FROM SUBSYSTEM = 0000
COMMAND WAS GET COMMON CHARACTERISTICS
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The unknown error code occurs when the UDA returns an error code from an operation that test 3 does not recognize. Possible UDA microcode change without test 3 update.

Check 03001 for explanation of 'REAL TIME STATE' and 'STATUS'

03008 CZUDC HRD ERR 03008 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
COMMAND DID NOT RETURN EXPECTED RESPONSE CODE
COMMAND WAS GET COMMON CHARACTERISTICS
EXPECED RESPONSE 7E
ACTUAL RESPONSE 7D
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

This is caused by receiving an UNSUCCESSFUL response from the drive, or the drive sending some response other than the correct response for the request sent to the drive. See the contents of status for the unexpected response error (or reason).

Check 03001 for explanation of 'REAL TIME STATE' and 'STATUS'

03009 CZUDC HRD ERR 03009 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DRIVE NOT ASSERTING RECEIVER READY IN DRIVE STATE
REAL TIME STATE 0003

Test 3 inits the drive and checks the drive's real time state. If RECEIVER READY was not asserted after a period of time this error message is printed.

03010 CZUDC HRD ERR 03010 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
FAILED TO RECEIVE VALID DRIVE STATE
REAL TIME STATE 0003

There are two types of invalid state: no clocks or 'hard'
errors. If after getting state and no clocks occur,
error 3010 is reported. Check the drive state for further
information.

03011 CZUDC HRD ERR 03011 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
CANNOT RECEIVE DRIVE STATE FROM DRIVE
CHECK IF DRIVE IS POWERED ON.
REAL TIME STATE 0003

After the test 3 sends the drive a DISCONNECT command
test 3 should be able to receive state from the drive.
The drive may have spun down after the DISCONNECT command.

03012 CZUDC HRD ERR 03012 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
DRIVE STATE RECEIVED HAS BAD PARITY
REAL TIME STATE 0003

As in 3010, we can get two types of invalid state. If
parity or pulse errors occur for 1/2 a second, either
the transmitter or receiver is bad. This could be caused
by a bad transmitter or receiver or by a hit on the SDI
cable.

03013 CZUDC DVC FTL 03013 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
NO VALID STATE FROM DRIVE
REAL TIME STATE 0003

The drive received either one of the two types of invalid
state that are described in 3010 and 3012. Check state
for further information. This could be caused by a bad
transmitter or receiver or by a hit on the SDI cable.

03014 CZUDC HRD ERR 03014 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
SUBUNIT CHARACTERISTICS SAY THERE ARE ZERO READ ONLY GROUPS
IN THE DIAGNOSTIC AREA

After interrogating the subunit characteristics, test 3
finds out that the drive claims there are zero read only
groups in the diagnostic area. There must be at least
one for the test to run.

03015 CZUDC HRD ERR 03015 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
SUBUNIT CHARACTERISTICS SAY THERE ARE LESS THAN 1 READ/WRITE
GROUPS IN THE DIAGNOSTIC AREA

After interrogating the subunit characteristics, test 3 finds out that the drive claims there are zero read/write groups in the diagnostic area. There must be at least one for the test to run.

03016 CZUDC HRD ERR 03016 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
NEITHER R/W READY NOR ATTENTION SET AFTER RECALIBRATE COMMAND
REAL TIME STATE 0003

After a RECALIBRATE command, R/W READY or ATTENTION did not set. Check the state for further information. This could be caused by a bad transmitter or receiver or by a hit on the SDI cable.

03017 CZUDC HRD ERR 03017 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
SUBUNIT CHARACTERISTICS SAY LESS THAN 1 DIAGNOSTIC CYLINDER

After interrogating the subunit characteristics, test 3 finds out that the drive claims there are zero diagnostic cylinders. There must be at least one for the test to run.

03018 CZUDC HRD ERR 03018 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
READ/WRITE READY DROPPED BEFORE FORMAT OPERATION

The R/W READY signal was deasserted by the drive before a format operation was going to be sent by the UDA. The drive may have gone off line or is not transmitting properly or the UDA may not be receiving properly or the SDI cable took a hit.

03019 CZUDC HRD ERR 03019 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
FORMAT OPERATION REPORTED TIME-OUT FAILURE
CYLINDER aaa. GROUP bb. TRACK cc.

The format operation sent by the UDA failed. The command timed out possibly due to receiver ready being dropped or communication problem (bad transmitter or receiver or hit on the SDI cable)

Where:

aaa is the cylinder value in decimal.
bb is the group value in decimal.
cc is the track value in decimal.

03020 CZUDC HRD ERR 03020 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
AFTER RECAL, ERROR BITS WERE SET
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

After sending a RECALIBRATE command, the ATTENTION bit was set. Test 3 then sent a GET STATUS command and found the error bits were set. For further information, check the state and the status.

Check 03001 for explanation of 'REAL TIME STATE' and 'STATUS'

03022 CZUDC HRD ERR 03022 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
READ/WRITE READY DROPPED BEFORE WRITE OPERATION

The R/W READY signal was deasserted by the drive before a write operation was going to be sent by the UDA. The drive may have gone off line or is not transmitting properly or the UDA may not be receiving properly or the SDI cable took a hit.

03023 CZUDC HRD ERR 03023 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
COULD NOT WRITE AND READ ANY BLOCK ON THIS TRACK. ON LAST BLOCK:
WRITE OPERATION REPORTED FAILURE -- ERROR CODE aaa OCTAL.
DBN bbb. CYLINDER ccc. GROUP dd. TRACK ee.

After each track in the diagnostic space is formatted, at least one block must be able to have data written to it and read from it and the data must be correct. Not one block (DBN bbb.) from track (ee) was able to pass. The error code (aaa) gives the reason for the write operation failure.

Where:

aaa is the error code in octal.

It may have one of the following values:

2 = drive failure

3 = requested LBN is a secondary revector.

<<< NOTE >>> We are working with DBN's

4 = header compare failure
(desired header not found)

153 = suspected positioner error

213 = read/write ready failure

253 = drive data or state clock timeout
(indicates cable/transmitter/
receiver broken)

313 = receiver ready timeout

413 = drive state receive error during write

bbb is the DBN in decimal.

ccc is the cylinder value in decimal.

dd is the group value in decimal.

ee is the track value in decimal.

03024 CZUDC HRD ERR 03024 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
READ/WRITE READY DROPPED BEFORE READ OPERATION

The R/W READY signal was deasserted by the drive before a read operation was going to be sent by the UDA. The drive may have gone off line or is not transmitting properly or the UDA may not be receiving properly or the SDI cable took a hit.

03025 CZUDC HRD ERR 03025 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
COULD NOT WRITE AND READ ANY BLOCK ON THIS TRACK. ON LAST BLOCK:
READ OPERATION REPORTED FAILURE -- ERROR CODE aaa OCTAL.
DBN bbb. CYLINDER ccc. GROUP dd. TRACK ee.

After each track in the diagnostic space is formatted, at least one block must be able to have data written to it and read from it and the data must be correct. Not one block (DBN bbb.) from track (ee) was able to pass. The error code (aaa) gives the reason for the read operation failure.

Where:

aaa is the error code in octal.

It may have one of the following values:

2 = drive failure

3 = requested LBN is a secondary revector.

<<< NOTE >>> We are working with DBN's

4 = header comparison failure
(desired header not found)

52 = SERDES overrun error

150 = data sync timeout on read

153 = suspected positioner error

213 = read/write ready failure

253 = drive data or state clock timeout
(indicates cable/transmitter/
receiver broken)

313 = receiver ready timeout

413 = drive state receive error during write

bbb is the DBN in decimal.

ccc is the cylinder value in decimal.

dd is the group value in decimal.

ee is the track value in decimal.

03026 CZUDC HRD ERR 03026 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
COULD NOT WRITE AND READ ANY BLOCK ON THIS TRACK. ON LAST BLOCK:
DATA COMPARE FAILURE ON WORD aa.
EXPECTED DATA bbbb
ACTUAL DATA cccc
CYLINDER ddd. GROUP ee. TRACK ff.

After each track in the diagnostic space is formatted, at least one block must be able to have data written to it and read from it and the data must be correct. Not one block (DBN bbb.) from track (ee) was able to pass. The data read did not match the data written.

Where:

aa is the offset in decimal into the buffer where the error occurred.
bbbb is the expected data in hex.
cccc is the actual data in hex.
ddd is the cylinder value in decimal.
ee is the group value in decimal.
ff is the track value in decimal.

03027 CZUDC HRD ERR 03027 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
SEEK COMPLETE TIME-OUT -- READ/WRITE READY DID NOT SET
SEEK WAS TO CYLINDER aaa. GROUP bb.

After a SEEK command has been successfully sent from the UDA to the drive, the signal READ/WRITE READY must be set to indicate that the seek completed. If READ/WRITE READY never is asserted by the drive after the seek, the seek times out and error 3027 is presented.

Where:

aaa is the cylinder in decimal.
bb is the group in decimal.

03028 CZUDC HRD ERR 03028 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
NO BLOCK ON THIS TRACK CAN BE READ. LAST BLOCK TRIED:
aBN bbbb. CYLINDER ccc. GROUP dd. TRACK ee.

After a seek to a track, at least one block must be able to be read to assure that test 3 can read the header. If not one block was successful, error message 3028 appears.

Where:

a is 'L' for LBN, 'D' for DBN, or 'X' for XBN.
bbbb is the block number in decimal.
ccc is the cylinder in decimal.
dd is the group number in decimal.
ee is the track number in decimal.

03029 CZUDC HRD ERR 03029 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
AVAILABLE WAS NOT ASSERTED AFTER DISCONNECT
STATE RECEIVED state

After the DISCONNECT command was sent, the AVAILABLE flag should be asserted after a period of time. If it never was, then error 3029 appears. There maybe a problem with a transmitter or a receiver or the SDI cable at this point.

03030 CZUDC HRD ERR 03030 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
INVALID COMMAND aaaa WAS SUCCESSFUL

Some invalid level 2 commands are sent over the SDI. The drive should find these illegal commands and flag them as such. If the drive doesn't, then error 3030 will appear.

Where aaaa is the invalid command in hex.

03031 CZUDC HRD ERR 03031 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
COMMAND WITH type LENGTH = a WAS SUCCESSFUL

SDI level 2 commands with invalid lengths are sent to the drive to check if the drive can find them.

Where:

type could be 'COMMAND' or 'RESPONSE' for which
field was affected
a is the invalid length

03032 CZUDC HRD ERR 03032 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
UNIT DID NOT REPORT TRANSMISSION ERROR
WHEN reason

Invalid level 1 sequences were sent to the drive. Several sequences are tried and the drive should find fault with everyone of them.

Where reason could be one of the following:

AN END FRAME WAS SENT AFTER A START FRAME TIMED OUT
A CONTINUE OR END FRAME DID NOT FOLLOW A START FRAME
AN END FRAME WAS SENT WITH NO START FRAME
AN END FRAME WITH A BAD CHECKSUM WAS SENT
A CONTINUE FRAME WAS SENT WITH NO START FRAME

03033 CZUDC HRD ERR 03033 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
UNIT ACCEPTED AN INVALID GROUP NUMBER FROM GROUP SELECT LEVEL 1

A level 1 select group command with an illegal group number is sent to the drive. If the drive accepted it, then error 3033 will be displayed.

03034 CZUDC DVC FTL ERR 03034 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
UNABLE TO CORRECTLY READ OVERLAY x
THIS UDA AND ALL DRIVES ATTACHED WILL BE REMOVED FROM TESTING

There are two overlays in test 3. For some reason that the overlay cannot be read correctly, error 3034 will be displayed. Since no code can be loaded into the UDA at this point, the UDA and all attached drives will cease to be tested. The reason for this may be bad UNIBUS memory or board 1 may be failing.

<<< NOTE >>> This is -- NOT -- a drive failure.

03035 CZUDC DVC FTL ERR 03035 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
SUCCESSFULLY WROTE ON DMN AREA WHEN DRIVE WAS WRITE PROTECTED

An attempt was made to write on a write protected drive. It should have resulted in an error response from the disk drive, but it didn't.

3.2.7 TEST 4 INFORMATIONAL MESSAGES

UNIT u UDA AT cccccc DRIVE n RUNTIME hh:mm:ss
A CORRECTABLE ECC ERROR EXISTS IN type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder

The above message occurs when Test 4 1) detects an ECC error and 2) is able to correct it, and 3) the corrections are less than the drive ECC threshold, (a SDI DRIVE CHARACTERISTIC) and 4) the EDC computed over the corrected sector matched the EDC read.

UNIT unit UDA AT udaadr DRIVE plug RUNTIME hh:mm:ss
INITIAL WRITE COMPLETE

Whenever Test 4 is STARTed with initial write enabled, <<OR>> whenever it is STARTed or RESTARTed and the diagnostic area is being tested on a drive not in read only mode, the disk will be initially written. The above message occurs when the initial write completes.

UNIT unit UDA AT udaadr DRIVE plug RUNTIME hh:mm:ss
READ ONLY DRIVE, INITIAL WRITE WILL NOT BE PERFORMED

If an initial write is to be performed (see above for conditions) and a unit or subunit is in read only mode, (can be set in the manual intervention questions) an initial write will not be performed, and this message will print to inform the operator.

NOTE: DATA COMPARE ERRORS RESULT IF THE DISK IS NOT INITIALLY WRITTEN!!

UNIT unit UDA AT udaadr DRIVE plug RUNTIME hh:mm:ss
THE PREVIOUS DEVICE FATAL WILL CAUSE THE FOLLOWING DRIVES
TO BE DROPPED: plug, plug+1, plug+2, plug+3

plug: drive plug number -- each subunit's plug number is displayed. for a single subunit drive (such as and RA80) only one plug number is displayed.

If a device fatal error occurs and dropping is enabled, <<ALL>> subunits on the unit that the device fatal occurred must be dropped. To inform the operator, this message is printed after the device fatal error message.

NOTE: IF MORE THAN ONE UDA IS ON A SYSTEM, THIS MESSAGE MAY NOT IMMEDIATELY FOLLOW THE DEVICE FATAL IF AN ERROR HAPPENS AT THE SAME TIME ON ANOTHER UDA.

3.2.8 TEST 4 ERROR MESSAGES (04000 TO 04999)

04001 CZUDC SFT ERR 04001 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ATTN ASSERTED DURING SEEK -- ERROR OR LOGGABLE INFORMATION
SEEK FROM GRP group CYL cylinder TO GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

This error occurs when the drive asserts the SDI ATTENTION signal without asserting the READ/WRITE READY signal, indicating the unsuccessful completion of a seek.

See retry/recovery section for recovery details.

04002 CZUDC SFT ERR 04002 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ATTN ASSERTED UNEXPECTEDLY, ASYN DRIVE ERROR OR LOGGABLE
INFORMATION -- THIS IS AN <<UNCOUNTED>> SOFT ERROR
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

This is an asynchronous drive error. Asynchronous drive errors are those errors reported by the drive which are not related to a level 2 command. These errors are reported by the drive using the SDI ATTENTION signal. The operator must look at the status returned to determine the error that occurred.

See retry/recovery section for recovery details.

04003 CZUDC SFT ERR 04003 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
SEEK DID NOT COMPLETE, NEITHER ATTN OR R/W RDY WAS ASSERTED
BEFORE TIMEOUT
SEEK FROM GRP group CYL cylinder TO GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

This error occurs when the drive fails to assert READ/WRITE READY before the seek timeout, which indicates the successful completion of a seek.

See retry/recovery section for recovery details.

04004 CZUDC HRD ERR 04004 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
RCT AREA CORRUPTED, COULD NOT FIND REPLACEMENT FOR
LBN THAT WAS REVECTORED
ATTEMPTING TO READ RCT LBN bn
SEARCHING FOR LBN bn

CZUDC HRD ERR 04004 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
RCT AREA CORRUPTED, COULD NOT FIND REPLACEMENT FOR
LBN WITH HEADER NOT FOUND
ATTEMPTING TO READ RCT LBN bn
SEARCHING FOR LBN bn

Error 4004 will occur only when Test 4 is running in the customer data area. It occurs when 1) A sector is either marked revectorred or the header can't be found in two revolutions of the disk (both cases should be revectorred) and 2) The replacement for that sector isn't found in the RCT and 3) a NULL entry isn't found at the end of the RCT (see DEC STANDARD 166, Replacement and Caching Table Format). In either case, the subunit should be reformatted, and the cause of the RCT corruption determined.

04005 CZUDC HRD ERR 04005 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
HEADER NOT FOUND DURING WRITE
DBN bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder

Error 4005 occurs only when Test 4 is writing a DBN or RBN. This is because bad blocks in the diagnostic area are not revectorred, and RBN's are what LBN's are revectorred to, so they should never be bad. Test 4 reports this error if the header being searched for couldn't be found in two revolutions of the disk.

04006 CZUDC SFT ERR 04006 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
SELECT TRACK AND WRITE LEVEL 1 CMD NOT EXECUTED
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

Select track and read or write not executed occurs when the UDA attempts to send the select track and read/write level 1 cmd, but receiver ready is deasserted or the state is invalid so it cannot send the command (the SERDES could also be broken so it's unable to send the command). The same error is generated if the UDA gets a header sync timeout, and when it looks at the drive's state, it is either invalid or receiver ready is deasserted (header sync timeout is <<NOT>> a error -- it's quite normal on a high-density disk).

See retry/recovery section for recovery details.

04007 CZUDC SFT ERR 04007 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ECC DETECTED ERROR
RETRY retry
ERROR RECOVERY LEVEL level
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder

Error 4007 occurs if an ECC error is detected but ECC correction is disabled.

See retry/recovery section for recovery details.

04008 CZUDC SFT ERR 04008 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ECC DETECTED ERROR, BUT CORRECTION FAILED
RETRY retry
ERROR RECOVERY LEVEL level
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder

Error 4008 occurs if an ECC error is detected, but the correction algorithm is unable to correct the errors.

NOTE: THIS IS USUALLY (BUT NOT ALWAYS) INDICATIVE OF A BAD SPOT IN THE ECC RESIDUE AREA AFTER THE DATA AREA OF THE SECTOR.

See retry/recovery section for recovery details.

04009 CZUDC SFT ERR 04009 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ECC CORRECTIONS EXCEED THRESHOLD
RETRY retry
ERROR RECOVERY LEVEL level
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder

Error 4009 occurs if an ECC error is detected, the correction algorithm succeeds in correcting the errors, but the number of bits that were corrected exceeds the correction threshold (a SDI DRIVE CHARACTERISTIC).

See retry/recovery section for recovery details.

04010 CZUDC SFT ERR 04010 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ECC CORRECTION SUCCEEDED, BUT EDC DETECTS ERROR
RETRY retry
ERROR RECOVERY LEVEL level
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder

Error 4010 could be caused by several problems:

1) A buffer with a few ECC errors that can be corrected, but the EDC was incorrectly computed or written, or 2) The ECC algorithm incorrectly corrected the buffer and/or the EDC value, (but corrections were less than the threshold) or 3) UDA buffer RAM problem.

See retry/recovery section for recovery details.

04011 CZUDC HRD ERR 04011 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ERROR RECOVERY TRIED ALL LEVELS WITHOUT SUCCESS
type bn
GRP group CYL cylinder

Error 4011 occurs when retries are enabled, and Test 4 has tried all retries on all levels of error recovery. See ECC and EDC retries in the retry/recovery section.

```
04012 CZUDC HRD ERR 04012 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
DATA COMPARISON FAILED
ECC OR EDC HAD DETECTED ERROR IN BUFFER
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
PATTERN NUMBER pattern
OFFSET OF ERROR WITHIN BUFFER: buffer_offset
OFFSET OF ERROR WITHIN DISPLAYED LIST: list_offset (1ST WORD OFFSET 0)
  data0 data1 data2 data3 data4 data5
  data6 data7 data8 data9 data10 data11
```

```
CZUDC HRD ERR 04012 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
DATA COMPARISON FAILED
ECC OR EDC HAD <<NOT>> DETECTED ERROR IN BUFFER
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
PATTERN NUMBER pattern
OFFSET OF ERROR WITHIN BUFFER: buffer_offset
OFFSET OF ERROR WITHIN DISPLAYED LIST: list_offset (1ST WORD OFFSET 0)
  data0 data1 data2 data3 data4 data5
  data6 data7 data8 data9 data10 data11
```

- pattern: The pattern number (decimal) that failed the comparison.
- buffer_offset: The offset of the error (decimal) within the sector read, where the first word in the sector is offset 0
- list_offset: The offset of the error (decimal) within the displayed list, where the first word in the list is offset 0
- dataX: Test 4 displays twelve data words read from the sector. They are displayed left to right, top to bottom.

Error 4012 occurs when a data compare detects a difference between the buffer read and a known data pattern. The operator is informed if the error was detected by the ECC or EDC. The first word of the sector which may or may not be printed, depending on the position of the error, is the pattern number replicated in each nibble of the word. If a disk is not initially written, it is likely that data comparison failures will occur in the first word of the sector. The following is the first word of the sector for the sixteen different patterns.

pattern	word 0	pattern	word 0
1	010421	9	114631
2	021042	10	125252
3	031463	11	135673
4	042104	12	146314
5	052525	13	156735
6	063146	14	167356
7	073567	15	177777
8	104210	16	000000

Note that pattern 16 is mapped to pattern 0.

04013 CZUDC DEV FTL ERR 04013 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
DRIVE NOT ONLINE TO UDA, AND NOT SPINABLE

If a drive drops offline while being tested (a normal occurrence during Test 4) and some event happens that makes the drive unspinnable (such as the operator popping out the run/stop switch) error 4013 will be printed. If the operator inhibits dropping units, Test 4 will go into error recovery and loop on error 4023, spindle dropped ready.

04014 CZUDC DEV FTL ERR 04014 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
UNABLE TO COMPLETE SEEK -- TRIED 3 TIMES
type bn
GRP group CYL cylinder

Once a seek has been attempted 3 times, and never successfully completed, error 4014 will be printed and the entire unit dropped. If the operator inhibits dropping units, the drive will be recalibrated, and the seek will be attempted again.

04015 CZUDC SFT ERR 04015 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
SEEK REQUIRED retries RETRIES BEFORE COMPLETING
GRP group CYL cylinder

retries: The number of times the seek was re-issued

If a seek required retries, error 4015 would print to notify the operator.

04016 CZUDC DEV FTL ERR 04016 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ERRORS DURING DRIVE INITIALIZATION AND SETUP
THIS UDA AND ALL DRIVES ATTACHED WILL BE REMOVED FROM TESTING

If any errors occur during drive and test initialization, DRIVES ATTACHED TO THE UDA THAT HAD THE DRIVE INITIALIZATION ERRORS WILL NOT BE TESTED. In this case, error 4016 will be printed to notify the operator. THIS ERROR DOES <<NOT>> REFER TO UDA INITIALIZATION. This error is unaffected by the operator inhibiting the dropping of units.

04017 CZUDC DEV FTL ERR 04017 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
NO VALID STATE FROM DRIVE
NO DRIVE CLOCKS

CZUDC DEV FTL ERR 04017 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
NO VALID STATE FROM DRIVE
HARD PARITY OR PULSE ERROR FOR 1/2 A SECOND

If Test 4 is <<EVER>> unable to get valid drive state, the drive is immediately dropped, and error 4017 is printed. There are two types of invalid state: no clocks or 'hard' errors. If Test 4 <<EVER>> detects no clocks, the driver is dropped IMMEDIATELY. Parity and pulse errors are normal, so Test 4 tolerates them, <<UNLESS THEY HAPPEN CONTINUOUSLY FOR 1/2 A SECOND>>. If they do occur for 1/2 a second, either the transmitter or receiver is bad, and the drive is dropped. If the operator has inhibited the dropping of units, Test 4 will retry the module that the error occurred on.

04018 CZUDC DEV FTL ERR 04018 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ATTEMPT TO WRITE ON WRITE PROTECTED DRIVE
ERROR CODE RETURNED FROM UDA: code
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

code: The error (in octal) returned to Test 4 from the UDA when Test 4 attempted to write on the write protected drive.

The UDA error codes (in octal) are as follows:

code	error
2	SELECT TRACK AND WRITE LEVEL 1 CMD NOT EXECUTED
3	LBN IS REVECTORED
4	HEADER NOT FOUND
153	SEEK OR HEAD SELECT ERROR
213	R/W RDY DROPPED
253	DATA OR STATE CLOCK TIMEOUT
313	RCVR RDY DROPPED
413	REAL TIME STATE RECEIVE ERROR

If an attempt is made to write on a write protected drive, the drive <<SHOULD>> drop READ/WRITE READY -- this is an error code of 213. If <<ANY>> other code is returned from the drive, the drive is causing the write to fail in an incorrect manner.

If Test 4 attempts to write on a write protected drive, error 4018 is printed. Test 4 requires the drive to detect the attempt to write when write protected and return an error for this error to be printed. If the operator has inhibited the dropping of units, a seek will be issued and the write attempted again.

04019 CZUDC HRD ERR 04019 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
HEADER NOT FOUND DURING READ
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder

Error 4019 occurs only when Test 4 is reading a DBN or RBN. This is because bad blocks in the diagnostic area are not revectorred, and RBN's are what LBN's are revectorred to, so they should never be bad. Test 4 reports this error if the header being searched for couldn't be found in two revolutions of the disk.

04020 CZUDC SFT ERR 04020 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
SELECT TRACK AND READ LEVEL 1 CMD NOT EXECUTED
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

Select track and read or write not executed occurs when the UDA attempts to send the select track and read/write level 1 cmd, but receiver ready is deasserted or the state is invalid so it cannot send the command (the SERDES could also be broken so it's unable to send the command). The same error is generated if the UDA gets a header sync timeout, and when it looks at the drive's state, it is either invalid or receiver ready is deasserted (header sync timeout is <<NOT>> a error -- it's quite normal on a high-density disk).

See retry/recovery section for recovery details.

04021 CZUDC DEV FTL ERR 04021 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
DRIVE NOT FORMATTED IN 512 BYTE MODE -- UNABLE TO TEST
XBN 0 MODE WORD: mode

mode: The mode word found on the drive's XBN 0

Error 4021 occurs only when Test 4 is going to test in the customer data area, and the mode word found in XBN 0 is not the 512 byte mode word (126736 octal). See DEC STANDARD 166 'FCT Structure'. Inhibiting the dropping of units has no effect on this error.

04023 CZUDC DEV FTL ERR 04023 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
UNABLE TO CONTINUE TESTING
PORT SWITCH OUT
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

If, during testing, the operator disables the port that Test 4 is using by popping out the port switch, Test 4 prints error 4023. CHANGING THE STATE OF THE PORT SWITCH FOR THE PORT THAT Test 4 IS <<NOT>> USING HAS NO EFFECT ON THE TEST. If dropping of units is inhibited, Test 4 will loop in error recovery, printing this error, until the error state is corrected (by some external action).

CZUDC DEV FTL ERR 04023 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
UNABLE TO CONTINUE TESTING
RUN/STOP SWITCH OUT
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

If, during testing, the operator pops out the run/stop switch, Test 4 prints error 4023. If dropping of units is inhibited, Test 4 will loop in error recovery, printing this error, until the error state is corrected (by some external action).

CZUDC DEV FTL ERR 04023 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxx'xx DRIVE xxx RUNTIME hh:mm:ss
UNABLE TO CONTINUE TESTING
SPINDLE DROPPED READY
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

If, during testing, the spindle drops from its ready state, error 4023 is printed. If dropping of units is inhibited, Test 4 will loop in error recovery, printing this error, until the error state is corrected (by some external action).

04024 CZUDC SFT ERR 04024 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
EDC DETECTED ERROR BUT ECC DID NOT
RETRY retry
ERROR RECOVERY LEVEL level
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
EDC COMPUTED edc EDC READ edc

edc: The edc computed and read in octal.

Error 4024 could be caused by several problems. 1) A buffer with no ECC errors, but the EDC was incorrectly computed or written, or 2) UDA buffer RAM problem, or 3) The error is such that the ECC really doesn't detect an error... This is unlikely.

See retry/recovery section for recovery details.

04025 CZUDC HRD ERR 04025 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
WRITE ATTEMPTED MAXIMUM TIMES
type bn

If three I/O errors occur when attempting to write to the drive (one I/O error if retries are disabled) error 4025 is printed to inform the operator.

04026 CZUDC HRD ERR 04026 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
READ ATTEMPTED MAXIMUM TIMES
type bn

If three I/O errors occur when attempting to read from the drive (one I/O error if retries are disabled) error 4026 is printed to inform the operator.

04028 CZUDC DEV FTL ERR 04028 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
BOTH READ ONLY <AND> WRITE ONLY BITS SET -- HOST ERROR

Error 4028 prints ONLY IF THERE IS A HOST CODE ERROR -- THIS IS NOT AN ERROR FROM A DRIVE. Inhibiting the dropping of units has no effect on this error.

04033 CZUDC DEV FTL ERR 04033 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
UNABLE TO CORRECTLY READ OVERLAY overlay_number
THIS UDA AND ALL DRIVES ATTACHED WILL BE REMOVED FROM TESTING

overlay_number: The overlay number in octal that could not be read.

Because of Test 4's size, most of the program is stored in host memory and is overlay driven. If any error is detected during a UNIBUS read of an overlay, Test 4 will retry the read (with no error report). It will attempt to read an overlay three times before error 4033 is printed, and the test immediately halted. This error can have several causes: 1) the UNIBUS died (it's improbable that you even get the message in this case) or 2) the UDA's UNIBUS interface died (also unlikely that you get a message), or 3) the host program wiped out the Test 4 overlays (since they are stored in host memory - most likely) or 4) a host memory problem - also likely. Inhibiting the dropping of units has no effect on this error.

04034 CZUDC SFT ERR 04034 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
SERDES OVERRUN ERROR DURING READ
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The SERDES overrun error is detected on a read operation and is indicative of a drive whose transfer rate is greater than 23 MHZ or a broken SERDES.

See retry/recovery section for recovery details.

04035 CZUDC SFT ERR 04035 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
DATA OR STATE CLOCK TIMEOUT DURING READ
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The loss of drive clock occurs when the UDA is clocking data to or from the drive through the SERDES. Failure of a word to be clocked in during a 125 millisecond time period triggers a loss of drive clock error.

See retry/recovery section for recovery details.

04036 CZUDC SFT ERR 04036 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
DATA SYNC TIMEOUT DURING READ
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

This error occurs on a read operation after the correct header has been found and the UDA times out waiting for the data sync word.

See retry/recovery section for recovery details.

04037 CZUDC SFT ERR 04037 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
R/W RDY DROPPED DURING READ
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The loss of read/write ready error is detected either before an I/O has begun when trying to send out the real time command or at the end of an I/O operation when checking for errors.

See retry/recovery section for recovery details.

04038 CZUDC SFT ERR 04038 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
RCVR RDY DROPPED DURING READ
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The loss of drive receiver ready is detected when the UDA is trying to send out a real-time read or write command.

See retry/recovery section for recovery details.

04040 CZUDC HRD ERR 04040 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ALL COPIES OF RCT READ WITH ERROR, SEARCHING FOR
LBN THAT WAS REVECTORED
LAST RCT LBN SEARCHED bn
SEARCHING FOR LBN bn

CZUDC HRD ERR 04040 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
ALL COPIES OF RCT READ WITH ERROR, SEARCHING FOR
LBN WITH HEADER NOT FOUND
LAST RCT LBN SEARCHED bn
SEARCHING FOR LBN bn

Error 4040 occurs when Test 4 is trying to find the RBN that replaces a LBN that was revectorized or whose header could not be found (both should be revectorized). Test 4 was unable to get a valid copy out of the M copies of the RCT due to I/O errors or ECC/EDC errors. M is a SDI DRIVE CHARACTERISTIC and is defined by the drive. This is indicative of either a bad pack (HDA) or that something wrote over the RCT incorrectly. Try to reformat the subunit.

04041 CZUDC HRD ERR 04041 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
COULD NOT FIND REPLACEMENT FOR
LBN THAT WAS REVECTORED
LBN TO REPLACE bn

CZUDC HRD ERR 04041 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
COULD NOT FIND REPLACEMENT FOR
LBN WITH HEADER NOT FOUND
LBN TO REPLACE bn

Error 4041 only occurs when Test 4 is running in the customer data area, and is trying to find the RBN that replaces a LBN that was revectorized (must be in the RCT) or whose header could not be found (should be in the RCT, unless the media under the header has 'grown' a bad spot recently). In either case, Test 4 was unable to find an entry in the RCT for the the sector and the subunit should be reformatted. In the case of the revectorized LBN, the cause of the RCT's corruption should be determined (even with the header not found, the RCT may have been corrupted because a header going bad without warning [eg. the formatter not being able to see it as a weak spot] is a very low probability occurrence).

04042 CZUDC DEV FTL ERR 04042 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
TIMEOUT WAITING FOR SECTOR OR INDEX PULSE
GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

Error 4042 occurs when the UDA microcode never detects a sector or index pulse from the drive before a read or write operation. If dropping of units is inhibited, a seek will be issued, and the write attempted again.

04044 CZUDC SFT ERR 04044 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
SEEK OR HEAD SELECT ERROR DETECTED DURING WRITE
ATTEMPT attempt
LBN bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

See error 4045 for description.

See retry/recovery section for recovery details.

04045 CZUDC SFT ERR 04045 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
SEEK OR HEAD SELECT ERROR DETECTED DURING READ
ATTEMPT attempt
LBN bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

Errors 4044 and 4045 occur when the header comparison routine determines that the drive is positioned at the wrong physical cylinder, or that the wrong head (which can be cylinders, groups or tracks, or any combination depending on the drive) had been selected. This error only occurs when the drive itself had not detected the misseek or incorrect head selected.

NOTE: These errors will only be detected when the operator is running Test 4 in the customer data area. This error will <<never>> appear when running in the diagnostic area.

See retry/recovery section for recovery details.

04047 CZUDC SFT ERR 04047 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
DATA OR STATE CLOCK TIMEOUT DURING WRITE
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The loss of drive clock occurs when the UDA is clocking data to or from the drive through the SERDES. Failure of a word to be clocked in during a 125 millisecond time period triggers a loss of drive clock error.

See retry/recovery section for recovery details.

04048 CZUDC SFT ERR 04048 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
R/W RDY DROPPED DURING WRITE
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The loss of read/write ready error is detected either before an I/O has begun when trying to send out the real time command or at the end of an I/O operation when checking for errors.

See retry/recovery section for recovery details.

04049 CZUDC SFT ERR 04049 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
RCVR RDY DROPPED DURING WRITE
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The loss of drive receiver ready is detected when the UDA is trying to send out a real-time read or write command.

See retry/recovery section for recovery details.

04050 CZUDC DEV FTL ERR 04050 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
BEGIN/END SET STARTING BLOCK NUMBER GREATER THAN ENDING BLOCK NUMBER

This is a Test 4 initialization error due to an operator error. Go back to the manual intervention questions and check the answers to the BEGIN/END set questions. Inhibiting the dropping of units has no effect on this error.

04051 CZUDC DEV FTL ERR 04051 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
THE BEGIN/END SETS OVERLAP

This is a Test 4 initialization error due to an operator error. Go back to the manual intervention questions and check the answers to the BEGIN/END set questions. Inhibiting the dropping of units has no effect on this error.

04052 CZUDC DEV FTL ERR 04052 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
BEGIN/END SET ENDING BLOCK NUMBER EXCEEDS MAXIMUM
MAXIMUM BLOCK NUMBER ON DEVICE IS maximum_block_number

maximum_block_number: This is the highest block number the operator can specify.

This is a Test 4 initialization error due to an operator error. Go back to the manual intervention questions and check the answers to the BEGIN/END set questions. Inhibiting the dropping of units has no effect on this error.

04053 CZUDC DEV FTL ERR 04053 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
DUPLICATE BAD BLOCKS

This is a Test 4 initialization error due to an operator error. Go back to the manual intervention questions and check the answers to the BAD BLOCK questions. Inhibiting the dropping of units has no effect on this error.

04054 CZUDC DEV FTL ERR 04054 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
BAD BLOCK NUMBER EXCEEDS MAXIMUM. MAXIMUM BLOCK NUMBER
ON DEVICE IS maximum_block_number

maximum_block_number: This is the highest block number the operator
can specify.

This is a Test 4 initialization error due to an operator error. Go back
to the manual intervention questions and check the answers to the
BAD BLOCK questions. Inhibiting the dropping of units has no effect
on this error.

04055 CZUDC DEV FTL ERR 04055 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
STARTING CYLINDER GREATER THAN ENDING CYLINDER

This is a Test 4 initialization error due to an operator error. Go back
to the manual intervention questions and check the answers to the
STARTING AND ENDING CYLINDER questions. Inhibiting the dropping of units
has no effect on this error.

04056 CZUDC DEV FTL ERR 04056 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
RANDOM AND SEQUENTIAL SEEKS CANNOT BE MIXED WITHIN A UNIT

Error 4056 is an operator error. The error occurs on a multiple subunit
drive when one subunit is selected to run in random mode, and another is
selected to run in sequential mode. This mix is not supported, so the
above message is issued. Inhibiting the dropping of units has no effect
on this error.

04057 CZUDC DEV FTL ERR 04057 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
OVERFLOW WHEN CALCULATING THE L/DBN FROM THE GIVEN CYLINDER
CYLINDER TOO LARGE

This is a Test 4 initialization error due to an operator error.
The operator entered a cylinder number, that when converted to a block
number, the block number exceeded $(2 \times 28) - 1$. Go back
to the manual intervention questions and check the answers to the
STARTING AND ENDING CYLINDER questions. Inhibiting the dropping of units
has no effect on this error.

04058 CZUDC DEV FTL ERR 04058 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
TRACK EXCEEDS MAXIMUM FOR DEVICE. MAXIMUM IS maximum_track

maximum_track: This is the highest track number the operator can specify.

This is a Test 4 initialization error due to an operator error. Go back to the manual intervention questions and check the answers to the TRACK questions. Inhibiting the dropping of units has no effect on this error.

CZUDC DEV FTL ERR 04058 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
GROUP EXCEEDS MAXIMUM FOR DEVICE. MAXIMUM IS maximum_group

maximum_group: This is the highest group number the operator can specify.

This is a Test 4 initialization error due to an operator error. Go back to the manual intervention questions and check the answers to the GROUP questions. Inhibiting the dropping of units has no effect on this error.

04059 CZUDC DEV FTL ERR 04059 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
TWO IDENTICAL TRACKS

This is a Test 4 initialization error due to an operator error. Go back to the manual intervention questions and check the answers to the TRACK questions. Inhibiting the dropping of units has no effect on this error.

CZUDC DEV FTL ERR 04059 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
TWO IDENTICAL GROUPS

This is a Test 4 initialization error due to an operator error. Go back to the manual intervention questions and check the answers to the GROUP questions. Inhibiting the dropping of units has no effect on this error.

04062 CZUDC DEV FTL ERR 04062 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
DBN COMPUTED FROM END CYLINDER GIVEN EXCEEDS MAXIMUM DBN NUMBER ON
DEVICE - CYLINDER TOO LARGE

This is a Test 4 initialization error.
Note that though there may be writeable DBN's on the 'last' cylinder,
the read only diagnostic area may start on that same cylinder, and Test 4
tries to write to the end of the cylinder that the operator specified.
Therefore, specify the previous cylinder if cylinders must be specified.
Inhibiting the dropping of units has no effect on this error.

CZUDC DEV FTL ERR 04062 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
OPERATOR ERROR IN ANSWERING MANUAL INTERVENTION QUESTIONS FOR THIS UNIT
LBN COMPUTED FROM END CYLINDER GIVEN EXCEEDS MAXIMUM LBN NUMBER ON
DEVICE - CYLINDER TOO LARGE

This is a Test 4 initialization error.
Note that though there may be writeable LBN's on the 'last' cylinder,
the RCT area may start on that same cylinder, and Test 4 tries to
write to the end of the cylinder that the operator specified. Therefore,
specify the previous cylinder if cylinders must be specified.
Inhibiting the dropping of units has no effect on this error.

04063 CZUDC SFT ERR 04063 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
REAL TIME STATE RECEIVE ERROR DURING WRITE
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The real time drive state receive error is detected at the end of an
I/O operation and indicates that there was a pulse or parity error
in the receipt of the drive's state during the I/O operation.

See retry/recovery section for recovery details.

04064 CZUDC SFT ERR 04064 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
REAL TIME STATE RECEIVE ERROR DURING READ
ATTEMPT attempt
type bn
SECTORS FROM INDEX sector TRK track GRP group CYL cylinder
ORIGIN OF SEEK: GRP group CYL cylinder
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

The real time drive state receive error is detected at the end of an I/O operation and indicates that there was a pulse or parity error in the receipt of the drive's state during the I/O operation.

See retry/recovery section for recovery details.

04068 CZUDC HRD ERR 04068 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
UNKNOWN ERROR CODE DURING WRITE
ERROR CODE RETURNED error_code
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

error_code: This is the error code returned to Test 4 by the UDA that Test 4 does not recognize.

The unknown error code occurs when the UDA returns an error code from an operation that Test 4 does not recognize. Possible UDA microcode change without Test 4 update.

See retry/recovery section for recovery details.

04069 CZUDC HRD ERR 04069 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
UNKNOWN ERROR CODE DURING READ
ERROR CODE RETURNED error_code
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

error_code: This is the error code returned to Test 4 by the UDA that Test 4 does not recognize.

The unknown error code occurs when the UDA returns an error code from an operation that Test 4 does not recognize. Possible UDA microcode change without Test 4 update.

See retry/recovery section for recovery details.

04070 CZUDC SFT ERR 04070 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
TIMEOUT OF SEND
command type
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

command_type: See section following error 4078 for a description

If Test 4 tries to send a level 2 command to the drive, and receiver ready is deasserted, error 4070 occurs.

See retry/recovery section for recovery details.

04071 CZUDC SFT ERR 04071 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
TIMEOUT OF RECEIVE
command type
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

command_type: See section following error 4078 for a description

This error is a failure of the drive to respond to an SDI level 2 command (see the SDI specification) before the drive-supplied command timeout expires.

See retry/recovery section for recovery details.

04072 CZUDC SFT ERR 04072 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
FIRST WORD RECEIVED WAS NOT START FRAME
command type
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

command_type: See section following error 4078 for a description

The first word received by the UDA from the drive was not a valid message start frame.

See retry/recovery section for recovery details.

04073 CZUDC SFT ERR 04073 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
FRAMING ERROR ON LEVEL 0 RECEIVE
command type
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

command_type: See section following error 4078 for a description

Error 4073 is caused by one or more of the following conditions:
1) Illegal frame code -- the frame is not a message start, continue,
or end frame. 2) Illegal sequence of frames -- such as a message
start frame without ever receiving a message end frame. This can be
caused by the drive sending a response before the UDA asserts receiver
ready, or a random hit on the SDI cable that garbles a frame or a bad
drive transmitter or UDA receiver.

See retry/recovery section for recovery details.

04074 CZUDC SFT ERR 04074 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
CHECKSUM ERROR ON LEVEL 0 RECEIVE
command type
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

command_type: See section following error 4078 for a description

The checksum attached to a message end frame did not match the checksum
computed over the level 2 command. This could be caused by a bad drive
transmitter, bad UDA receiver, incorrectly computed checksum by the
drive (unlikely) or a random hit on the SDI cable.

See retry/recovery section for recovery details.

04075 CZUDC SFT ERR 04075 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
BUFFER SIZE SMALLER THAN RESPONSE
command type
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

command_type: See section following error 4078 for a description

The buffer size set aside for the response was not large enough for the
response received. This is caused by the drive sending a response that
is incorrect for the request sent to the drive, or the drive sending some
garbage with the response.

See retry/recovery section for recovery details.

04076 CZUDC SFT ERR 04076 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
RESPONSE OF LEVEL 2 CMD NOT AS EXPECTED
command_type
EXPECTED RESPONSE expected_response
RESPONSE RECEIVED response_received
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

command_type: See section following error 4078 for a description
expected_response: This is the correct response (HEX) for the command.
response_received: This is the response received from the drive, (HEX)
where a 7D is an unsuccessful response. Any other
than a 7D for this value indicates a <<VERY>> sick
drive.

This is caused by receiving an UNSUCCESSFUL response from the drive, or
the drive sending some response other than the correct response for the
request sent to the drive. See the contents of status for the unexpected
response error (or reason).

See retry/recovery section for recovery details.

04077 CZUDC HRD ERR 04077 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
DRIVE NEVER DEASSERTED RECEIVER READY AFTER SEND
command_type
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

command_type: See section following error 4078 for a description

This is caused by the drive not seeing a command sent by
the UDA. The drive must deassert receiver ready to acknowledge
that it did see a command via the SDI. If the drive saw only
part of the command, it would have marked the command as
unsuccessful. But in this case, the drive did not see any
of the command and is now waiting for a command to be sent
from the UDA.

04078 CZUDC HRD ERR 04078 ON UNIT 00 TST 04 SUB 000 PC: xxxxxx
DISK EXERCISER DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hh:mm:ss
UNKNOWN ERROR CODE RETURNED FROM LEVEL 2 RECEIVE
command_type
ERROR CODE RETURNED error_code
REAL TIME STATE 0003
STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

command_type: See section following error 4078 for a description

error_code: This is the error code returned to Test 4 by the UDA that Test 4 does not recognize.

The unknown error code occurs when the UDA returns an error code from an operation that Test 4 does not recognize. Possible UDA microcode change without Test 4 update.

See retry/recovery section for recovery details.

NOTE: Errors 4070 - 4078 will become device fatals if attempted 3 times. If dropping of units are inhibited, error recovery is the same as if the error was a soft error.

command_type: in errors 4070-4078 command_type is one of the following level 2 commands:

- ATTEMPTING TO BRING DRIVE ONLINE
- ATTEMPTING TO ISSUE SEEK
- ATTEMPTING TO GET STATUS
- ATTEMPTING DRIVE CLEAR CMD
- ATTEMPTING TO BRING DRIVE ONLINE
- ATTEMPTING TO CHANGE MODE
- ATTEMPTING ERROR RECOVERY CMD
- ATTEMPTING TO ISSUE SEEK
- ATTEMPTING TO RECALIBRATE

The following commands types occur only during initialization, and will cause a device fatal if they occur. Inhibiting the dropping of units has no effect on these errors.

- ATTEMPTING TO SPIN UP DRIVE
- ATTEMPTING TO GET COMMON CHAR
- ATTEMPTING TO GET SUBUNIT CHAR

If <<ANY>> error occurs during initialization, <<NO>> testing is done on <<ANY>> drive attached to the UDA that the initialization error occurred on. See error number 4016.

3.2.9 SPECIAL DEVICE FATAL (05000)

05000 CZUDC DVC FTL 05000 ON UNIT 00 TST 002 SUB 000 PC: xxxxxx
DISK zzzzzzzz DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
UNABLE TO FIND REQUESTED DRIVE FOR TESTING
THE FOLLOWING IS VISIBLE ON THE PORTS
UDA PORT 0 -- description
UDA PORT 1 -- description
UDA PORT 2 -- description
UDA PORT 3 -- description

Where zzzzzzzz is either 'RESIDENT', 'FUNCION' or 'EXERCISER'.
This message is presented when the specified drive
was not found by test 2, test 3 or test 4 on any of
the ports. A description of what was each port follows.

NO DRIVE ATTACHED

- There is nothing on the port. If there is suppose to be a drive on this port, make sure there is an odd number of cables between the UDA and the drive and make sure the cables are properly attached.

RCVR RDY NEVER ASSERTED

- The device on the port did not assert RCVR RDY while trying to get state.

TIMEOUT OF SEND

- Sending an SDI command timed out. RCVR RDY is not asserted.

TIMEOUT OF RECEIVE

- Receiving an SDI command timed out. The drive failed to respond to an SDI level 2 command before a timeout expired.

FIRST WORD RECEIVED WAS NOT START FRAME

- The first word received by the UDA from the drive was not a valid message start frame.

FRAMING ERROR ON LEVEL 0 RECEIVE

- The device and the UDA are out of sync or an illegal frame code (the frame is not a message start, continue, or end frame) or illegal sequence of frames. This can be caused by the drive sending a response before the UDA asserts receiver ready, or a random hit on the SDI cable that garbles a frame or a bad drive transmitter or UDA receiver.

CHECKSUM ERROR ON LEVEL 0 RECEIVE

- The checksum attached to a message end frame did not match the checksum computed over the level 2 command. This could be caused by a bad drive transmitter, bad UDA receiver, incorrectly computed checksum by the drive (unlikely) or a random hit on the SDI cable.

RESPONSE LONGER THAN EXPECTED FOR CMD

- The buffer size set aside for the response was not large enough for the response received. This is caused by the drive sending a response that is incorrect for the request sent to the drive, or the drive sending some garbage with the response.

DRIVE n[, consecutive drive numbers if subunit drive] [further explanation]

- A drive was found at the end of the cable. It may be a subunit drive, so all the subunit numbers are printed. A further explanation may be presented. These further explanations are:

DRIVE NOT AVAILABLE TO THIS UDA

- The drive was found but is not available to this UDA. It may be dual ported and the drive is online to another controller.

UNSPINABLE DRIVE

- The drive is unspinnable. The drive may be powered up but the RUN/STOP switch may be popped out.

3.3 TEST 4 RETRY/RECOVERY METHODS

ECC Error on Disk Read

ECC DETECTED ERROR, BUT CORRECTION FAILED
ECC CORRECTIONS EXCEED THRESHOLD
ECC DETECTED ERROR (If ECC correction disabled)

Retry/Recovery - The UDA or Test 4 will first re-read the sector with the erroneous ECC N times, then N times for each level of error recovery the drive supports. The value of N is an SDI drive characteristic. This retry mechanism will persist until either the recovery level reaches zero or the operation succeeds. It should be noted that the manual intervention questions can disable retries (in this case the recovery fails the first time) and disable error correction (i.e., no ECC correction will be performed). ECC correction and retries are <<ALWAYS>> enabled when the Test 4 is reading the RCT.

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - Test 4 will issue a hard error for the sector. No soft errors will be counted.

Error Detecting Code (EDC) Error

EDC DETECTED ERROR BUT ECC DID NOT
ECC CORRECTION SUCCEEDED, BUT EDC DETECTS ERROR

This error is indicative of a UDA hardware error, either a SERDES failure or an undetected RAM failure, or a sector that was written with an incorrectly computed EDC.

Retry/Recovery - The UDA or Test 4 will re-read the sector with the erroneous EDC N times, then N times for each level of error recovery the drive supports. The value of N is an SDI drive characteristic. This retry mechanism will persist until either the recovery level reaches zero or the operation succeeds. It should be noted that the manual intervention questions can disable retries (in this case the recovery fails the first time). Retries are <<ALWAYS>> enabled when the Test 4 is reading the RCT.

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - Test 4 will issue a hard error for the sector. No soft errors will be counted.

SDI Level 2 and Asynchronous Errors

The SDI level 2 errors are as follows:

- o Packet acknowledge failure
- o Level 2 command error response, 'DE' bit set
- o Level 2 command error response, 'PE' or 'RE' bit set
- o Receipt of erroneous drive response
- o Seek complete timeout
- o Asynchronous drive errors

Level 2 errors are always retried, even if retries are disabled in the manual intervention questions.

In the following retry/recovery algorithms, the Test 4 'Generic error recovery' is the following steps:

1. Issue online command
 2. Get status
 - 2a. If the port, run or spindle ready (PS, RU or SR) bit is deasserted, an Immediate device fatal error is reported and the unit and all its subunits are dropped from testing.
 - 2b. If the recalibrate requested (RR) bit is set, Test 4 will issue a RECALIBRATE, then SEEK <<AFTER>> generic error recovery is complete.
 - 2c. If the drive error (DE) bit is set, Test 4 will issue a SEEK <<AFTER>> generic error recovery is complete.
 3. If no drive errors, go to 5
 4. Send DRIVE CLEAR command
 5. Change mode
- NOTE: If the drive's timeout expires once, so the drive asserts attention just to get Test 4 to issue a level 2, Test 4 will go through the above error recovery. However, since the timeout expiring is not an error, no error message is issued.

Packet Acknowledge Failure

TIMEOUT OF SEND
TIMEOUT OF RECEIVE

The timeout of send occurs when the UDA attempts to send a level 2 command to the drive, but the drive's receiver ready is not asserted. Timeout of receive is a failure of the drive to respond to an SDI level 2 command (see the SDI specification) before the drive-supplied command timeout expires. These errors are grouped together because their recoveries are the same.

Retry/Recovery - UDA - The steps listed below are performed.

1. The drive is initialized.
2. An SDI GET STATUS command is issued.
3. If the status obtained in the previous step indicated error conditions, these error conditions are resolved and then cleared by an SDI DRIVE CLEAR command.
4. An SDI SEEK command is issued.
5. The command is retried.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. The drive is initialized
2. Test 4 Generic error recovery is performed
3. An SDI SEEK command is issued.
4. The command is retried.

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - The above sequence will be repeated two times and, if the failure persists, the Test 4 will issue a device fatal error and the drive and all its subunits will be dropped. It should be noted that the retry strategy for SDI level 2 errors involves issuing additional level 2 commands. The retry count is the sum of all retries on all SDI level 2 commands, including those commands issued in recovery attempts.

Level 2 Command Error Response - 'DE' Bit Set

RESPONSE OF LEVEL 2 CMD NOT AS EXPECTED
SEEK RECEIVED UNSUCCESSFUL RESPONSE

An UNSUCCESSFUL response to a level 2 command, with the 'DE' bit set in the status response, notifies the Test 4 that a drive error was detected (or occurred) in connection with the execution of the SDI command.

Retry/Recovery - UDA - The steps listed below are performed.

1. An SDI GET STATUS command is issued.
2. The drive error is cleared by an SDI DRIVE CLEAR command and a SEEK command is issued for the cylinder where the drive was positioned when the error was reported.
3. The command is retried.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. Test 4 Generic error recovery is performed
Note that because the 'DE' bit is set, Test 4 generic error recovery will issue a SEEK (see generic error recovery)
2. The command is retried

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - The above sequence is repeated two times and, if the failure persists, the Test 4 will issue a device fatal error and the drive and all its subunits will be dropped.
Note that the
retry strategy for SDI level 2 errors involves issuing additional level 2 commands. The retry count is the sum of all retries on all SDI level 2 commands, including those commands issued in recovery attempts.

Level 2 Command Error Response - 'PE' or 'RE' Bit Set

RESPONSE OF LEVEL 2 CMD NOT AS EXPECTED
SEEK RECEIVED UNSUCCESSFUL RESPONSE

An UNSUCCESSFUL response to a level 2 command with the 'PE' or 'RE' bit set in the status response notifies the Test 4 that the command either was not appropriate for the state of the drive, or that the command contained invalid arguments.

Retry/Recovery - UDA - The steps listed below are performed.

1. An SDI GET STATUS command is issued
2. The drive error is cleared by an SDI DRIVE CLEAR command.
3. The controller verifies the state of the drive and, if possible, retries the level 2 command. Otherwise, the UDA notifies the host and bypasses subsequent retries.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. Test 4 Generic error recovery is performed
2. The command is retried

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - The above sequence is repeated two times and, if the failure persists, the Test 4 will issue a device fatal error and the drive and all its subunits will be dropped.

Note that the retry strategy for SDI level 2 errors involves issuing additional level 2 commands. The retry count is the sum of all retries on all SDI level 2 commands, including those commands issued in recovery attempts.

Receipt of an Erroneous Drive Response

FIRST WORD RECEIVED WAS NOT START FRAME
FRAMING ERROR ON LEVEL 0 RECEIVE
CHECKSUM ERROR ON LEVEL 0 RECEIVE
BUFFER SIZE SMALLER THAN RESPONSE
UNKNOWN ERROR CODE RETURNED FROM LEVEL 2 RECEIVE (hard error)

The first word not start frame error is caused when the UDA does not see a valid message start frame as the first frame received from the drive. The framing error is caused by the UDA receiving an illegal frame code -- the frame is not a message start, continue, or end frame or illegal sequence of frames -- such as a message start frame without ever receiving a message end frame. The checksum error occurs when a message end frame checksum did not match the checksum computed over the level 2 command. The buffer size smaller than response error occurs when the buffer set aside for the response was not large enough for the response received. The unknown error code is returned when the UDA returns an error code that the Test 4 does not recognize. These errors are grouped together because their recoveries are the same.

Retry/Recovery - UDA - The steps listed below are performed.

1. An SDI GET STATUS command is issued.
2. If the status obtained in the previous step indicated error conditions, these error conditions are resolved and then cleared by an SDI DRIVE CLEAR command.
3. The command is retried.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. Test 4 Generic error recovery is performed
2. The command is retried

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - The above sequence is repeated two times and, if the failure persists, the Test 4 will issue a device fatal error and the drive and all its subunits will be dropped. Note that the retry strategy for SDI level 2 errors involves issuing additional level 2 commands. The retry count is the sum of all retries on all SDI level 2 commands, including those commands issued in recovery attempts.

Seek Complete Timeout

ATTN ASSERTED DURING SEEK -- ERROR OR LOGGABLE INFORMATION
SEEK DID NOT COMPLETE, NEITHER ATTN OR R/W RDY WAS ASSERTED

This error occurs when the drive fails to assert READ/WRITE READY, indicating the successful completion of a seek, or asserts the SDI ATTENTION signal without asserting the READ/WRITE READY signal, indicating the unsuccessful completion of a seek.

Retry/Recovery - UDA - The steps listed below are performed.

1. An SDI GET STATUS command is issued.
2. If the status obtained in the previous step indicated error conditions, these error conditions are resolved and then cleared by an SDI DRIVE CLEAR command.
3. The SEEK is retried.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. Test 4 Generic error recovery is performed
2. The SEEK is retried

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - The above sequence is repeated two times and, if the failure persists, the Test 4 will issue a device fatal error and the drive and all its subunits will be dropped.

Note that the retry strategy for SDI level 2 errors involves issuing additional level 2 commands. The retry count is the sum of all retries on all SDI level 2 commands, including those commands issued in recovery attempts.

Asynchronous Drive Errors

ATTN ASSERTED UNEXPECTEDLY, ASYN DRIVE ERROR OR LOGGABLE INFORMATION -- THIS IS AN <<UNCOUNTED>> SOFT ERROR

Asynchronous drive errors are those errors reported by the drive which are not related to a level 2 or command. These errors are reported by the drive using the SDI ATTENTION signal. Examples are OFF CYLINDER and HDA OVERTEMPERATURE errors. Drive errors are reported to the controller by the 'DE' or 'WE' bit being set in the error byte in the status response.

Retry/Recovery - UDA - The steps listed below are performed.

1. An SDI GET STATUS command is issued.
2. The drive error is cleared by an SDI DRIVE CLEAR command and, if the error is not 'WE', a SEEK command is issued for the cylinder where the drive was last positioned.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. Test 4 Generic error recovery is performed
2. A SEEK is issued

NOTE: A 'WE' is a write on a write protected drive; Test 4 detects this in a different manner, so 'WE' will never be set.

Recovery Failure -

NOTE: There is a difference between the UDA in controller mode and the Test 4 for this type of error.

The UDA in controller mode will repeat the above sequence two times and, if the drive error persists, the drive would be marked as offline.

Test 4 will <<NOT>> drop the drive after two retries. Instead, the drive will be dropped due to a side affect of such an error: A seek never completing, (causing a device fatal error) or Spindle ready dropping (causing a device fatal error).

Drive I/O Errors

The drive I/O errors occur either during the header compare process (i.e., before I/O actually begins) or during the I/O operation itself. They are as follows:

- o Header not found
- o Seek or head select error
- o Data sync timeout
- o Data or state clock timeout during operation (read/write)
- o Receiver ready dropped during operation (read/write)
- o Read/write ready dropped during operation (read/write)
- o SERDES overrun error
- o Drive failed to execute select track and (read/write)
- o Real time state receive error

Header not found (header compare error)

HEADER NOT FOUND DURING (read/write)

This error occurs when the header compare routine fails to find the desired header (or a revectorized version of the desired header) in two disk revolutions.

Retry/Recovery - UDA and Test 4 - Failure to find the desired header in two rotations of the disk will cause the Test 4 to search the Replacement and Caching Table (RCT) to check if the logical block number has been replaced. If a match is found, the Test 4 will perform the desired operation on the revectorized block. Enabling/disabling retries has no affect on this operation.

Recovery success - No error is reported or counted.

Recovery Failure - A hard error (header not found) is reported.

Seek or head select error (Positioner Error)

SEEK OR HEAD SELECT ERROR DETECTED DURING (read/write)

This error occurs when the header comparison routine determines that the drive is positioned at the wrong cylinder and that the drive has not detected a seek error.

NOTE: The header comparison routine is active <<ONLY>> in the customer data area. This error will never be detected in the diagnostic area.

Retry/Recovery - UDA - The steps listed below are performed.

1. An SDI GET STATUS command is issued.
2. If the status obtained in the previous step indicated error conditions, these error conditions are resolved and then cleared by an SDI DRIVE CLEAR command.
3. An SDI RECALIBRATE command is issued.
4. An SDI SEEK command is issued.
5. The I/O operation is retried.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. Test 4 Generic error recovery is performed
2. An SDI RECALIBRATE command is issued.
3. An SDI SEEK command is issued.
4. If retries are disabled, Immediate recovery failure. Retries are <<ALWAYS>> enabled when the Test 4 is reading the RCT.
5. The I/O operation is retried.

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - The above sequence is repeated two times and, if a drive I/O error persists, a hard error is reported for the sector. No soft errors are counted.

Data Sync Timeout Error

DATA SYNC TIMEOUT DURING READ

This error occurs on a read operation after the correct header has been found and the UDA times out waiting for the data sync word.

Retry/Recovery - UDA - The steps listed below are performed.

1. An SDI GET STATUS command is issued.
2. If the status obtained in the previous step indicated error conditions, these error conditions are resolved and then cleared by an SDI DRIVE CLEAR COMMAND.
3. An SDI SEEK command is issued.
4. The read operation is retried.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. Test 4 Generic error recovery is performed
2. An SDI SEEK command is issued.
3. If retries are disabled, Immediate recovery failure. Retries are <<ALWAYS>> enabled when the Test 4 is reading the RCT.
4. The read operation is retried.

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - The above sequence is repeated two times and, if a drive I/O error persists, a hard error is reported for the sector. No soft errors are counted.

Data or state clock timeout (Loss of Drive Clock)
Receiver ready failure (Loss of Drive Receiver Ready)

DATA OR STATE CLOCK TIMEOUT DURING (read/write)
RCVR RDY DROPPED DURING (read/write)
COULD NOT SEND SELECT TRACK AND (read/write) CMD OR
HEADER SYNC TIMEOUT WITH INVALID STATE

The loss of drive clock occurs when the UDA is clocking data to or from the drive through the SERDES. Failure of a word to be clocked in during a 125 millisecond time period triggers a loss of drive clock error. The loss of drive receiver ready is detected when the UDA is trying to send out a real-time read or write command. Unable to select track and read or write occurs when the UDA attempts to send the select track and read/write level 1 cmd, but receiver ready is deasserted or the state is invalid so it cannot send the command (the SERDES could also be broken so it's unable to send the command). The same error is generated if the UDA gets a header sync timeout, and when it looks at the drive's state, it is either invalid or receiver ready is deasserted (header sync timeout is <<NOT>> a error -- it's quite normal on a high-density disk). These errors are grouped together because their recoveries are the same.

Retry/Recovery - UDA - The steps listed below are performed.

1. The drive is initialized.
2. An SDI GET STATUS command is issued.
3. If the status obtained in the previous step indicated error conditions, these error conditions are resolved and then cleared by an SDI DRIVE CLEAR command.
4. An SDI SEEK command is issued.
5. The I/O operation is retried.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. The drive is initialized
2. Test 4 Generic error recovery is performed
3. An SDI SEEK command is issued.
4. If retries are disabled, Immediate recovery failure. Retries are <<ALWAYS>> enabled when the Test 4 is reading the RCT.
5. The I/O operation is retried.

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - The above sequence is repeated two times and, if a drive I/O error persists, a hard error is reported for the sector. No soft errors are counted.

Read/Write ready dropped (Loss of Drive Read/Write Ready)
SERDES Overrun Error
Real Time State Receive Error (Real Time Drive State Receive Error)

R/W RDY DROPPED DURING (read/write)
SERDES OVERRUN ERROR DURING READ
REAL TIME STATE RECEIVE ERROR DURING (read/write)
UNKNOWN ERROR CODE DURING (read/write)

The loss of read/write ready error is detected either before an I/O has begun when trying to send out the real time command or at the end of an I/O operation when checking for errors. The SERDES overrun error is detected on a read operation and is indicative of a drive whose transfer rate is greater than 23 MHZ or a broken SERDES. The real time drive state receive error is detected at the end of an I/O operation and indicates that there was a pulse or parity error in the receipt of the drive's state during the I/O operation. The unknown error code is returned when the UDA returns an error code that the Test 4 does not recognize. They are grouped together because their recoveries are the same.

Retry/Recovery - UDA - The steps listed below are performed.

1. An SDI GET STATUS command is issued.
2. If the status obtained in the previous step indicated error conditions, these error conditions are resolved and then cleared by an SDI DRIVE CLEAR command.
3. An SDI SEEK command is issued.
4. The I/O operation is retried.

Retry/Recovery - Test 4 - The steps listed below are performed.

1. Test 4 Generic error recovery is performed
2. An SDI SEEK command is issued.
3. If retries are disabled, Immediate recovery failure. Retries are <<ALWAYS>> enabled when the test 4 is reading the RCT.
4. The read operation is retried.

Recovery success - One soft error is counted for the entire operation including retries.

Recovery Failure - The above sequence is repeated two times and, if a drive I/O error persists, a hard error is reported for the sector. No soft errors are counted.

3.4 DEC STANDARD 166 EXCERPTS

3.4.1 THE REPLACEMENT AND CACHING TABLES

The Replacement and Caching Tables record the locations of all revectored LBN sectors and the status of each RBN on the unit. Each copy of the table is organized in ascending RBN order, with an entry for each RBN sector on the unit. There are 'n' copies of the table on the unit, where 'n' is a device characteristic. The tables are stored at the high address end of the LBN area of the unit. Table entries (and RBNs) are allocated via a hash algorithm described later.

Replacement And Caching Table Format -

Each entry in the Replacement and Caching Table represents an RBN on the unit. The table is ordered in ascending RBN order. Thus the first entry corresponds to the first RBN on the unit, etc. The size of each copy of the table may exceed that required to contain an entry for each RBN on the unit since additional entries may be required to align the table so that adjacent copies can begin on a track boundary. Entries that do not correspond to RBNs on the unit are called 'null entries'; there is always at least one null entry at the end of the RCT. All other entries past this last null entry are undefined.

NOTE

The RCT pad area is controller specific and should never be accessed by the host.

The format of a replacement block descriptor in the Replacement and Caching Tables is:

```
!<-----16 bits----->!
+-----+
!               LBN (low)               !
+-----+
!  CODE !               LBN (high)       !
+-----+
! 4 bits!<-----12 bits----->!
```

Where:

LBN is the Logical Block Number of a revectored LBN sector.

CODE is one of the following octal values:

00 - Unallocated (empty) replacement block.

02 - Allocated replacement block - primary RBN.

03 - Allocated replacement block - non-primary RBN.

04 - Unusable replacement block.

* 05 - Alternate unusable replacement block

10 - Null entry - no corresponding RBN sector.

For codes 00, 04, and 10 the LBN field is always zero.

NOTE

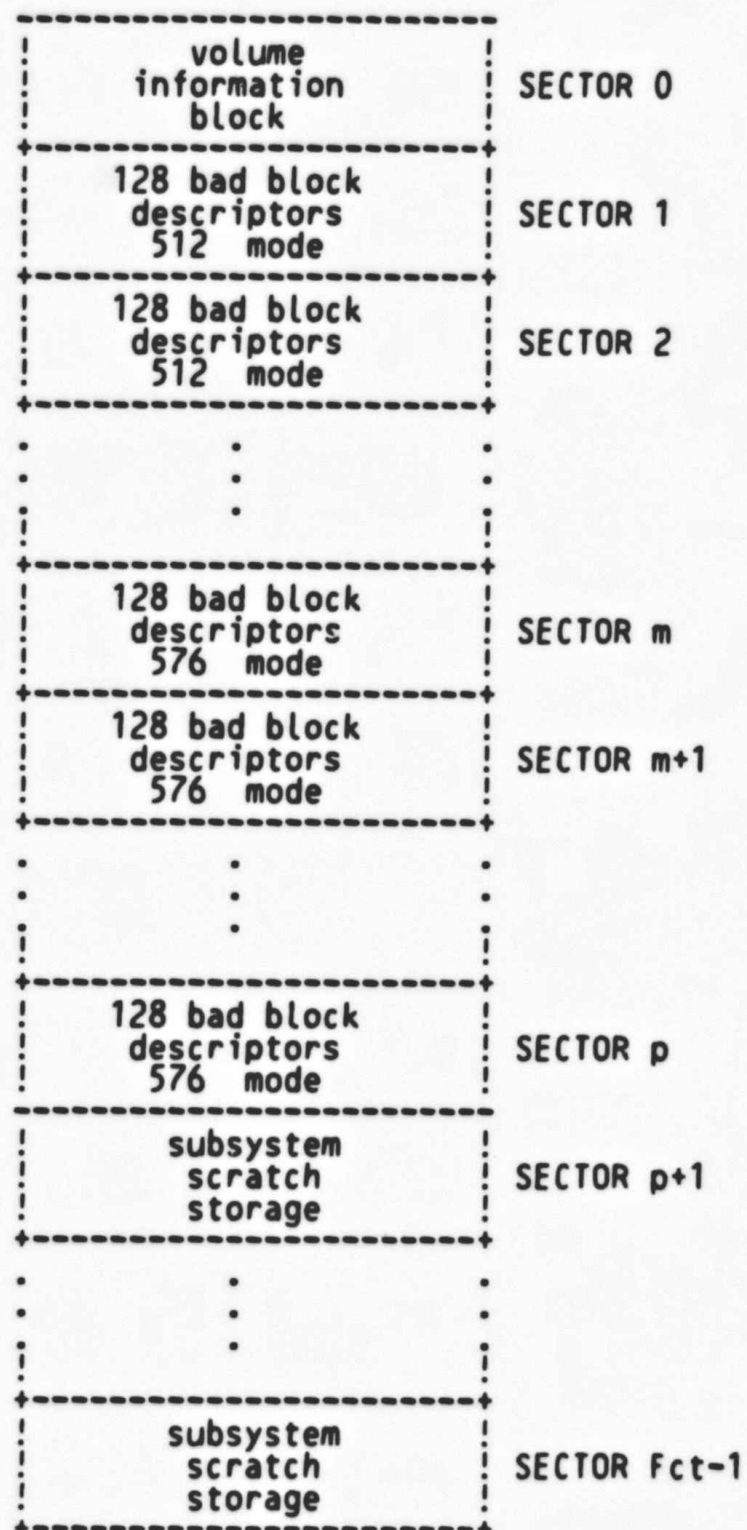
* This code is reserved. Programs should treat this code as if it were code 04.

Embedded-controllers with no distinction between primary and secondary RBN's must use:

1. Code 02 if the replacement block can be retrieved with little degradation of performance for all blocks.
2. Code 03 if accessing the replacement block has a large impact on performance for all blocks.

3.4.2 FCT Structure

Each copy of the FCT is composed of one volume information block, one 512 byte format table, one 576 byte format table, and one subsystem temporary storage area (distributed amongst the alignment pads). An FCT copy has the following format:



The XBN area itself is always formatted to contain 512 byte sectors. The calculations for m and p are:

$$m := (((Lc * g * t * r) + 1) / 2) + 127 / 128$$

$$p := 2 * m$$

Sector 0 contains various volume identification information. The format is:

media mode	WORD 0
formatting instance number	WORD 1
volume serial number least significant word	WORD 2
volume serial number	WORD 3
volume serial number	WORD 4
volume serial number most significant word	WORD 5
date that volume was first formatted (low)	WORD 6
date that volume was first formatted	WORD 7
date that volume was first formatted	WORD 8
date that volume was first formatted (high)	WORD 9
date of most recent volume formatting (low)	WORD 10
date of most recent volume formatting	WORD 11
date of most recent volume formatting	WORD 12
date of most recent volume formatting (high)	WORD 13
number of used entries in 512 table (low)	WORD 14

number of used entries in 512 table (high)	WORD 15
number of used entries in 576 table (low)	WORD 16
number of used entries in 576 table (high)	WORD 17
XBN of scratch area in this copy (low)	WORD 18
XBN of scratch area in this copy (high)	WORD 19
size of scratch area in this copy	WORD 20
zeros	
zeros	WORD 255

Where:

WORD 0: 'Media Mode' - is '126736' for a 512 byte format and '074161' for a 576 byte format. During formatting the media mode word is set to zero.

4.0 PERFORMANCE AND PROGRESS REPORTS

At the end of each pass, the pass count is given along with the total number of errors reported since the diagnostic was started. The "EOP" switch can be used to control how often the end of pass message is printed. Section 2.2 describes switches.

A statistical report will automatically be printed periodically (approximately every fifteen minutes) and at the end of test #4. It can be suppressed by setting the Inhibit Statistical Report flag (e.g. START/FLAGS:ISR). This is the same report that can be printed on demand with the PRINT command.

During tests 1, 2, and 3, the report will look like the following example:

TEST 1 IN PROGRESS RUN TIME 2:24:10

During test #4, the report will contain statistics on each drive for the current pass of the test; for example:

TEST 4 IN PROGRESS RUN TIME 2:24:10

UNIT	DRIVE	SERIAL-NUMBER	SEEKS X1000	MBYTES READ	MBYTES WRITTEN	HARD ERRORS	SOFT ERRORS	ECC
0	0	1002	12	36	22	0	0	1
1	4	7342102112	14	42	29	0	2	0

Explanation of each column:

- UNIT DRIVE The unit number (number of HW P-table).
- DRIVE The drive number (the number which appears on the "unit plug" on the front of the disk drive).
- SERIAL-NUMBER The decimal serial number of the disk drive.
- SEEKS X1000 The decimal number of seeks performed by this drive during this pass of test 4. Multiply value by 1000.
- MBYTES READ The number of mega-bytes (million bytes) read by this drive during this pass of test 4. It is this value that is used to optionally drop a drive by the READ TRANSFER LIMIT software question.
- MBYTES WRITTEN The number of mega-bytes written by this drive during this pass of test 4.
- HARD ERRORS The number of hard error reports printed for this drive during this pass of test 4. It is this value that is used to optionally drop a drive by the ERROR LIMIT software question.

SOFT ERRORS

The number of soft errors reported for the drive during this pass of test 4. A soft error is any error condition that resulted in a retry operation that eventually succeeded in recovering from the error condition. One soft error is counted even though several retry attempts may be made and does not correspond to the number of soft error reports printed. To see the soft error reports, you must change the default answer to the SUPPRESS PRINTING SOFT ERRORS software question.

ECC

The number of times data read from the drive was modified using the error correction code (ECC) and resulted in a matching error detection code (EDC).

5.0 TEST SUMMARIES

The UDA Host Resident Diagnostic consists of one PDP-11 diagnostic supervisor program that runs in the PDP-11 processor and four programs that run in the UDA's buffer memory through an interpreter called the "diagnostic machine" which resides in the UDA. The PDP-11 program mainly is responsible for downline loading the "diagnostic machine" programs into the UDA and starting their execution. The "diagnostic machine" program controls the testing from that point by requesting the PDP-11 processor to supply information, print error messages and update statistics. The "diagnostic machine" program informs the PDP-11 processor when a test is complete.

The four "diagnostic machine" programs are in the ZUDDBO.PAK data file which is read from the XXDP+ system device by the PDP-11 program. The data file comes with listings of each program.

5.1 TEST # 1 - UNIBUS ADDRESSING TEST

The purpose of test #1 is to complete the testing of the Unibus interface in the UDA. The UDA resident diagnostic is not able to completely test the Unibus interface because communication with the PDP-11 processor is necessary. Specifically, this test will:

1. Check that every address line on the Unibus can be driven to both one and zero states.
2. Check that the UDA can interrupt the PDP-11 processor at the proper priority level and vector.
3. Exercise the Unibus interface by transferring blocks of data to and from Unibus memory.

This test assumes that the following are being tested by the UDA Resident Diagnostic:

1. All data bits can be written and read correctly.
2. NPR cycles can be executed correctly.

Test 1 is divided into six subtests. One at a time, each UDA selected for testing will run each subtest.

Subtest 1 makes sure that the UDAIP and UDASA registers are existant and runs the first part of the UDA's resident diagnostics.

Subtest 2 initializes the UDA into diagnostic loop mode. In this mode any value written into the UDASA is echoed in the UDASA.

In subtest 3, the UDA is initialized with interrupts enabled. The vector address and priority level will be determined solely from the answers to the hardware questions. If the hardware vectors to the wrong address, it is impossible to determine the result. A descriptive error message of the problem will not occur (the program or processor may hang or an unrelated message may occur). Therefore, the message "TESTING INTERRUPT ABILITY OF UDA AT ADR xxxxxx VEC xxx..." is printed just before the UDA is requested to cause an interrupt and the word "COMPLETED" is printed (on the same line) when the interrupt test is completed. If the word "COMPLETED" does not follow the first message, it should be apparent that the interrupt caused the diagnostic or processor to go astray. The priority level of the interrupt request is also verified.

Subtest 4 and 5 initializes the UDA using different sizes of the host communications area. The different sizes of the host communications area are supplied to allow the UDA Resident Diagnostic to do the most Unibus address testing possible. Interrupts are disabled. Any UDA Resident Diagnostic errors will be reported. Subtest 4 initializes the UDA with the smallest ring buffer size possible. Subtest 5 initializes the UDA with a large ring buffer area.

Subtest 6 downline loads a "diagnostic machine" program into the UDA. The "diagnostic machine" program is downline loaded from the memory space included in the host communications area when the UDA was first initialized. The UDA Resident Diagnostic has already verified that it can access these memory addresses, so the downline load command should perform properly. The "diagnostic machine" program is then started.

The "diagnostic machine" program asks the PDP-11 program to fill free memory (that memory available to the PDP-11 program that is not being used by the program or the Runtime Services) with an addressing pattern and report the location and size of the free memory. Every location of free memory is read and the data checked. Then, one by one, each address line is tested as follows:

1. Determine a test address by taking the first address of free memory and complimenting the address bit to be tested.
2. Read from the test address.
3. If a non-existent memory error occurs, the test is complete.
4. Write all ones to the first address of free memory then read from the test address. If data read is not all ones, then test is complete.
5. Write zeros to the first address of free memory then read from the test address. If data read is not zeros, then test is complete.
6. Report Unibus addressing error.

When all address bits have been tested, then block transfers to and from memory are tested with different data patterns. This data is transferred at the rate disk data is transferred to and from memory during normal UDA operation.

The next UDA selected for testing is then be tested in the same manner. When all UDAs have been tested, test #1 ends.

5.2 TEST # 2 - DISK RESIDENT DIAGNOSTIC TEST

The purpose of test #2 is to execute the diagnostics that run in each disk drive. These diagnostic programs may be resident in the disk drive or require downline loading from the ZUDDBO.PAK data file. (There currently are no disk drives that require downline loading and no such files exist in the ZUDDBO.PAK file. This program is designed such that they can be easily added in a future release.) This UDA diagnostic program only knows the procedure to execute the disk resident diagnostics and how to determine whether a test passed or failed.

One at a time, each UDA selected for testing is initialized and a "diagnostic machine" program downline loaded. The "diagnostic machine" program asks what drives are to be tested, then issues several commands to the disk drive and check for the correct response from the drive. This should serve as a good indicator that the UDA and disk drive can communicate.

A DIAGNOSE command is then issued to the drive to request the drive run all of its diagnostics. If the disk drive requests a downline load of a drive diagnostic, the diagnostic program is read from the XXDP+ load device, downline loaded into the disk drive and started. There is no limit to the number of downline loads that can be requested by a drive.

If the "Manual Intervention Mode" software question was answered 'N' (default) testing proceeds to the next drive. When all drives on the UDA have been tested, the next UDA selected for testing is tested in the same manner. When all UDA's have been tested, test #2 ends.

If the "Manual Intervention Mode" software question was answered 'Y', an interactive mode is entered to allow the operator to perform diagnostic activities on the disk drive as desired. The Service Manual for the disk drive must be used to determine what diagnostic capabilities are available.

First, a brief description of available commands is printed as follows:

TEST #2 MANUAL INTERVENTION ON UNIT xx UDA AT xxxxxx DRIVE xxx
TO WRITE AND READ MEMORY:
W DATA REGION OFFSET
R REGION OFFSET
TO RUN A DIAGNOSTIC:
D REGION
TO EXIT QUESTIONING:
E
DATA, REGION AND OFFSET ARE HEX VALUES.
?

Commands may be typed after the question mark prompt. Each command is processed as entered and results displayed immediately. The exit command will allow the diagnostic to proceed.

Read and write commands remember the region and offset values. Successive read and successive write commands automatically increment to the next offset if the region and offset values are not typed. If a region is typed but not an offset, offset zero is used.

Examples:

1. W FF FFFC 4
2. W 02
3. R FFFC 4
FFFC 0004/ FF
4. R
FFFC 0005/ 02
5. W 21 FFFC
6. R
FFFC 0000/ 21

Command 1 writes one byte (FF) into region FFFC, offset 4. Command 2 writes one byte (02) into the next byte - region FFFC, offset 0005. Commands 3 and 4 read the bytes back. Command 5 writes one byte (21) into the first byte of region FFFC. Command 6 reads back that byte.

The diagnose command remembers the region from previous diagnose commands only, because the region containing the diagnostic is generally not the same region used to write parameters or read results. If the diagnostic returns any data, the data is printed immediately.

5.3 TEST # 3 - DISK FUNCTION TEST

The purpose of test #3 is to functionally test the disk drive. On a drive that is well diagnosed by its disk resident diagnostics (executed by test #2) these functional tests will have little value. On a drive that has no or minimal resident diagnostics, these functional tests will have more value.

Test #3 starts by initializing each UDA selected for testing and then downline loading a "diagnostic machine" program into each UDA. Once all UDAs have been started, the PDP-11 program responds to requests from all UDAs. When all the UDAs have indicated the end of testing, test #3 ends.

The "diagnostic machine" program performs the following functions on each drive:

1. Issue a DRIVE CLEAR command.
2. Issue RECALIBRATE command.
3. Issue a CHANGE MODE command to enable diagnostic cylinder access, set the drive to 512 byte sector size, and write protect.
4. Issue INITIATE SEEK command to last diagnostic cylinder.
5. Read all factory formatted sector headers. If no headers on a track can be read, report the error, otherwise continue.
6. Starting with cylinder 0, group 0 and incrementing through every cylinder on the disk, seek to a group, read a header on track 0 and then seek to the factory formatted diagnostic cylinder. Read from the diagnostic cylinder to verify disk positioned correctly.
7. Attempt to write on the first diagnostic cylinder while write protected.
8. Issue a CHANGE MODE command to enable formatting operations and disable write protect.
9. Format all writable DBNs in 512 byte format.
10. Write and read several data patterns to each writable DBN. Report an error if all DBNs on one track have an error.
11. Send invalid SDI level 2 and level 1 commands and check the results.
12. Issue a DISCONNECT command.

5.4 TEST # 4 - DISK EXERCISER

The purpose of test #4 is to exercise the disk drives in a manner similar to normal usage under standard operating systems. Execution of this test should give an indication of the performance of the disk drive. This test may be run for long or short periods of time, depending on how the software questions are answered.

These are two modes of operation for test #4:

1. Default operation on the entire area selected (customer or diagnostic) with all parameters selected for random operation as shown by default answers below.
2. Manual intervention mode where a number of questions are asked and operation is controlled by their answers.

Which mode is entirely determined by the answer to the first software question asking, "Enter manual intervention mode for special diagnosis?" This question would normally have been answered 'N' (default) and testing will begin immediately. If answered 'Y', the following series of questions will be asked for each unit selected for testing:

THE FOLLOWING QUESTIONS REFER TO UNIT xx UDA AT xxxxxx DRIVE xxx

This message will identify to which drive the questions are being asked. The entire series of questions will be asked for each drive, there is no short way to answer like in the hardware questions.

NUMBER OF BAD BLOCKS (D) 0 ?

An answer in the range of 1 to 16 will allow that many bad block numbers to be entered. The program will allow writes and reads to these blocks but no error messages will be printed for these blocks. Errors encountered on these blocks will not appear in the statistics. Answer zero to bypass entering bad blocks.

BAD BLOCK (A) ?

This question will be asked the number of times requested by the previous answer. Any decimal number that can be converted into a 28-bit binary value will be accepted. No other error checking will be made at this time to determine if the block number actually exists on the disk.

DO YOU WANT TO CHANGE TESTING PARAMETERS FOR THIS DRIVE (L) N ?

Answer 'N' to bypass all further questioning on this drive.
Answer 'Y' to be asked the following questions.

READ ONLY (L) N ?

Answer 'Y' to dictate read only and prevent test #4 from performing any writes to the disk.

WRITE ONLY (L) N ?

This question will only be asked if the previous question was answered 'N'. Answer 'Y' to dictate write only.

CHECK ALL WRITES BY READING (L) N ?

Answer 'Y' to cause all writes to be checked by reading the data immediately after the write operation.

RANDOMLY CHECK WRITES BY READING (L) Y ?

This question will only be asked if the previous question was answered 'N'. Answer 'Y' for the write check to be performed randomly. Answer 'N' if write checks are not desired. This question is asked no matter how previous questions were asked.

DATA PATTERN - 0 FOR RANDOM SELECTION (D) 0 ?

There are 16 data patterns available, selected as 1 to 16. Pattern number 0 will cause patterns 1 to 15 to be randomly selected for each write. If pattern number 16 is selected, the following set of questions will be asked for a pattern to be input.

ENABLE ECC DATA CORRECTION (L) Y ?

A 'Y' answer will enable the use of ECC to correct data errors. If the number of corrections is within the drive's threshold, an informational message will be printed identifying the block number. These ECC corrections will also appear in the statistical report for the drive.

An 'N' answer will prevent the use of ECC. All ECC errors will cause an error message to be printed and retries to be attempted.

COMPARE ALL DATA READ (L) N ?

Answer 'Y' to cause a data compare after every read.

RANDOMLY COMPARE DATA READ (L) Y ?

This question will only be asked if the previous question was answered 'N'. Answer 'Y' for the data compare to be performed on random records. Answer 'N' if data compares are not desired.

ENABLE RETRIES (L) Y

A 'Y' answer will enable retries to be performed on disk errors.

RANDOM ACCESS MODE (L) Y ?

Answer 'Y' to cause block numbers to be chosen randomly.
Answer 'N' to cause block numbers to be selected sequentially up and down the disk surface.

DO YOU WISH TO:

- 0 - TEST ENTIRE AREA SELECTED
 - 1 - SPECIFY BEGIN/END SETS TO TEST
 - 2 - SPECIFY TRACKS AND CYLINDERS TO TEST
 - 3 - SPECIFY GROUPS AND CYLINDERS TO TEST
 - 4 - SPECIFY CYLINDERS TO TEST
- (D) 0 ?

This question specifies the options available to limit testing to a portion of the selected area (customer or diagnostic) of the disk. A zero answer is the default which specifies to use the entire area for the test. Other answers will cause additional questions to be asked.

NUMBER OF BEGIN/END SETS (D) 1 ?
BEGIN BLOCK (A) 0 ?
END BLOCK (A) 0 ?

These questions are asked if begin/end sets were selected to limit the testing area (Answer 1). One to four sets may be specified. The BEGIN BLOCK and END BLOCK questions are asked as many times as needed.

NUMBER OF TRACKS TO TEST (D) 1 ?
TRACK (D) 0 ?

NUMBER OF GROUPS TO TEST (D) 1 ?
GROUP (D) 0 ?

One of these sets of questions is asked if either tracks and cylinders or groups and cylinders was specified to limit the testing area (Answers 2 or 3). Up to seven tracks or groups may be specified on which testing will be limited.

DO YOU WISH TO LIMIT THE CYLINDERS TESTED (L) N ?

This question is asked only after the tracks or groups have been specified above. If testing is to be further limited to a set of cylinders, answer 'Y' and the following two questions will be asked:

STARTING CYLINDER (A) 0 ?
ENDING CYLINDER (A) 0 ?

These questions are asked if the question immediately above was answered "Y" or if cylinders were selected to limit the testing area (Answer 4). One set of cylinder numbers may be specified to limit the testing area.

After the above questions have been asked for all drives selected for testing, the following questions will be asked if data pattern 16 was selected for any drive:

NUMBER OF WORDS IN DATA PATTERN 16 (D) 1 ?
DATA WORD (O) 0 ?

Data pattern 16 can be input by these questions. A data pattern consists of a buffer of one to 16 words which is repeated throughout the data portion of the disk block. Enter the contents of the data pattern buffer. The DATA WORD question will be repeated as needed.

Test #4 will then initialize each UDA selected for testing and downline load a "diagnostic machine" program into each UDA. The "diagnostic machine" program asks what drives are to be tested and then for the parameters for each drive (the answers to the manual intervention questions or their defaults). Once all UDAs have been started, the PDP-11 program responds to requests from all UDAs.

The disks are then be exercised according to the parameters. The exercise consists of selecting a disk sector, seeking to the proper cylinder, then reading or writing the sector. The parameters control how the disk sector is selected, whether the sector is written or read and whether a write is followed by a read (write check).

The "diagnostic machine" program periodically sends statistics to the PDP-11 program. These statistics include counts of reads, writes, seeks and errors on a per drive basis. The PDP-11 program accumulates the statistics from all the UDAs and watches for the transfer limit to be exceeded. As long as the error log is not enabled, the exceeding of the transfer limit will cause the end of test #4.

Each time an error occurs, the "diagnostic machine" tells the PDP-11 program. A message is printed (or stored in the log buffer) and then the error limit for the drive is checked. If the error limit has been reached, the drive is dropped from testing. If no more drives remain to be tested, test #4 will end (unless the error log is enabled).

When the end of test #4 occurs, the accumulated statistics for each drive is printed. This statistical report can be printed at any time during test #4 by typing control-C then the PRINT command.

The data patterns used by test #4 are indicated below. Each pattern is generated by writing the pattern number in each 4-bit nibble of the first word, then repeating the data pattern (sequence of one to 16 words) throughout the rest of the data buffer. Pattern number 16 writes nibbles of zeros. When pattern number zero is used, the actual pattern number written (1 to 15) is placed in the nibbles.

- PATTERN 0 This pattern number is used to indicate any pattern number 1 to 15 chosen at random.
- PATTERN 1 Words in pattern sequence - 1
Sequence (Octal) 105613
Sequence (Hex) 8888
- PATTERN 2 Words in pattern sequence - 1
Sequence (Octal) 031463
Sequence (Hex) 3333
- PATTERN 3 Words in pattern sequence - 1
Sequence (Octal) 030221
Sequence (Hex) 3091
- PATTERN 4 Words in pattern sequence - 16 (Shifting ones)
Sequence (Octal) 000001, 000003, 000007, 000017, 000037,
000077, 000177, 000377, 000777, 001777,
003777, 007777, 017777, 037777, 077777,
177777
Sequence (Hex) 0001, 0003, 0007, 000F, 001F, 003F,
007F, 00FF, 01FF, 03FF, 07FF, 0FFF,
1FFF, 3FFF, 7FFF, FFFF
- PATTERN 5 Words in pattern sequence - 16 (Shifting zeros)
Sequence (Octal) 177776, 177774, 177770, 177760, 177740,
177700, 177600, 177400, 177000, 176000,
174000, 170000, 160000, 140000, 100000,
000000
Sequence (Hex) FFFE, FFFC, FFF8, FFF0, FFE0, FFC0,
FF80, FF00, FE00, FC00, F800, F000,
E000, C000, 8000, 0000

- PATTERN 6 Words in pattern sequence - 16
Sequence (Octal) 000000, 000000, 000000, 177777, 177777,
177777, 000000, 000000, 177777, 177777,
000000, 177777, 000000, 177777, 000000,
177777
Sequence (Hex) 0000, 0000, 0000, FFFF, FFFF, FFFF,
0000, 0000, FFFF, FFFF, 0000, FFFF,
0000, FFFF, 0000, FFFF
- PATTERN 7 Words in pattern sequence - (BINARY 1011011011011001)
Sequence (Octal) 133331
Sequence (Hex) B6D9
- PATTERN 8 Words in pattern sequence - 16
Sequence (Octal) 052525, 052525, 052525, 125252, 125252,
125252, 052525, 052525, 125252, 125252,
052525, 125252, 052525, 125252, 052525,
125252
Sequence (Hex) 5555, 5555, 5555, AAAA, AAAA, AAAA,
5555, 5555, AAAA, AAAA, 5555, AAAA,
5555, AAAA, 5555, AAAA
- PATTERN 9 Words in pattern sequence - 1 (BINARY 1101101101101100)
Sequence (Octal) 155554
Sequence (Hex) DB6C
- PATTERN 10 Words in pattern sequence - 16
Sequence (Octal) 026455, 026455, 026455, 151322, 151322,
151322, 026455, 026455, 151322, 151322,
026455, 151322, 026455, 151322, 026455,
151322
Sequence (Hex) 2D2D, 2D2D, 2D2D, D2D2, D2D2, D2D2,
2D2D, 2D2D, D2D2, D2D2, 2D2D, D2D2,
2D2D, D2D2, 2D2D, D2D2
- PATTERN 11 Words in pattern sequence - 1 (BINARY 0110110110110110)
Sequence (Octal) 066666
Sequence (Hex) 6DD6

- PATTERN 12 Words in pattern sequence - 16 (Ripple one)
Sequence (Octal) 000001, 000002, 000004, 000010, 000020,
000040, 000100, 000200, 000400, 001000,
002000, 004000, 010000, 020000, 040000,
100000
Sequence (Hex) 0001, 0002, 0004, 0008, 0010, 0020,
0040, 0080, 0100, 0200, 0400, 0800,
1000, 2000, 4000, 8000
- PATTERN 13 Words in pattern sequence - 16 (Ripple zero)
Sequence (Octal) 177776, 177775, 177773, 177767, 177757,
177737, 177677, 177577, 177377, 176777,
175777, 173777, 167777, 157777, 137777,
077777
Sequence (Hex) FFFE, FFFD, FFFB, FFF7, FFEF, FFDF,
FFBF, FF7F, FEFF, FDFF, FBFF, F7FF,
EFFF, DFFF, BFFF, 7FFF
- PATTERN 14 Words in pattern sequence - 3
Sequence (Octal) 155555, 133333, 155555
Sequence (Hex) DB6D, B6DB, DB6D
- PATTERN 15 Words in pattern sequence - 16
Sequence (Octal) 133331, 133331, 133331, 155554, 155554,
155554, 133331, 133331, 155554, 155554,
133331, 155554, 133331, 155554, 133331,
155554
Sequence (Hex) B6D9, B6D9, B6D9, DB6C, DB6C, DB6C,
B6D9, B6D9, DB6C, DB6C, B6D9, DB6C,
B6D9, DB6C, B6D9, DB6C
- PATTERN 16 This is the operator selectable pattern in manual
intervention mode. Questions are asked when test #4 is
started for the operator to input the number of words in
the sequence and the contents of the words.

Sample of terminal dialogue going through manual intervention questions:

DR>STA/TEST:4

CHANGE HW (L) ? N

CHANGE SW (L) ? Y

ENTER MANUAL INTERVENTION MODE FOR SPECIAL DIAGNOSIS (L) N ? Y

REMAINING SOFTWARE QUESTIONS APPLY TO TEST 4 ONLY

ERROR LIMIT (D) 32 ?

READ TRANSFER LIMIT IN MEGABYTES - 0 FOR NO LIMIT (D) 0 ?

SUPPRESS PRINTING SOFT ERRORS (L) Y ? N

DO INITIAL WRITE ON START (L) Y ?

ENABLE ERROR LOG (L) N ?

THE FOLLOWING QUESTIONS REFER TO UNIT 0 UDA AT 172150 DRIVE 0

NUMBER OF BAD BLOCKS (D) 0 ? 2

BAD BLOCK (A) ? 234

BAD BLOCK (A) ? 8900

DO YOU WANT TO CHANGE TESTING PARAMETERS FOR THIS DRIVE (L) N ? Y

READ ONLY (L) N ?

WRITE ONLY (L) N ?

CHECK ALL WRITES BY READING (L) N ? Y

DATA PATTERN - 0 FOR RANDOM SELECTION (D) 0 ? 1

ENABLE ECC DATA CORRECTION (L) Y ?

COMPARE ALL DATA READ (L) N ? Y

ENABLE RETRIES (L) Y ?

RANDOM ACCESS MODE (L) Y ? N

DO YOU WISH TO:

0 - TEST ENTIRE AREA SELECTED

1 - SPECIFY BEGIN/END SETS TO TEST

2 - SPECIFY TRACKS AND CYLINDERS TO TEST

3 - SPECIFY GROUPS AND CYLINDERS TO TEST

4 - SPECIFY CYLINDERS TO TEST

(D) 0 ? 1

NUMBER OF BEGIN/END SETS (D) 1 ?

BEGIN BLOCK (A) 0 ?

END BLOCK (A) 0 ? 200

NUMBER OF WORDS IN DATA PATTERN 16 (D) 1 ?

DATA WORD (O) 0 ?

&

1
 32
 33
 34
 35
 36
 37
 38
 39
 40
 42

.SBTTL PROGRAM HEADER
 BGNMOD
 :++
 : THE PROGRAM HEADER IS THE INTERFACE BETWEEN
 : THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
 :--
 POINTER BGNRPT, BGNSW, BGNSFT, ERRIBL, BGNSETUP
 HEADER CZUDC,A,9,0,1,PRI07 ;FIELD SERVICE

002000 103
 002001 132
 002002 125
 002003 104
 002004 103
 002005 000
 002006 000
 002007 000
 002010 101
 002011 071
 002012 000001
 002014 000000
 002016 034122
 002020 034366
 002022 002136
 002024 002154
 002026 035152
 002030 000000
 002032 000000
 002034 000001
 002036 000000
 002040 002124
 002042 000340
 002044 000000
 002046 000000
 002050 003
 002051 003

LSNAME::
 .ASCII /C/
 .ASCII /Z/
 .ASCII /U/
 .ASCII /D/
 .ASCII /C/
 .BYTE 0
 .BYTE 0
 .BYTE 0
 LSREV::
 .ASCII /A/
 LSDEPO::
 .ASCII /9/
 LSUNIT::
 .WORD TSPTHV
 LSTIML::
 .WORD 0
 LSHPCP::
 .WORD LSHARD
 LSSPCP::
 .WORD LSSOFT
 LSHPTP::
 .WORD LSHW
 LSSPTP::
 .WORD LSSW
 LSLADP::
 .WORD LSLAST
 L\$STA::
 .WORD 0
 L\$CO::
 .WORD 0
 L\$DTYP::
 .WORD 1
 L\$APT::
 .WORD 0
 L\$DTP::
 .WORD L\$DISPATCH
 L\$PRIO::
 .WORD PRI07
 L\$ENVI::
 .WORD 0
 L\$EXP1::
 .WORD 0
 L\$MREV::
 .BYTE C\$REVISION
 .BYTE C\$EDIT

002052
002052 000000
002054 000000
002056
002056 000000
002060
002060 002436
002062
002062 025550
002064
002064 000000
002066
002066 000000
002070
002070 000000
002072
002072 000000
002074
002074 000000
002076
002076 002462
002100
002100 104035
002102
002102 002162
002104
002104 026600
002106
002106 030206
002110
002110 030204
002112
002112 026572
002114
002114 000000
002116
002116 000000
002120
002120 000000

LSEF::
 .WORD 0
 .WORD 0
L\$SPC::
 .WORD 0
L\$DEVP::
 .WORD L\$DVTYP
L\$REPP::
 .WORD L\$RPT
L\$EXP4::
 .WORD 0
L\$EXP5::
 .WORD 0
L\$AUT::
 .WORD 0
L\$DUT::
 .WORD 0
L\$LUN::
 .WORD 0
L\$DESP::
 .WORD L\$DESC
L\$LOAD::
 EMT ESLOAD
L\$ETP::
 .WORD L\$ERRTBL
L\$ICP::
 .WORD L\$INIT
L\$CCP::
 .WORD L\$CLEAN
L\$ACP::
 .WORD L\$AUTO
L\$PRT::
 .WORD L\$PROT
L\$TEST::
 .WORD 0
L\$DLY::
 .WORD 0
L\$HIME::
 .WORD 0

1
2
3
4
5
6
7
8
9

.SBTTL DISPATCH TABLE

:++
: THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
: IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
:--

DISPATCH 4

002122
002122 000004
002124
002124 030230
002126 031404
002130 031502
002132 031540

.WORD 4
LSDISPATCH::
.WORD T1
.WORD T2
.WORD T3
.WORD T4

.SBTTL DEFAULT HARDWARE P-TABLE

:++
: THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
: THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE
: IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P-TABLES,
: AND IS USED AS A 'TEMPLATE' FOR BUILDING THE P-TABLES.
:--

1							
2							
3							
4							
5							
6							
7							
8							
9							
10	002134		BGNHW	DFPTBL			
	002134	000006				.WORD	L10000-LSHW/2
	002136					LSHW::	
	002136					DFPTBL::	
11							
12	002136	172150	.WORD	172150		:	UNIBUS ADDRESS
13	002140	000154	.WORD	154		:	VECTOR ADDRESS
14	002142	000005	.WORD	5.		:	BR LEVEL
15	002144	000077	.WORD	63.		:	UNIBUS BURST RATE
16	002146	000000	.WORD	0.		:	LOGICAL DRIVE NUMBER
17	002150	000000	.WORD	0.		:	CUSTOMER DATA AREA
18	002152		ENDHW				
	002152						L10000:

```
1          .SBTTL  SOFTWARE P-TABLE
2
3
4          :++
5          : THE SOFTWARE TABLE CONTAINS VARIOUS DATA USED BY THE
6          : PROGRAM AS OPERATIONAL PARAMETERS.  THESE PARAMETERS ARE
7          : SET UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR
8          : AT RUN TIME.
9          :--
10         BGNSW  SFPTBL
11
12
13         002152 000003          .WORD  L10001-L$SW/2
14         002152 000003          SFPTBL::
15         002154
16         002154
17
18         .WORD  32.          :OFFSET  USE
19         002156 000000          : 0.  ERROR LIMIT
20         002160 040400          : 2.  DATA TRANSFER LIMIT (MEGABITS)
21         002162          : 4.  SINGLE BIT QUESTIONS
22
23         ENDSW
24
25         L10001:
26
27         ENDMOD
```


1
2
3 002162
4
5
6
7
8
9
10 002162

.SBTTL GLOBAL EQUATES SECTION

BGNMOD

;++
: THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
: ARE USED IN MORE THAN ONE TEST.
:--

EQUALS

:
: BIT DIFINITIONS

100000	BIT15== 100000
040000	BIT14== 40000
020000	BIT13== 20000
010000	BIT12== 10000
004000	BIT11== 4000
002000	BIT10== 2000
001000	BIT09== 1000
000400	BIT08== 400
000200	BIT07== 200
000100	BIT06== 100
000040	BIT05== 40
000020	BIT04== 20
000010	BIT03== 10
000004	BIT02== 4
000002	BIT01== 2
000001	BIT00== 1

001000	BIT9== BIT09
000400	BIT8== BIT08
000200	BIT7== BIT07
000100	BIT6== BIT06
000040	BIT5== BIT05
000020	BIT4== BIT04
000010	BIT3== BIT03
000004	BIT2== BIT02
000002	BIT1== BIT01
000001	BIT0== BIT00

:
: EVENT FLAG DEFINITIONS
: EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION

000040	EF.START== 32.	: START COMMAND WAS ISSUED
000037	EF.RESTART== 31.	: RESTART COMMAND WAS ISSUED
000036	EF.CONTINUE== 30.	: CONTINUE COMMAND WAS ISSUED
000035	EF.NEW== 29.	: A NEW PASS HAS BEEN STARTED
000034	EF.PWR== 28.	: A POWER-FAIL/POWER-UP OCCURRED

:
: PRIORITY LEVEL DEFINITIONS

000340	PRI07== 340
000300	PRI06== 300
000240	PRI05== 240
000200	PRI04== 200

```
000140      PRI03== 140
000100      PRI02== 100
000040      PRI01== 40
000000      PRI00== 0
           ;
           ;OPERATOR FLAG BITS
           ;
000004      EVL==      4
000010      LOT==     10
000020      ADR==     20
000040      IDU==     40
000100      ISR==    100
000200      UAM==    200
000400      BOE==    400
001000      PNT==   1000
002000      PRI==   2000
004000      IXE==   4000
010000      IBE==  10000
020000      IER==  20000
040000      LOE==  40000
100000      HOE== 100000
```

11
12

```
000015      CR=      15
```

;VALUE TO PASS TO PRINT MACRO TO END LINE

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33

:MACRO DEFINITIONS FOR GLOBAL EQUATES

:THESE MACROS ARE USED TO DEFINE INDEXES INTO A TABLE

:CALLING SEQUENCE MUST BE

```
TABLE  
ITEM NAME BYTES  
ITEM NAME BYTES  
ITEM NAME BYTES  
END SIZE
```

:TABLE DEFINES THAT A TABLE IS ABOUT TO BE DEFINED AND END TERMINATES THE DEFINITION.
:ANY NUMBER OF ITEM LINES CAN APPEAR. NAME IS THE NAME OF THE SYMBOL BEING EQUATED TO
:THE INDEX. THE INDEX ALWAYS STARTS AT ZERO. BYTES SPECIFIES THE SIZE OF THE VALUE TO BE
:STORED AT THAT INDEX IN BYTES. THE SIZE ARGUMENT TO THE END STATEMENT IS OPTIONAL, IT
:BE EQUATED TO THE SIZE OF THE TABLE IN BYTES. THE SYMBOL TINDEX IS USED TO KEEP TRACK
:OF THE INDEX VALUE AND WILL BE EQUAL TO THE SIZE OF THE TABLE AFTER THE END STATEMENT.

```
.MACRO TABLE  
TINDEX=0
```

```
.ENDM
```

```
.MACRO ITEM NAME BYTES  
NAME=TINDEX  
TINDEX=TINDEX+BYTES
```

```
.ENDM
```

```
.MACRO END SIZE  
.IF NB SIZE  
SIZE=TINDEX  
.ENDC
```

```
.ENDM
```



```

1      ;UDA BIT DEFINITIONS
2
3      ;UDASA REGISTER UNIVERSAL READ BITS
4
5      004000      SA.S1= 004000      ;STEP 1 STATUS BIT
6      010000      SA.S2= 010000      ;STEP 2 STATUS BIT
7      020000      SA.S3= 020000      ;STEP 3 STATUS BIT
8      040000      SA.S4= 040000      ;STEP 4 STATUS BIT
9      100000      SA.ERR= 100000     ;ERROR INDICATOR
10
11     ;UDASA REGISTER ERROR STATUS BITS
12
13     003777      SA.ERC= 003777     ;ERROR CODE
14
15     ;UDASA REGISTER STEP ONE READ BITS
16
17     002000      SA.NV= 002000     ;NON SETTABLE INTERRUPT VECTOR
18     001000      SA.A2= 001000     ;22 BIT ADDRESS BUS
19     000400      SA.DI= 000400     ;ENHANCED DIAGNOSTICS
20     ;           ;           ;ALL BITS RESERVED
21
22     ;UDASA REGISTER STEP ONE WRITE BITS
23
24     000177      SA.VEC= 000177     ;INTERRUPT VECTOR (DIVIDED BY 4)
25     000200      SA.INT= 000200     ;INTERRUPT ENABLE DURING INITIALIZATION
26     003400      SA.MSG= 003400     ;MESSAGE RING LENGTH
27     034000      SA.CMD= 034000     ;COMMAND RING LENGTH
28     040000      SA.WRP= 040000     ;WRAP BIT
29     100000      SA.STP= 100000     ;STEP - MUST ALWAYS BE WRITTEN A ONE
30
31     000400      SA.MS1= 000400     ;LSB OF MESSAGE RING LENGTH
32     004000      SA.CM1= 004000     ;LSB OF COMMAND RING LENGTH
33
34     ;UDASA REGISTER STEP TWO READ BITS
35
36     000007      SA.MSE= 000007     ;MESSAGE RING LENGTH ECHO
37     000070      SA.CME= 000070     ;COMMAND RING LENGTH ECHO
38     ;           ;           ;RESERVED
39     000200      SA.STE= 000200     ;STEP ECHO
40     003400      SA.CTP= 003400     ;CONTROLLER TYPE
41
42     ;UDASA REGISTER STEP TWO WRITE BITS
43
44     000001      SA.PRG= 000001     ;ENABLE VAX UNIBUS ADAPTER PURGE INTERRUPT
45     ;           ;           ;LOW ORDER MESSAGE RING BYTE ADDRESS
    
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22

```
      ;UDASA REGISTER STEP THREE READ BITS
000177 SA.VCE= 000177      ;INTERRUPT VECTOR ECHO
000200 SA.INE= 000200      ;INTERRUPT ENABLE ECHO
000400 SA.NVE= 000400      ;VECTOR NOT PROGRAMMABLE
      ;      003000      ;RESERVED

      ;UDASA REGISTER STEP THREE WRITE BITS
      ;      077777      ;HIGH ORDER MESSAGE RING BYTE ADDRESS
100000 SA.TST= 100000      ;PURGE POLE TEST ENABLE

      ;UDASA REGISTER STEP FOUR READ BITS
000377 SA.MCV= 000377      ;UDA MICROCODE VERSION
      ;      003400      ;RESERVED

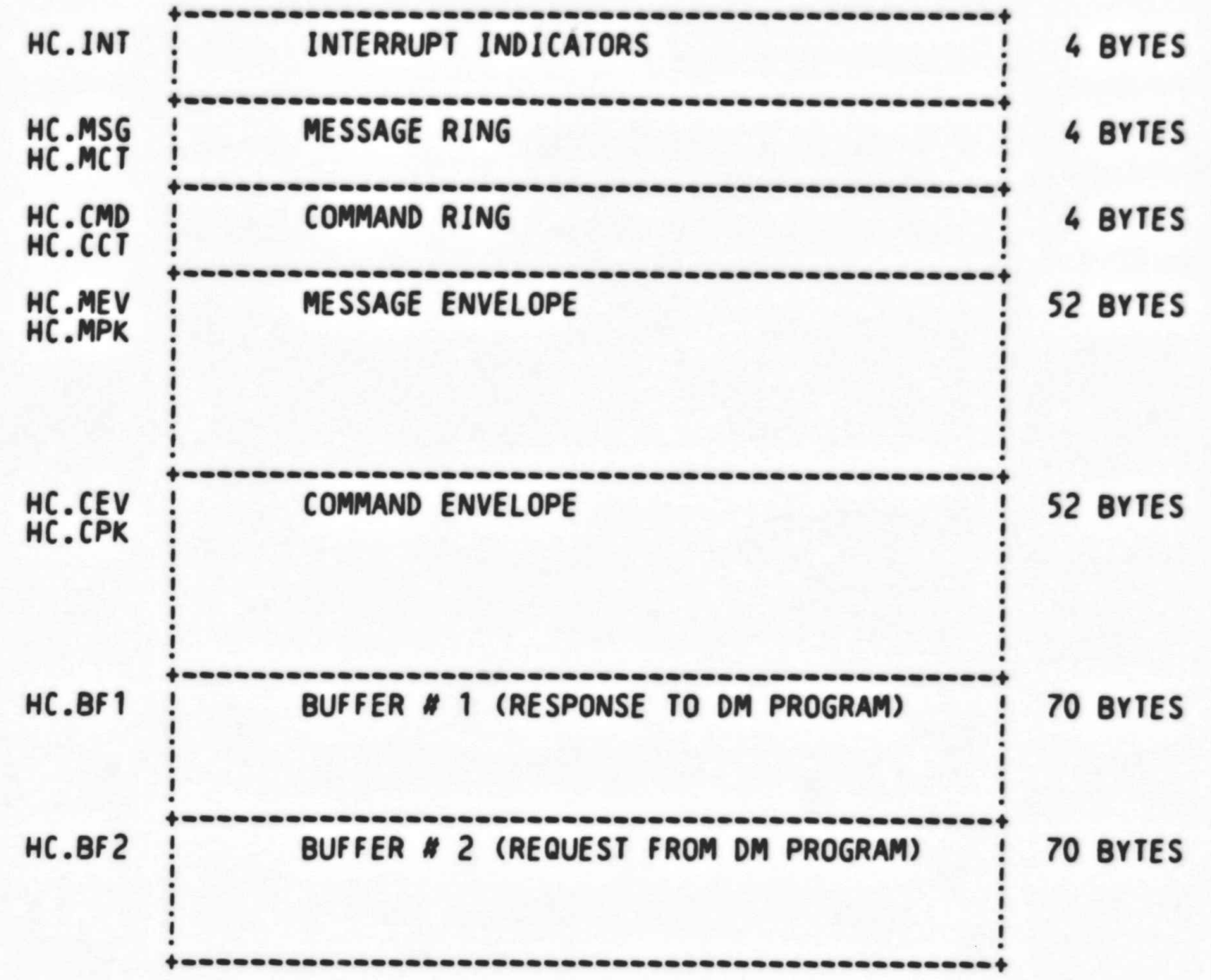
      ;UDASA REGISTER STEP FOUR WRITE BITS
000001 SA.GO= 000001      ;GO BIT TO START UDA FIRMWARE
000002 SA.LFC= 000002      ;LAST FAILURE CODE REQUEST
000374 SA.BST= 000374      ;BURST LEVEL
```

```

1          ;COMMAND/MESSAGE DESCRIPTOR BIT DEFINITIONS
2
3          100000      RG.OWN= 100000          ;SET WHEN UDA OWNS RING
4          040000      RG.FLG= 040000          ;FLAG BIT
5
6          ;OFFSETS INTO HOST COMMUNICATIONS AREA WITH ONE DESCRIPTOR TO EACH RING
7          ;AND TWO PACKET AND BUFFER AREAS.
8
9          000004      HC.ISZ= 4.              ;SIZE OF INTERRUPT INDICATOR WORDS
10         000004      HC.RSZ= 4.              ;SIZE OF RING IN BYTES
11         000004      HC.ESZ= 4.              ;SIZE OF ENVELOPE WORDS BEFORE PACKET
12         000060      HC.PSZ= 48.             ;SIZE OF COMMAND AND MESSAGE PACKETS
13         000106      HC.BSZ= 70.             ;SIZE OF BUFFER
14
15         000000      HC.INT= 0.              ;INTERRUPT INDICATOR WORDS START
16         000004      HC.MSG= HC.INT+HC.ISZ   ;MESSAGE RING START
17         000006      HC.MCT= HC.MSG+2.       ;MESSAGE RING CONTROL WORD
18         000010      HC.CMD= HC.MSG+HC.RSZ   ;COMMAND RING START
19         000012      HC.CCT= HC.CMD+2.       ;COMMAND RING CONTROL WORDS
20         000014      HC.MEV= HC.CMD+HC.RSZ   ;MESSAGE ENVELOPE START
21         000020      HC.MPK= HC.MEV+HC.ESZ   ;MESSAGE PACKET START
22         000100      HC.CEV= HC.MPK+HC.PSZ   ;COMMAND ENVELOPE START
23         000104      HC.CPK= HC.CEV+HC.ESZ   ;COMMAND PACKET START
24         000164      HC.BF1= HC.CPK+HC.PSZ   ;FIRST BUFFER
25         000272      HC.BF2= HC.BF1+HC.BSZ   ;SECOND BUFFER
26
27         000400      HC.SIZ= HC.BF2+HC.BSZ   ;TOTAL SIZE OF HOST COMM AREA
28
29         ;VIRTUAL CIRCUIT IDENTIFIERS
30
31         000000      MSCP= 0                  ;MSCP CIRCUIT
32         000001      LOG= 1                  ;LOG CIRCUIT
33         177777      DIAG= -1                ;DIAGNOSTIC CIRCUIT
34         001000      DUP= 1000               ;DIAGNOSTIC AND UTILITIES PROTOCOL
    
```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34

.....



```

1      :COMMAND PACKET OPCODES
2
3      000001      OP.ABO= 1      :ABORT COMMAND
4      000020      OP.ACC= 20     :ACCESS COMMAND
5      000010      OP.AVL= 10     :AVAILABLE COMMAND
6      000021      OP.CCD= 21     :COMPARE CONTROLLER DATA COMMAND
7      000040      OP.CMP= 40     :COMPARE HOST DATA COMMAND
8      000022      OP.ERS= 22     :ERASE COMMAND
9      000023      OP.FLU= 23     :FLUSH COMMAND
10     000002      OP.GCS= 2      :GET COMMAND STATUS COMMAND
11     000003      OP.GUS= 3      :GET UNIT STATUS COMMAND
12     000011      OP.ONL= 11     :ONLINE COMMAND
13     000041      OP.RD= 41      :READ COMMAND
14     000024      OP.RPL= 24     :REPLACE COMMAND
15     000004      OP.SCC= 4      :SET CONTROLLER CHARACTERISTICS COMMAND
16     000012      OP.SUC= 12     :SET UNIT CHARACTERISTICS COMMAND
17     000042      OP.WR= 42      :WRITE COMMAND
18     000030      OP.MRD= 30     :MAINTENANCE READ COMMAND
19     000031      OP.MWR= 31     :MAINTENANCE WRITE COMMAND
20     000200      OP.END= 200    :END PACKET FLAG
21     000007      OP.SEX= 7      :SERIOUS EXCEPTION END PACKET
22     000100      OP.AVA= 100    :AVAILABLE ATTENTION MESSAGE
23     000101      OP.DUP= 101    :DUPLICATE UNIT NUMBER ATTENTION MESSAGE
24     000102      OP.SHC= 102    :SHADOW COPY COMPLETE ATTENTION MESSAGE
25     000103      OP.RLC= 103    :RESET COMMAND LIMIT ATTENTION MESSAGE
26
27     000001      OP.GSS= 1      :DUP GET STUD STATUS
28     000002      OP.ESP= 2      :DUP EXECUTE SUPPLIED PROGRAM
29     000003      OP.ELP= 3      :DUP EXECUTE LOCAL PROGRAM
30     000004      OP.SSD= 4      :DUP SEND STUD DATA
31     000005      OP.RSD= 5      :DUP RECEIVE STUD DATA
32
33     :NOTE: END PACKET OPCODES (ALSO CALLED ENDCODES) ARE FORMED BY ADDING THE END
34     :PACKET FLAG TO THE COMMAND OPCODE. FOR EXAMPLE, A READ COMMAND'S END PACKET
35     :CONTAINS THE VALUE OP.RD+OP.END IN ITS OPCODE FIELD. THE INVALID COMMAND END
36     :PACKET CONTAINS JUST THE END PACKET FLAG (I.E., OP.END) IN ITS OPCODE FIELD.
37     :THE SERIOUS EXCEPTION END PACKET CONTAINS THE SUM OF THE END PACKET FLAG
38     :PLUS THE SERIOUS EXCEPTION OPCODE SHOWN ABOVE (I.E., OP.SEX+OP.END) IN ITS
39     :OPCODE FIELD.
40
41     :COMMAND OPCODE BITS 3 THROUGH 5 INDICATE THE COMMAND CLASS, WHICH IS ENCODED
42     :AS FOLLOWS:
43     : 000 IMMEDIATE COMMANDS
44     : 001 SEQUENTIAL COMMANDS
45     : 010 NON-SEQUENTIAL COMMANDS THAT DO NOT INCLUDE A BUFFER DESCRIPTOR
46     : 100 NON-SEQUENTIAL COMMANDS THAT DO INCLUDE A BUFFER DESCRIPTOR
    
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39

:COMMAND MODIFIERS

040000
100000
010000
004000
002000
000100
000400
000200
000100
000400
000020
000001
000001
000002
000001
000001
000001
000002
000004
000010
000001

MD.CMP= 040000
MD.EXP= 100000
MD.ERR= 010000
MD.SCH= 004000
MD.SCL= 002000
MD.SEC= 000100
MD.SER= 000400
MD.SSH= 000200
MD.WBN= 000100
MD.WBV= 000400
MD.SEQ= 000020
MD.SPD= 000001
MD.FEU= 000001
MD.VOL= 000002
MD.NXU= 000001
MD.RIP= 000001
MD.IMF= 000002
MD.SWP= 000004
MD.CWB= 000010
MD.PRI= 000001

:CLEAR SERIOUS EXCEPTION
:COMPARE
:EXPRESS REQUEST
:FORCE ERROR
:SUPPRESS CACHING (HIGH SPEED)
:SUPPRESS CACHING (LOW SPEED)
:SUPPRESS ERROR CORRECTION
:SUPPRESS ERROR RECOVERY
:SUPPRESS SHADOWING
:WRITE-BACK (NON-VOLATILE)
:WRITE BACK (VOLATILE)
:WRITE SHADOW SET ONE UNIT AT A TIME
:SPIN-DOWN
:FLUSH ENTIRE UNIT
:VOLATILE ONLY
:NEXT UNIT
:ALLOW SELF DESTRUCTION
:IGNORE MEDIA FORMAT ERROR
:SET WRITE PROTECT
:CLEAR WRITE-BACK DATA LOST
:PRIMARY REPLACEMENT BLOCK

:END PACKET FLAGS

000200
000100
000040
000020

EF.BBR= 000200
EF.BBU= 000100
EF.LOG= 000040
EF.SEX= 000020

:BAD BLOCK REPORTED
:BAD BLOCK UNREPORTED
:ERROR LOG GENERATED
:SERIOUS EXCEPTION

:CONTROLLER FLAGS

000200
000100
000040
000020
000002
000001

CF.ATN= 000200
CF.MSC= 000100
CF.OTH= 000040
CF.THS= 000020
CF.SHD= 000002
CF.576= 000001

:ENABLE ATTENTION MESSAGES
:ENABLE MISCELLANEOUS ERROR LOG MESSAGES
:ENABLE OTHER HOST'S ERROR LOG MESSAGES
:ENABLE THIS HOST'S ERROR LOG MESSAGES
:SHADOWING
:576 BYTE SECTORS


```
1          ;STATUS AND EVENT CODE DEFINITIONS
2
3          000037      ST.MSK= 37          ;STATUS / EVENT CODE MASK
4          000040      ST.SUB= 40         ;SUB-CODE MULTIPLIER
5          000000      ST.SUC= 0          ;SUCCESS
6          000001      ST.CMD= 1          ;INVALID COMMAND
7          000002      ST.ABO= 2          ;COMMAND ABORTED
8          000003      ST.OFL= 3          ;UNIT-OFFLINE
9          000004      ST.AVL= 4          ;UNIT-AVAILABLE
10         000005      ST.MFE= 5          ;MEDIA FORMAT ERROR
11         000006      ST.WPR= 6          ;WRITE PROTECTED
12         000007      ST.CMP= 7          ;COMPARE ERROR
13         000010      ST.DAT= 10         ;DATA ERROR
14         000011      ST.HST= 11        ;HOST BUFFER ACCESS ERROR
15         000012      ST.CNT= 12        ;CONTROLLER ERROR
16         000013      ST.DRV= 13        ;DRIVE ERROR
17         000037      ST.DIA= 37        ;MESSAGE FROM AN INTERNAL DIAGNOSTIC
18
19         ;DUP MESSAGE TYPES
20         .
21         010000      DU.QUE = 10000     ;QUESTION
22         020000      DU.DFL = 20000     ;DEFAULT QUESTION
23         030000      DU.INF = 30000     ;INFORMATION
24         040000      DU.TER = 40000     ;TERMINATOR
25         050000      DU.FTL = 50000     ;FATAL ERROR
26         060000      DU.SPC = 60000     ;SPECIAL
27
```



```

1          ;CONTROLLER TABLE DEFINITIONS
2
3          ;ONE TABLE WILL BE SET UP BY INITIALIZE SECTION FOR EACH UDA SELECTED
4          ;FOR TESTING. TABLES ARE CONTIGUOUS. THE END OF THE TABLES IS
5          ;MARKED BY A WORD OF ZEROS.
6
7          ;THE FIRST TABLE IS POINTED TO BY THE CONTENTS OF CTABS.
8          ;THE NUMBER OF TABLES IS CONTAINED IN CTRLRS.
9
10         002162      TABLE          ;START A TABLE DEFINITION
11
12         002162      ITEM C.UADR      2          ;UNIBUS ADDRESS OF UDAIP REGISTER
13         002162      ITEM C.UNIT      2
14         000077      CT.UNT= 000077      ; LOGICAL UNIT NUMBER (FIRST)
15         100000      CT.AVL= BIT15      ; SET WHEN NOT AVAILABLE FOR TESTING
16         002162      ITEM C.VEC      2
17         000777      CT.VEC= 000777      ; VECTOR ADDRESS
18         007000      CT.BRL= 007000      ; BR LEVEL
19         002162      ITEM C.BST      2          ; BURST LEVEL
20         002162      ITEM C.JSR      2          ; INTERRUPT SERVICE ROUTINE FOR CONTROLLER
21         002162      ITEM C.JAD      2          ; THESE TWO WORDS LOADED WITH [JSR R0,UDASRV]
22         002162      ITEM C.FLG      2          ; FLAGS
23         000002      CT.RN= BIT1      ; DM PROGRAM RUNNING
24         000004      CT.CMD= BIT2      ; COMMAND ISSUED, WAITING FOR RESPONSE
25         000010      CT.MSG= BIT3      ; MESSAGE RESPONSE RECEIVED
26
27         000020      CT.REQ= BIT4      ; WHENEVER THIS BIT IS SET, CT.CMD IS CLEARED
28
29
30
31
32
33         002162      ITEM C.RING      2          ; BUFFER HAS BEEN GIVEN TO UDA FOR REQUEST
34         002162      ITEM C.DR0      2          ; SET WHENEVER READ STUD DATA COMMAND
35         002162      ITEM C.DR1      2          ; GIVEN TO UDA
36         002162      ITEM C.DR2      2          ; RING BUFFER ADDRESS
37         002162      ITEM C.DR3      2          ; POINTER TO DRIVE TABLES
38         002162      ITEM C.DR4      2          ; IF ZERO, NO DRIVE TABLE EXISTS
39         002162      ITEM C.DR5      2
40         002162      ITEM C.DR6      2
41         002162      ITEM C.DR7      2
42         002162      ITEM C.TO      2          ; TIMEOUT COUNTER
43         002162      ITEM C.TOH      2          ; (TWO WORDS)
44         002162      ITEM C.REF      2          ; COMMAND REFERENCE NUMBER
45
46         002162      END C.SIZE          ;SIZE OF CONTROLLER TABLE IN BYTES
    
```

```

1          ;DRIVE TABLE DEFINITIONS
2
3          ;
4          ;ONE DRIVE TABLE WILL BE SET UP BY THE INITIALIZE SECTION FOR EACH
5          ;DRIVE SELECTED FOR TESTING.  EACH TABLE IS POINTED TO BY A
6          ;WORD IN THE CONTROLLER TABLE ON WHICH THE DRIVE EXISTS.
7 002162   TABLE                               ;START A TABLE DEFINITION
8
9 002162   ITEM D.DRV                           2           ;DRIVE NUMBER
10 002162  ITEM D.UNIT                          2
11          DT.UNT= 000077                       ; LOGICAL UNIT NUMBER OF DRIVE
12          DT.AVL= BIT15                         ; SET WHEN NOT AVAILABLE FOR TESTING
13 002162  ITEM D.PRM                          2           ;HARDWARE QUESTION FLAGS
14          D.IW =BIT14                          ;INITIAL WRITE
15          D.DCY =BIT13                         ;DIAGNOSTIC CYLINDERS
16          D.ECC =BIT12                         ;ECC CORRECTION ENABLED
17          D.RO =BIT11                          ;READ ONLY
18          D.WO =BIT10                          ;WRITE ONLY
19          D.RET =BIT9                          ;RETRIES ENABLED
20          D.CYL =BIT8                          ;START/END CYLINDERS SPECIFIED
21          D.SEQ =BIT6                          ;SEQUENTIAL ACCESS
22          D.BE =BIT5                          ;BEGIN/END BLOCKS USED
23          D.TR =BIT4                          ;WHEN D.BE=0: 1 - TRACKS, 0 - GROUPS
24          D.WC =BIT3                          ;WRITE CHECKS ENABLED
25          D.WCA =BIT2                         ;ALWAYS WRITE CHECK
26          D.DC =BIT1                          ;DATA COMPARES ENABLED
27          D.DCA =BIT0                         ;ALWAYS DATA COMPARE
31          DDEF=D.ECC+D.WC+D.DC+D.RET          ;DEFAULT D.PRM
33          D.ZERO=BIT15+BIT7+D.IW             ;BITS TO BE CLEARED
34 002162  ITEM D.PAT                          2           ;DATA PATTERN NUMBER
35 002162  ITEM D.BB                          2           ;BAD BLOCK COUNT
36 002162  ITEM D.BB01                        4           ;BAD BLOCK 1
37 002162  ITEM D.BB02                        4           ;
38 002162  ITEM D.BB03                        4           ;
39 002162  ITEM D.BB04                        4           ;
40 002162  ITEM D.BB05                        4           ;
41 002162  ITEM D.BB06                        4           ;
42 002162  ITEM D.BB07                        4           ;
43 002162  ITEM D.BB08                        4           ;
44 002162  ITEM D.BB09                        4           ;
45 002162  ITEM D.BB10                       4           ;
46 002162  ITEM D.BB11                       4           ;
47 002162  ITEM D.BB12                       4           ;
48 002162  ITEM D.BB13                       4           ;
49 002162  ITEM D.BB14                       4           ;
50 002162  ITEM D.BB15                       4           ;
51 002162  ITEM D.BB16                       4           ;

```


1	002162	ITEM D.BEC	2	:BEGIN/END SET COUNT
2	002162	ITEM D.BGN1	4	:BEGIN BLOCK 1
3	002162	ITEM D.END1	4	:END
4	002162	ITEM D.BGN2	4	:BEGIN BLOCK 2
5	002162	ITEM D.END2	4	:END
6	002162	ITEM D.BGN3	4	:BEGIN BLOCK 3
7	002162	ITEM D.END3	4	:END
8	002162	ITEM D.BGN4	4	:BEGIN BLOCK 4
9	002162	ITEM D.END4	4	:END
10	002162	ITEM D.BCYL	4	:BEGIN CYLINDER
11	002162	ITEM D.ECYL	4	:END CYLINDER
12	002162	ITEM D.XFRW	2	:MEGABITS WRITTEN COUNT
13	002162	ITEM D.XFRR	2	:MEGABITS READ COUNT
14	002162	ITEM D.HERR	2	:HARD ERROR COUNTER
15	002162	ITEM D.SERR	2	:SOFT ERROR COUNTER
16	002162	ITEM D.SEEK	2	:NUMBER OF SEEKS X1000
17	002162	ITEM D.ECCC	2	:ECC COUNTER
18	002162	ITEM D.SERN	6	:DRIVE SERIAL NUMBER
23				
24	002162	END D.SIZE		:SIZE OF DRIVE TABLE IN BYTES
25				
26		:DM PROGRAM HEADER DEFINITIONS		
27				
28	000000	DMTRLN= 0		:OFFSET TO SIZE OF PROGRAM NEEDING DOWNLINE LOAD
29	000004	DMOVRL= 4		:OFFSET TO SIZE OF OVERLAY
30	000040	DMMAIN= 40		:OFFSET TO FIRST WORD OF MAIN PROGRAM
31	001000	DMFRST= 1000		:ADDRESS IN DM FILE CONTAINING FIRST BYTE OF HEADER

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44

```
:USEFUL INSTRUCTION DEFINITIONS

.MACRO AND ARG,ADR                ;LOGICAL AND INSTRUCTION
.LIST                               BIC #^C<ARG>,ADR
.NLIST
.ENDM

.MACRO OR ARG,ADR                 ;LOGICAL OR INSTRUCTION
.LIST                               BIS #ARG,ADR
.NLIST
.ENDM

.MACRO PUSH ARG                   ;PUSH INSTRUCTION
.IRP X,<ARG>
.LIST                               MOV X,-(SP)
.NLIST
.ENDM

.MACRO POP ARG                    ;POP INSTRUCTION
.IRP X,<ARG>
.LIST                               MOV (SP)+,X
.NLIST
.ENDM

.MACRO .BR ADR                    ;A BRANCH TO THE NEXT LOCATION
.IF P2
    .IF NE .-ADR
        .ERROR ;ILLEGAL .BR TO ADR
    .ENDC
.ENDC
.ENDM

.MACRO ASSUME FIRST CONDITION SECOND
    .IF CONDITION <FIRST>--<SECOND>
    .IFF
        .ERROR ;BAD ASSUME OF <FIRST> CONDITION <SECOND>
    .ENDC
.ENDM
```

```
1      ;PRINT CHARACTER
2      ; ARGUMENT MUST BE SOURCE STATEMENT TO MOVE CHARACTER TO PRINT (MOV ARG,RO)
3      ; EX: 'PRINT R1' WILL PRINT THE CHARACTER IN R1
4      ; SPECIAL CASE: 'PRINT #CR' WILL PRINT END OF LINE SEQUENCE
5      ; THE PRINTING IS DONE AT THE MODE OF THE LAST PRINT LINE CALL
6      ; IE., PNTX, PNTB, PNTX, PNTS
7
8      .MACRO PRINT ARG1
9          .IF DIF <ARG1>,RO
10             .LIST
11
12             .NLIST
13             MOV B ARG1,RO
14
15             .ENDC
16             .LIST
17             CALL CPNT
18             .NLIST
19
20 .ENDM
21
22 ;PROCESSING MACRO FOR NEXT SET OF FORMATTED MESSAGE MACROS
23
24 .MACRO PNT... RTN,ADR,ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
25     PNT.CT=0
26     .IRP AA,<ARG8,ARG7,ARG6,ARG5,ARG4,ARG3,ARG2,ARG1>
27         .IF NB,<AA>
28             .LIST
29             MOV AA,-(SP)
30             .NLIST
31             PNT.CT=PNT.CT+2
32             .ENDC
33         .ENDM
34         .LIST
35
36         .NLIST
37         JSR R1,RTN
38         .WORD ADR
39         .WORD PNT.CT
40
41 .ENDM
42
43 ;PRINT FORMATTED MESSAGE MACROS
44 ; USE THESE MACROS TO PRINT A FORMATTED MESSAGE
45 ; FIRST ARGUMENT MUST BE ADDRESS OF FIRST CHARACTER OF MESSAGE STRING
46 ; TO BE PUT INTO WORD (.WORD ARG)
47 ; UP TO 8 SOURCE STATEMENTS MAY FOLLOW TO SPECIFY PARAMETERS TO BE
48 ; USED BY THE FORMAT
49
50 .MACRO PNTF ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
51     PNT... LPNTF ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
52 .ENDM
53
54 .MACRO PNTB ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
55     PNT... LPNTB ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
56 .ENDM
57
58 .MACRO PNTX ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
59     PNT... LPNTX ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
60 .ENDM
61
62 .MACRO PNTS ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
63     PNT... LPNTS ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
64 .ENDM
65
66 .MACRO PNT ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
```

58
59

.ENDM PNT... LPNT ADR ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8


```

1      .SBTTL GLOBAL DATA SECTION
2
3
4      :++
5      : THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
6      : IN MORE THAN ONE TEST.
7      :--
8      002162          ERRTABL
9      002162          000000          L$ERRTABL::
10     002164          000000          ERRRTYP:: .WORD 0
11     002166          000000          ERRNBR:: .WORD 0
12     002170          000000          ERRMSG:: .WORD 0
13     002172          022144          000          ERRBLK:: .WORD 0
14     002174          000          000          PTYPE: .WORD PF          :PRINT TYPE
15     002176          000          000          ERRCHR: .BYTE 0,0          :FIRST BYTE LOADED WITH OUTPUT CHARACTER
16     002200          000          000          FFREE:: .BLKW 1          :SECOND BYTE REMAINS ZERO TO STOP OUTPUT
17     002202          000          000          FSIZE:: .BLKW 1          :FIRST FREE WORD IN MEMORY
18     002204          000          000          FMEM: .BLKW 1          :SIZE OF FREE MEMORY IN WORDS
19     002206          000          000          FMEMS: .BLKW 1          :COPY OF FFREE AT END OF INIT SECTION
20     002210          000          000          CTABS: .BLKW 1          :COPY OF FSIZE AT END OF INIT SECTION
21     002212          000          000          CTRLRS: .BLKW 1          :START OF CONTROLLER TABLE STORAGE
22     002214          000          000          TSTTAB: .BLKW 1          :COUNT OF UDA CONTROLLERS IN PTABLES
23     002216          000          000          DMPROG: .BLKW 1          :POINTER TO FIRST CONTROLLER TABLE UNDER TEST
24     002220          000          000          DMEND: .BLKW 1          :START ADDRESS OF DM PROGRAM
25     002222          000          000          DMENDS: .BLKW 1          :END ADDRESS OF DM PROGRAM(FIRST FREE MEMORY ADR)
26     002224          000          000          KTBASA: .BLKW 1          :FREE MEMORY SIZE FROM END OF DM PROGRAM
27     002226          000          000          KTBASO: .BLKW 1          :HIGH TWO BYTES OF BASE ADDRESS FOR KT ACCESS
28     002230          000002          000          IFLAGS:: .BLKW 1          :LOW BYTE OF ADDRESS FOR KT ACCESS
29     002232          000004          000          ICONT ==BIT1          :FLAGS FROM INIT CODE FOR TEST 4
30     002234          000010          000          IREST ==BIT2          : CONTINUE EVENT FLAG
31     002236          000020          000          ISTRT ==BIT3          : RESTART FLAG
32     002240          000000          000          INTRCV: .BLKW 1          : START FLAG
33     002240          000000          000          TNUM: .WORD 0          : START FLAG HOLD FOR T4UPRM ROUTINE
34     002240          000000          000          URUN: .BLKW 1          :NUMBER OF TEST EXECUTING
35     002240          000000          000          URNING: .BLKW 1          :NUMBER OF UNITS TO RUN AT ONE TIME
36     002240          000000          000          UCNT: .BLKW 1          :NUMBER OF UNITS STILL RUNNING
37     002240          000000          000          UCNT: .BLKW 1          :COUNTER OF UNITS UNDER TEST
38     002240          000000          000          INTRCV: .BLKW 1          :INTERRUPT RECEIVED FLAG FOR INT TESTING
    
```

1	002242				FNAME:		
5	002242	132	125	104		.ASC.Z\ZUDDBO.PAK\	:NAME OF DATA FILE
7						.EVEN	
8	002256	000000			FDATA:	.WORD 0	
9	002260	000000			FILOPN:	.WORD 0	:FILE OPEN WHEN NON-ZERO
10	002262				TEMP:	.BLKW 12.	:TEMPORY STORAGE FOR GMANI RESPONSES
11							
12	002312	000001			PAT16C:	.WORD 1	:COUNT OF WORDS IN DATA PATTERN 16
13	002314	000000			PAT16W:	.WORD 0	:WORD SEQUENCE FOR DATA PATTERN 16
14	002316	000000				.WORD 0	
15	002320	000000				.WORD 0	
16	002322	000000				.WORD 0	
17	002324	000000				.WORD 0	
18	002326	000000				.WORD 0	
19	002330	000000				.WORD 0	
20	002332	000000				.WORD 0	
21	002334	000000				.WORD 0	
22	002336	000000				.WORD 0	
23	002340	000000				.WORD 0	
24	002342	000000				.WORD 0	
25	002344	000000				.WORD 0	
26	002346	000000				.WORD 0	
27	002350	000000				.WORD 0	
28	002352	000000				.WORD 0	

```
1          ;CLOCK CONTROL
2
3 002354 000000      KW.CSR: .WORD 0          ;CSR OF CLOCK
4 002356            KW.BRL: .BLKW 1          ;BR LEVEL
5 002360            KW.VEC: .BLKW 1          ;VECTOR
6 002362            KW.HZ: .BLKW 1          ;HERTZ (50. OR 60.)
7 002364            KW.EL: .BLKW 2          ;ELAPSED TIME
8 002370            STIME: .BLKW 2          ;STATISTICAL REPORT TIMER
9
10 002374           NXMAD: .BLKW 1           ;SET TO ALL ONES BY NON-EXISTANT ADDRESS
11 002376 177777     KTMEM: .WORD -1        ;SET TO ALL ONES IF NO KT EXISTS
12
13 002400           T2WRR: .BLKW 1          ;WRITE/READ REGION
14 002402           T2WRO: .BLKW 1          ;WRITE/READ OFFSET
15 002404           T2DR: .BLKW 1          ;DIAGNOSE REGION
16
17
18          ;ERROR LOG CONTROL WORDS
19
20 002406           LBUFS: .BLKW 1          ;START ADDRESS OF LOG/ZERO IF NONE
21 002410           LBUFN: .BLKW 1          ;ADDRESS FOR MORE DATA FOR LOG
22 002412           LBUFE: .BLKW 1          ;LAST ADDRESS AVAILABLE FOR LOG DATA
23
24          ;DISK DIAGNOSTIC DLL CONTROL WORDS
25
26 002414           DLL: .BLKW 1            ;DOWNLINE LOAD RESPONSE CODE = 0 - NO DATA,
27                                     ;1 - PROGRAM PROVIDED, 2- PROGRAM NOT FOUND
28 002416           DLLDR: .BLKW 1         ;DRIVE NUMBER REQUESTING PROGRAM
29 002420           DLLV: .BLKW 1         ;A VALUE FROM DM PROGRAM TO BE RETURNED
30 002422           DLLR: .BLKW 1         ;REGION
31 002424           DLLADR: .BLKW 2        ;ADDRESS WHERE PROGRAM STORED
32 002430           DLLSIZ: .BLKW 1        ;SIZE OF PROGRAM IN BYTES
33 002432           DLLNAM: .BLKW 2        ;NAME OF PROGRAM IN RAD50
```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
17

.SBTTL GLOBAL TEXT SECTION
:++
: THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
: MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
: MORE THAN ONE TEST.
:--

:
: NAMES OF DEVICES SUPPORTED BY PROGRAM
: DEVTYP <LOGICAL DISK DRIVE>

002436
002436
002436 114 117 107

LSDVTYP::
 .ASCIZ /LOGICAL DISK DRIVE/
 .EVEN

: TEST DESCRIPTION
: DESCRIPT <CZUDCA9 UDA & DISK DRV DIAG>

002462
002462
002462 103 132 125

L\$DESC::
 .ASCIZ /CZUDCA9 UDA & DISK
 .EVEN

```

1
2
3 002516      116      125      115 T4BB:  .ASCIZ\NUMBER OF BAD BLOCKS\
4 002543      104      117      040 T4DMN: .ASCIZ\DO YOU WANT TO CHANGE TESTING PARAMETERS FOR THIS DRIVE\
5 002633      102      101      104 T4BBI: .ASCIZ\BAD BLOCK\
6 002645      122      105      101 T4RO:  .ASCIZ\READ ONLY\
7 002657      127      122      111 T4WO:  .ASCIZ\WRITE ONLY\
8 002672      103      110      105 T4WCA: .ASCIZ\CHECK ALL WRITES BY READING\
9 002726      122      101      116 T4WCR: .ASCIZ\RANDOMLY CHECK WRITES BY READING\
10 002767     104      101      124 T4DP:  .ASCIZ\DATA PATTERN - 0 FOR RANDOM SELECTION\
11 003035     105      116      101 T4ECC: .ASCIZ\ENABLE ECC DATA CORRECTION\
12 003070     103      117      115 T4DCA: .ASCIZ\COMPARE ALL DATA READ\
13 003116     122      101      116 T4DCR: .ASCIZ\RANDOMLY COMPARE DATA READ\
14 003151     105      116      101 T4RET: .ASCIZ\ENABLE RETRIES\
15 003170     122      101      116 T4SEK: .ASCIZ\RANDOM ACCESS MODE\
16 003213     040      040      000 T4OPT7: .ASCIZ\ \
17 003216     116      125      115 T4BE:  .ASCIZ\NUMBER OF BEGIN/END SETS\
18 003247     102      105      107 T4BEG: .ASCIZ\BEGIN BLOCK\
19 003263     105      116      104 T4END: .ASCIZ\END BLOCK\
20 003275     116      125      115 T4TRC: .ASCIZ\NUMBER OF TRACKS TO TEST\
21 003326     124      122      101 T4TRAK: .ASCIZ\TRACK\
22 003334     116      125      115 T4GRC: .ASCIZ\NUMBER OF GROUPS TO TEST\
23 003365     107      122      117 T4GRP: .ASCIZ\GROUP\
24 003373     104      117      040 T4CYL: .ASCIZ\DO YOU WISH TO LIMIT THE CYLINDERS TESTED\
25 003445     123      124      101 T4CYLB: .ASCIZ\STARTING CYLINDER\
26 003467     105      116      104 T4CYLE: .ASCIZ\ENDING CYLINDER\
27 003507     116      125      115 T4DPC: .ASCIZ\NUMBER OF WORDS IN DATA PATTERN 16\
28 003552     104      101      124 T4DPD: .ASCIZ\DATA WORD\
29 003564     101      122      105 INITWC: .ASCIZ\ARE YOU SURE CUSTOMER DATA CAN BE DESTROYED\
    
```

; FORMAT STATEMENTS USED IN PRINT CALLS

1					
2					
3	003640	045	124	000	ERRONE: .ASCIZ\%T\
4	003643	045	116	000	ERRNL: .ASCIZ\%N\
5	003646	042	040	040	RNTIM: .ASCIZ\'' RUNTIME 'D16':'\
6	003671	104	071	042	RNTIM1: .ASCIZ\D9':'\
7	003677	104	071	000	RNTIM2: .ASCIZ\D9\
8	003702	042	040	040	ERRME1: .ASCIZ\'' * * * ERROR PROCESSING MESSAGE STRING * * *\N\
9	003771	116	042	122	MXFERP: .ASCIZ\N'REACHED TRANSFER LIMIT - TESTING STOPPED'\N\
10	004046	116	042	125	ERRLIM: .ASCIZ\N'UNIT 'D6' REACHED ERROR LIMIT - WILL NO LONGER BE TESTED'\N\
11	004143	116	042	124	INTST0: .ASCIZ\N'TESTING INTERRUPT ABILITY OF UDA AT ADR 'D16' VEC 'D9'...'N\
12	004240	042	103	117	INTST1: .ASCIZ\''COMPLETED'\N\
13	004255	116	042	103	INITWA: .ASCIZ\N'CUSTOMER DATA WILL BE DESTROYED ON:'NS5'UNIT'S5'UDA AT'S3'DRIVE'\N\
14	004361	045	123	066	INITWB: .ASCIZ\%S6%D2%S6%O6%S4%D3%\N\
15	004406	116	042	115	T4WARN: .ASCIZ\N'MANUAL INTERVENTION NOT ALLOWED. TEST 4 USING DEFAULT PARAMETERS'\N\
16	004513	116	042	124	T4QHED: .ASCIZ\N'THE FOLLOWING QUESTIONS REFER TO UNIT 'D6' UDA AT 'D16' DRIVE 'D9'\N\
17	004620	116	042	104	T4OPT1: .ASCII\N'DO YOU WISH TO:'N\
18	004643	042	040	040	.ASCII\'' 0 - TEST ENTIRE AREA SELECTED'\N\
19	004706	042	040	040	.ASCII\'' 1 - SPECIFY BEGIN/END SETS TO TEST'\N\
20	004756	042	040	040	.ASCII\'' 2 - SPECIFY TRACKS AND CYLINDERS TO TEST'\N\
21	005034	042	040	040	.ASCII\'' 3 - SPECIFY GROUPS AND CYLINDERS TO TEST'\N\
22	005112	042	040	040	.ASCIZ\'' 4 - SPECIFY CYLINDERS TO TEST'\N\
23	005155	042	114	111	INP28A: .ASCIZ\''LIMITS - LO= 0, HI= 268435455'\N\
24	005216	042	111	116	INP28B: .ASCIZ\''INVALID CHAR, TYPE DECIMAL NUMBER 0 TO 268435455'\N\
25	005302	116	042	125	MESSG: .ASCIZ\N'UNIT 'D6' UDA AT 'D16' DRIVE 'D9S'\N\
26	005346	116	042	115	T2WARN: .ASCIZ\N'MANUAL INTERVENTION NOT ALLOWED. TEST 2 RUNNING UNATTENDED'\N\
27	005445	116	042	124	T2CMS1: .ASCII\N'TEST #2 MANUAL INTERVENTION ON UNIT 'D8' UDA AT 'D16' DRIVE 'D9N'\N\
28	005547	042	124	117	.ASCII\''TO WRITE AND READ MEMORY:'N\
29	005603	042	040	040	.ASCII\'' W DATA REGION OFFSET'\N\
30	005634	042	040	040	.ASCII\'' R REGION OFFSET'\N\
31	005660	042	124	117	.ASCII\''TO RUN A DIAGNOSTIC:'N\
32	005707	042	040	040	.ASCII\'' D REGION'\N\
33	005724	042	124	117	.ASCII\''TO EXIT QUESTIONING:'N\
34	005753	042	040	040	.ASCII\'' E'\N\
35	005761	042	104	101	.ASCIZ\''DATA, REGION AND OFFSET ARE HEX VALUES.'N\
36	006034	042	077	040	T2CMS5: .ASCIZ\''? INPUT ERROR'\N\
37	006055	042	116	117	NOCLOCK: .ASCIZ\N'NO LINE CLOCK AVAILABLE FOR TIMING EVENTS'\N\
38	006132	116	042	103	LOGM1: .ASCIZ\N'CONTENTS OF ERROR LOG:'\N\
39	006164	116	042	105	LOGM2: .ASCIZ\N'END OF ERROR LOG'\N\
40	006211	116	042	105	LOGM3: .ASCIZ\N'ERROR LOG IS EMPTY'\N\
41					
42	006240	042	110	117	BASNO: .ASCIZ\''HOST PROGRAM'\N\
44	006257	042	125	116	BASN1: .ASCIZ\''UNIBUS ADDRESSING'\N\
45	006303	042	104	111	BASN2: .ASCIZ\''DISK RESIDENT'\N\
46	006323	042	104	111	BASN3: .ASCIZ\''DISK FUNCTION'\N\
49	006343	042	104	111	BASN4: .ASCIZ\''DISK EXERCISER'\N\
51	006364	042	040	040	BASL1: .ASCIZ\'' DM PC:'D12\
52	006402	042	040	040	BASL2: .ASCIZ\'' UDA AT 'D16\
53	006421	042	040	040	BASL3: .ASCIZ\'' DRIVE 'D9\
54	006436	000			BAS: .BYTE 0 ;NULL TO PRINT NOTHING
55					
56	006437	122	066	122	BASLN: .ASCIZ\R6R6R6R6\ ;USED TO PRINT BASIC LINE OF ERROR MESSAGE

1	006450			X1A:	
2	006450			X2A:	
3	006450			X3A:	
4	006450	042	111	040	X8A: .ASCIZ\''I DON'T LIKE THE ANSWERS YOU GAVE TO THE HARDWARE QUESTIONS'\
5	006547	122	065	122	X1: .ASCIZ\R5R6'UDA HAS MORE THAN ONE VECTOR, BR LEVEL OR BURST RATE'\
6	006643	122	065	122	X2: .ASCIZ\R5R6'TWO UNITS SELECT THE SAME DRIVE'\
7	006712	122	065	122	X3: .ASCIZ\R5R6'MORE THAN EIGHT DRIVES SELECTED ON THIS UDA'\
8	006775	122	064	042	X4: .ASCII\R4'NOT ENOUGH ROOM IN MEMORY TO TEST THE UNITS SELECTED'\
9	007066	042	120	114	.ASCIZ\''PLEASE START PROGRAM OVER AND TEST FEWER UNITS AT A TIME'\
10	007162	122	064	042	X5: .ASCIZ\R4'CHECKSUM ERROR IN DM PROGRAM FILE '\
11	007232	122	064	042	X6: .ASCIZ\R4'TABLE CONSISTANCY ERROR. PLEASE RE-LOAD PROGRAM'\
12	007317	122	064	042	X7: .ASCIZ\R4'ERROR IN DM PROGRAM FILE. DM PROGRAM NOT FOUND'\
13	007403	122	065	122	X8: .ASCIZ\R5R6'TWO UDA'S USE THE SAME VECTOR'\
14	007450			X20:	
15	007450	122	065	042	X38: .ASCII\R5'MEMORY ERROR TRYING TO READ UDA REGISTERS'\
16	007526	042	103	110	.ASCII\''CHECK UNIBUS SELECTION SWITCHES ON UDA MODULE M7161'\
17	007614	042	117	122	.ASCII\''OR UNIBUS'\
18	007630	042	117	122	.ASCIZ\''OR 'R7'\
19	007640	122	065	042	X21: .ASCII\R5'UDA RESIDENT DIAGNOSTICS DETECTED FAILURE'NR8\
20	007720	042	122	105	.ASCIZ\''REPLACE UDA MODULE M716'02N\
21	007755	122	065	042	X22: .ASCII\R5'STEP BIT DID NOT SET IN UDASA REGISTER DURING INITIALIZATION'\
22	010056	042	123	124	.ASCIZ\''STEP BIT EXPECTED '016NR8R7\
23	010113	122	065	042	X23A: .ASCII\R5'UDA DID NOT CLEAR RING STRUCTURE IN HOST MEMORY DURING INITIALIZATION'\
24	010225	104	071	042	.ASCII\D9' WORDS WERE TO BE CLEARED STARTING AT ADDRESS '016N\
25	010313	042	106	111	.ASCII\''FIRST SEVERAL WORDS NOT CLEARED (UP TO 6):'\
26	010370	123	066	042	.ASCIZ\S6'ADDRESS'S4'CONTENTS'\
27	010421	123	067	117	X23B: .ASCIZ\S7016S5016N\
28	010435	122	065	042	X24: .ASCII\R5'UDASA REGISTER DID NOT GO TO ZERO AFTER STEP 3 WRITE OF INITIALIZATION'\
29	010550	042	120	125	.ASCIZ\''PURGE/POLE DIAGNOSTICS WERE REQUESTED'NR8R7\
30	010625	122	065	042	X25: .ASCII\R5'UDA DID NOT RETURN CORRECT DATA IN UDASA REGISTER DURING INITIALIZATION'\
31	010741	042	040	040	.ASCIZ\'' UDASA EXPECTED '016NR8R7\
32	010776	122	065	042	X26: .ASCII\R5'DATA COMPARISON ERROR DURING DIAGNOSTIC PORT LOOP TEST'\
33	011071	042	040	040	.ASCII\'' DATA SENT TO UDASA '016N\
34	011125	042	040	040	.ASCIZ\'' RECEIVED FROM UDASA '016NR7\
35	011164	122	065	042	X27: .ASCII\R5'UDASA REGISTER DID NOT CHANGE AFTER WRITING TO IT'\
36	011252	042	111	116	.ASCIZ\''IN PORT LOOP DIAGNOSTIC'NR8R7\
37	011311	122	065	042	X28: .ASCIZ\R5'UDA DID NOT INTERRUPT THE PDP-11'NR7\
38	011361	122	065	042	X29: .ASCII\R5'UDA INTERRUPTED AT DIFFERENT BR LEVEL THAN SPECIFIED IN HARDWARE'\
39	011466	042	121	125	.ASCII\''QUESTIONS. INTERRUPT WAS AT BR LEVEL '03N\
40	011540	042	103	110	.ASCII\''CHECK PRIORITY PLUG ON UDA MODULE M7161'\
41	011612	042	117	122	.ASCIZ\''OR CHANGE HARDWARE QUESTIONS'\
42	011652	122	065	042	X30: .ASCIZ\R5'UDA REPORTED FATAL ERROR IN UDASA REGISTER WHILE RUNNING DM PROGRAM'NR8\
43	011765	122	065	042	X31: .ASCII\R5'NO INTERRUPT RECEIVED FROM DM PROGRAM FOR 3 MINUTES'\
44	012055	042	101	123	.ASCIZ\''ASSUME PROGRAM IS HUNG'\
45	012107	122	065	042	X32: .ASCIZ\R5'MESSAGE BUFFER RECEIVED FROM DM PROGRAM WITH UNKNOWN REQUEST NUMBER'\
46	012220	122	065	042	X35: .ASCIZ\R5'DM PROGRAM ASKED FOR DATA ON UNKNOWN DRIVE'\
47	012300	122	065	042	X36: .ASCII\R5'NO INTERRUPT RECEIVED FROM UDA FOR 30 SECONDS'\
48	012362	042	127	110	.ASCIZ\''WHILE LOADING DM PROGRAM'\
49	012416	122	065	042	X37: .ASCIZ\R5'UDA REPORTED FATAL ERROR IN UDASA REGISTER WHILE LOADING DM PROGRAM'NR8R7\

1	012533	042	115	105	XMSG1:	.ASCIZ\MESSAGE BUFFER CONTAINS:'N\
2	012567	123	063	117	XMSG2:	.ASCIZ\S3016S1016S1016S1016S1016S1016S1016N\
3	012634	122	065	042	XPKT1:	.ASCII\R5'RESPONSE PACKET FROM UDA DOES NOT CONTAIN EXPECTED DATA'N\
4	012730	042	105	111		.ASCII\EITHER UDA RETURNED ERROR STATUS OR PACKET WAS NOT RECEIVED CORRECTLY'N\
5	013040	123	063	042		.ASCIZ\S3'COMMAND PACKET SENT'S6'RESPONSE PACKET RECEIVED'N\
6	013125	123	066	117	XPKT2:	.ASCIZ\S6016S1016S14016S1016N\
7	013154	042	040	040	XSA:	.ASCIZ\ UDASA CONTAINS '016N\
8	013205	042	122	105	XFRU:	.ASCIZ'REPLACE UDA MODULE M7161'N\
12						.EVEN
13						


```

1      .SBTTL GLOBAL ERROR REPORT SECTION
2
3
4      :++
5      : THE GLOBAL ERROR REPORT SECTION CONTAINS MESSAGE PRINTING AREAS
6      : USED BY MORE THAN TEST TO OUTPUT ADDITIONAL ERROR INFORMATION. PRINTB
7      : (BASIC) AND PRINTX (EXTENDED) CALLS ARE USED TO CALL PRINT SERVICES.
8      :--
9      :
10     : LIST INSTRUCTIONS, SHIFTED RIGHT
11     : LIST TEST TAGS, SHIFTED RIGHT
12     : LIST SUBTEST TAGS, SHIFTED RIGHT
13     : LIST GLOBAL TAGS, SHIFTED RIGHT
14     : LIST OTHER TAGS, SHIFTED RIGHT
15
16     177777
17     177777
18     177777
19     177777
20     177777
21
22     013242 BGNMSG ERR001
23     013242 PNTB X1,#X1A
24     013242 012746 000450
25     013246 004137 022276
26     013252 006547
27     013254 000002
28     013256 ENDMSG
29
30     013260 BGNMSG ERR002
31     013260 PNTB X2,#X2A
32     013260 012746 006450
33     013264 004137 022276
34     013270 006643
35     013272 000002
36     013274 ENDMSG
37
38     013276 BGNMSG ERR003
39     013276 PNTB X3,#X3A
40     013276 012746 006450
41     013302 004137 022276
42     013306 006712
43     013310 000002
44     013312 ENDMSG
45
46     013314 BGNMSG ERR004
47     013314 PNTB X4
48     013314 004137 022276
49     013320 006775
50     013322 000000
51     013324 ENDMSG
52
53     013326 BGNMSG ERR005
54     013326 PNTB X5
55     013326 004137 022276
56     013332 007162
57     013334 000000
58     013336 ENDMSG
59
60     013340 BGNMSG ERR006
61     013340 PNTB X6
62     013340 004137 022276
63     013344 007232
64     013346 000000
65     013350 ENDMSG
    
```

```

MOV #X1A,-(SP)
JSR R1,LPNTB
.WORD X1
.WORD PNT.CT

MOV #X2A,-(SP)
JSR R1,LPNTB
.WORD X2
.WORD PNT.CT

MOV #X3A,-(SP)
JSR R1,LPNTB
.WORD X3
.WORD PNT.CT

JSR R1,LPNTB
.WORD X4
.WORD PNT.CT

JSR R1,LPNTB
.WORD X5
.WORD PNT.CT

JSR R1,LPNTB
.WORD X6
.WORD PNT.CT
    
```


37					
38	013352			BGNMSG	ERR007
39	013352				PNTB X7
	013352	004137	022276		
	013356	007317			
	013360	000000			
40	013362			ENDMSG	
41					
42	013364			BGNMSG	ERR008
43	013364				PNTB X8,#X8A
	013364	012746	006450		
	013370	004137	022276		
	013374	007403			
	013376	000002			
44	013400			ENDMSG	
45					
46	013402			BGNMSG	ERR020
47	013402				PNTB X20
	013402	004137	022276		
	013406	007450			
	013410	000000			
48	013412			ENDMSG	
49					
50	013414			BGNMSG	ERR021
51	013414	010201			MOV R2,R1
52	013416	000301			SWAB R1
53	013420				AND 2,R1
	013420	042701	177775		
54	013424	006201			
55	013426	005201			ASR R1
56	013430				INC R1
	013430	010146			PNTB X21,R2,R1
	013432	010246			
	013434	004137	022276		
	013440	007640			
	013442	000004			
57	013444			ENDMSG	
58					
59	013446			BGNMSG	ERR022
60	013446	042737	100000 024604		BIC #SA.ERR,UDARSD
61	013454				PNTB X22,UDARSD,R2
	013454	010246			
	013456	013746	024604		
	013462	004137	022276		
	013466	007755			
	013470	000004			
62	013472				
63	013512			ENDMSG	PRINTX #XFRU
64					
65	013514			BGNMSG	ERR023
66	013514				PNTB X23A,R1,FFREE
	013514	013746	002176		
	013520	010146			
	013522	004137	022276		
	013526	010113			
	013530	000004			
67	013532	005742			TST -(R2)

JSR R1,LPNTB
 .WORD X7
 .WORD PNT.CT

MOV #X8A,-(SP)
 JSR R1,LPNTB
 .WORD X8
 .WORD PNT.CT

JSR R1,LPNTB
 .WORD X20
 .WORD PNT.CT

BIC #^C<2>,R1

MOV R1,-(SP)
 MOV R2,-(SP)
 JSR R1,LPNTB
 .WORD X21
 .WORD PNT.CT

MOV R2,-(SP)
 MOV UDARSD,-(SP)
 JSR R1,LPNTB
 .WORD X22
 .WORD PNT.CT

MOV FFREE,-(SP)
 MOV R1,-(SP)
 JSR R1,LPNTB
 .WORD X23A
 .WORD PNT.CT

68	013534	005712		ERR23A: TST (R2)	
69	013536	001410		BEQ ERR23B	
70	013540			PNTB X23B,R2,(R2)	
	013540	011246			MOV (R2),-(SP)
	013542	010246			MOV R2,-(SP)
	013544	004137	022276		JSR R1,LPNTB
	013550	010421			.WORD X23B
	013552	000004			.WORD PNT.CT
71	013554	005304		DEC R4	
72	013556	001403		BEQ ERR23C	
73	013560	005722		ERR23B: TST (R2)+	
74	013562	005303		DEC R3	
75	013564	001363		BNE ERR23A	
76	013566			ERR23C: PNTB XFRU	
	013566	004137	022276		JSR R1,LPNTB
	013572	013205			.WORD XFRU
	013574	000000			.WORD PNT.CT
77	013576			ENDMSG	
78					
79	013600			BGNMSG ERR024	
80	013600			PNTB X24,R2	
	013600	010246			MOV R2,-(SP)
	013602	004137	022276		JSR R1,LPNTB
	013606	010435			.WORD X24
	013610	000002			.WORD PNT.CT
81	013612			ENDMSG	
82					
83	013614			BGNMSG ERR025	
84	013614			PNTB X25,R1,R2	
	013614	010246			MOV R2,-(SP)
	013616	010146			MOV R1,-(SP)
	013620	004137	022276		JSR R1,LPNTB
	013624	010625			.WORD X25
	013626	000004			.WORD PNT.CT
85	013630			ENDMSG	
86					
87	013632			BGNMSG ERR026	
88	013632			PNTB X26,R2,2(R4)	
	013632	016446	000002		MOV 2(R4),-(SP)
	013636	010246			MOV R2,-(SP)
	013640	004137	022276		JSR R1,LPNTB
	013644	010776			.WORD X26
	013646	000004			.WORD PNT.CT
89	013650			ENDMSG	
90					
91	013652			BGNMSG ERR027	
92	013652			PNTB X27,2(R4)	
	013652	016446	000002		MOV 2(R4),-(SP)
	013656	004137	022276		JSR R1,LPNTB
	013662	011164			.WORD X27
	013664	000002			.WORD PNT.CT
93	013666			ENDMSG	
94					
95	013670			BGNMSG ERR028	
96	013670			PNTB X28	
	013670	004137	022276		JSR R1,LPNTB
	013674	011311			.WORD X28

133	014026			BGNMSG ERR037		
134	014026			PNTB X37,R1		
	014026	010146				MOV R1,-(SP)
	014030	004137	022276			JSR R1,LPNTB
	014034	012416				.WORD X37
	014036	000002				.WORD PNT.CT
135	014040			ENDMSG		
136						
137	014042			BGNMSG ERR038		
138	014042			PNTB X38		
	014042	004137	022276			JSR R1,LPNTB
	014046	007450				.WORD X38
	014050	000000				.WORD PNT.CT
139	014052			ENDMSG		
140						
141	014054			PNTPKT: PNTB XPKT1		
	014054	004137	022276			JSR R1,LPNTB
	014060	012634				.WORD XPKT1
	014062	000000				.WORD PNT.CT
142	014064	010401		MOV R4,R1		
143	014066	062701	000104	ADD #HC.CPK,R1		
144	014072	010402		MOV R4,R2		
145	014074	062702	000020	ADD #HC.MPK,R2		
146	014100	012703	000014	MOV #12.,R3		
147	014104			PNTPKL: PNTB XPKT2,2(R1),(R1),2(R2),(R2)		
	014104	011246				MOV (R2),-(SP)
	014106	016246	000002			MOV 2(R2),-(SP)
	014112	011146				MOV (R1),-(SP)
	014114	016146	000002			MOV 2(R1),-(SP)
	014120	004137	022276			JSR R1,LPNTB
	014124	013125				.WORD XPKT2
	014126	000010				.WORD PNT.CT
148	014130	062701	000004	ADD #4,R1		
149	014134	062702	000004	ADD #4,R2		
150	014140	005303		DEC R3		
151	014142	001360		BNE PNTPKL		
152	014144	000207		RETURN		
153						
154	014146			MSGPKT: PNTB XMSG1		
	014146	004137	022276			JSR R1,LPNTB
	014152	012533				.WORD XMSG1
	014154	000000				.WORD PNT.CT
155	014156	016504	000016	MOV C.RING(R5),R4		
156	014162	062704	000272	ADD #HC.BF2,R4		
157	014166	012703	000005	MOV #5,R3		
158	014172			MSGPKL: PNTB XMSG2,(R4),2(R4),4(R4),6(R4),8.(R4),10.(R4),12.(R4)		
	014172	016446	000014			MOV 12.(R4),-(SP)
	014176	016446	000012			MOV 10.(R4),-(SP)
	014202	016446	000010			MOV 8.(R4),-(SP)
	014206	016446	000006			MOV 6(R4),-(SP)
	014212	016446	000004			MOV 4(R4),-(SP)
	014216	016446	000002			MOV 2(R4),-(SP)
	014222	011446				MOV (R4),-(SP)
	014224	004137	022276			JSR R1,LPNTB
	014230	012567				.WORD XMSG2
	014232	000016				.WORD PNT.CT
159	014234	062704	000016	ADD #14.,R4		

160 014240 005303
161 014242 001353
162 014244 000207

DEC R3
BNE MSGPKL
RETURN

1	014246			BGNMSG ERRRTN		:ERROR REPORT ROUTINE
2	014246	013702	002230	MOV TNUM,R2		:GET TEST NUMBER
3	014252	006302		ASL R2		:DOUBLE
4	014254	012703	006421	MOV #BASL3,R3		:GET ADDRESS OF DRIVE PRINT LINE
5	014260	005764	000004	TST 4(R4)		:CHECK IF DRIVE NUMBER GIVEN
6	014264	100002		BPL 1\$:BRANCH IF SO
7	014266	012703	006436	MOV #BAS,R3		
8	014272			1\$: PNTB BASLN,TNAMES-2(R2),#BASL1,(R4),#BASL2,(R5),R3,4(R4)		
	014272	016446	000004			MOV 4(R4),-(SP)
	014276	010346				MOV R3,-(SP)
	014300	011546				MOV (R5),-(SP)
	014302	012746	006402			MOV #BASL2,-(SP)
	014306	011446				MOV (R4),-(SP)
	014310	012746	006364			MOV #BASL1,-(SP)
	014314	016246	020322			MOV TNAMES-2(R2),-(SP)
	014320	004137	022276			JSR R1,LPNTB
	014324	006437				.WORD BASLN
	014326	000016				.WORD PNT.CT
9	014330			ASSUME C.UADR EQ 0		
10	014330	004737	025356	CALL RNTIME		:GET RUNTIME PARAMETERS
11	014334			PRINT #CR		:ADVANCE TO NEW LINE
	014334	112700	000015			MOVB #CR,R0
	014340	004737	022114			CALL CPNT
12	014344	062704	000006	ADD #6,R4		:INCREASE R4 TO POINT TO MESSAGE POINTER
13	014350	012402		MOV (R4)+,R2		:GET MESSAGE POINTER
14	014352	006302		ASL R2		:DOUBLE TO MAKE BYTE OFFSET
15	014354	063702	002214	ADD DMPROG,R2		:ADD TO START OF MESSAGE STRINGS
16	014360	067702	165630	ADD @DMPROG,R2		:ADD SIZE OF MAIN PROGRAM
17	014364	105712		TSTB (R2)		:CHECK FIRST BYTE
18	014366	001001		BNE NCON		:IF ZERO
19	014370	005202		INC R2		: INCREMENT TO NEXT BYTE
20	014372	012737	022214	002172 NCON: MOV #PX,Ptype		:CHANGE TO EXTENDED OUTPUT
21	014400	004737	020022	CALL OSTRNG		:OUTPUT ACCORDING TO STRING
22	014404			ENDMSG		

1	000001	SVCINS= 1
2	000001	SVCTST= 1
3	000001	SVCSUB= 1
4	000001	SVCGBL= 1
5	000001	SVCTAG= 1

: LIST INSTRUCTIONS, SHIFTED RIGHT
: LIST TEST TAGS, SHIFTED RIGHT
: LIST SUBTEST TAGS, SHIFTED RIGHT
: LIST GLOBAL TAGS, SHIFTED RIGHT
: LIST OTHER TAGS, SHIFTED RIGHT

1
2
3
4
5
6
7

.SBTTL GLOBAL SUBROUTINES SECTION
:MEMORY ALLOCATION ERROR
:THIS ROUTINE PRINTS A SYSTEM FATAL ERROR AND EXITS THE TEST
FMERR: ERRSF 4,,ERR004

014406
014406 104454
014410 000004
014412 000000
014414 013314
8 014416
014416 104444

DOCLN

:ABORT

TRAP CSERSF
.WORD 4
.WORD 0
.WORD ERR004
TRAP CSDCLN

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16 014420  
17 014420 013746 002176  
18 014424 160137 002200  
19 014430 002766  
20 014432 060101  
21 014434 060137 002176  
22 014440 012601  
23 014442 000207  
:  
:ALOCM  
:  
:ALLOCATE A BLOCK OF FREE MEMORY. REPORT ERROR IF MEMORY EXHAUSTED.  
:  
:INPUTS:  
: R1 - NUMBER OF WORDS TO ALLOCATE  
: FFREE - FIRST FREE WORD IN MEMORY  
: FSIZE - SIZE OF FREE MEMORY AVAILABLE IN WORDS  
:OUTPUTS:  
: R1 - ADDRESS OF FIRST WORD OF ALLOCATED MEMORY  
: FFREE - NEW FIRST FREE WORD IN MEMORY  
: FSIZE - SIZE OF FREE MEMORY LEFT AFTER ALLOCATION  
:SYSTEM FATAL ERROR WILL BE REPORTED IF NOT ENOUGH MEMORY AVAILABLE  
:AND ENTIRE PROGRAM WILL BE STOPPED.  
:  
ALOCM: PUSH FFREE ;SAVE FFREE AT ENTRY  
:MOV FFREE,-(SP)  
SUB R1,FSIZE ;REDUCE SIZE OF FREE MEMORY  
BLT FMERR ;REPORT ERROR IF NOT ENOUGH MEMORY  
ADD R1,R1 ;CHANGE WORDS TO BYTES  
ADD R1,FFREE ;CALCULATE NEW START OF FREE MEMORY  
POP R1 ;GET START OF ALLOCATED MEMORY  
:MOV (SP)+,R1  
RETURN
```


1
2
3
4
5
6
7
8
9
10
11
12
13
14 014444 012701 000200
15 014450 004737 014420
16 014454 010104
17 014456 010465 000016
18 014462 062701 000020
19 014466 010164 000004
20 014472 062701 000064
21 014476 010164 000010
22 014502 000207

```
:HCOMM
:
: ALLOCATES MEMORY FOR HOST COMM AREA AND PACKET BUFFERS WITH ONE
: DESCRIPTOR IN EACH RING. TO BE CALLED AFTER INITIALIZING
: A CONTROLLER WITH SA.MSG=0 AND SA.CMD=0.
:
: INPUTS:
: R5 - ADDRESS OF CONTROLLER TABLE
:
: OUTPUTS:
: CONTROLLER TABLE POINTING TO HOST COMM AREA
: RING POINTERS TO PACKETS
: R4 - ADDRESS OF HOST COMM AREA
:
HCOMM: MOV #HC.SIZ/2,R1      ;GET SIZE OF AREA TO ALLOCATE
        CALL ALOCM         ;ALLOCATE THE MEMORY
        MOV R1,R4          ;GET ADDRESS OF HOST COMM AREA
        MOV R4,C.RING(R5)  ;PLACE IN CONTROLLER TABLE
        ADD #HC.MPK,R1     ;COMPUTE START OF MESSAGE PACKET
        MOV R1,HC.MSG(R4)  ;PLACE IN RING
        ADD #<HC.CPK-HC.MPK>,R1 ;COMPUTE START OF COMMAND PACKET
        MOV R1,HC.CMD(R4)  ;PLACE IN RING
        RETURN
```

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14 014504 004737 025334  
15 014510 005037 002406  
16 014514 013737 002202 002176  
17 014522 013737 002204 002200  
18 014530 020137 002230  
19 014534 001007  
20 014536 013737 002216 002176  
21 014544 013737 002220 002200  
22 014552 000207  
23  
24 014554 004737 014566  
25 014560 010137 002230  
26 014564 000207
```

:TINIT
:INITIALIZE VARIABLES FOR TEST
:INPUTS:
R1 - TEST NUMBER
:OUTPUTS:
LBUFS - CLEARED (DELETES ERROR LOG)
FFREE - FROM FMEM
FSIZE - FROM FMEMS
TNUM - TEST NUMBER FROM R1
ALL REGISTERS CLOBERED

TINIT: CALL RESET ;RESET ALL DEVICES
CLR LBUFS ;CLEAR ERROR LOG BUFFER POINTER
MOV FMEM,FFREE ;INIT FFREE
MOV FMEMS,FSIZE ;INIT FSIZE
CMP R1,TNUM ;SEE IF SAME TEST RUNNING
BNE TINITR ;IF NOT, GO TO READ DM PROGRAM
MOV DMEND,FFREE ;CHANGE FREE MEMORY TO LEAVE
MOV DMENDS,FSIZE ; DM PROGRAM ALLOCATED
RETURN

TINITR: CALL READDM ;READ DM PROGRAM
MOV R1,TNUM ;STORE TEST NUMBER TO SHOW DM PROGRAM IN MEMORY
RETURN

```

2
3
4
5
6
7
8
9
10
11
12
13
14 014566 013737 002176 002214 READDM: MOV FFREE,DMPROG          ;GET STORAGE ADDRESS
15 014574 004737 024666          CALL RDREC
16 014600 103407          BCS README           ;CHECK IF ERROR
17 014602 013737 002176 002216 MOV FFREE,DMEND      ;SAVE END OF ADDRESS OF DM PROGRAM
18 014610 013737 002200 002220 MOV FSIZE,DMENDS    ; AND CURRENT SIZE OF FREE MEMORY
19 014616 000207          RETURN
20
21 014620          README: ERRSF 7,,ERR007          ;REPORT DM PROGRAM NOT FOUND
    014620 104454          TRAP CSERSF
    014622 000007          .WORD 7
    014624 000000          .WORD 0
    014626 013352          .WORD ERR007
22 014630          DOCLN
    014630 104444          TRAP CSDCLN
    
```



```

1      :RUNDM
2
3      :LOAD AND RUN A DM PROGRAM IN THE CONTROLLERS. RETURN WHEN ALL
4      :DM PROGRAMS HAVE TERMINATED.
5
6      :INPUTS:
7          TSTTAB - POINTER TO FIRST CONTROLLER TABLE
8          R1 - NUMBER OF CONTROLLERS TO TEST
9
10     :IMPLICIT INPUTS:
11         DMPROG - POINTER TO START OF DM PROGRAM IN MEMORY
12
13     :OUTPUTS:
14         Z SET IF NO CONTROLLERS SUCCESSFULLY STARTED
15         ALL REGISTERS ARE USED AND PREVIOUS CONTENTS DESTROYED.
16
15 014632 010137 002232  RUNDM:  MOV R1,URUN          ;SAVE NUMBER OF UNITS TO RUN
16 014636 005037 002234          CLR URNING        ;CLEAR NUMBER OF UNITS RUNNING
17
18     :LOAD DM PROGRAM INTO EACH CONTROLLER
19
20 014642 013737 002232 002236      MOV URUN,UCNT      ;SET COUNTER OF UNITS
21 014650 013705 002212          MOV TSTTAB,R5     ;GET FIRST CONTROLLER TABLE
22 014654          LDDM:
26 014654 005065 000014          CLR C.FLG(R5)    ;CLEAR ALL FLAGS
28 014660 116537 000002 002074      MOVB C.UNIT(R5),L$LUN ;SEE IF UNIT TO BE TESTED
29 014666 005765 000002          TST C.UNIT(R5)
30 014672 100405          BMI LDNEXT        ;IF NOT, DON'T LOAD THIS UNIT
31 014674          ASSUME CT.AVL EQ BIT15
32 014674 004737 022532          CALL LOADDM      ;LOAD THE DM PROGRAM
33 014700 001402          BEQ LDNEXT        ;IF ERROR, GO TO NEXT CONTROLLER
34 014702 005237 002234          INC URNING      ;IF NO ERROR, COUNT UNIT RUNNING
35 014706 062705 000046      LDNEXT:  ADD #C.SIZE,R5 ;MOVE TO NEXT CONTROLLER TABLE
36 014712 005337 002236          DEC UCNT        ;CHECK IF MORE CONTROLLERS
37 014716 001356          BNE LDDM        ;LOAD NEXT
38
39     :CHECK IF ANY CONTROLLERS LOADED
40
41 014720 005737 002234          TST URNING      ;ANY UNITS LOADED?
42
43     :THE DM PROGRAMS ARE NOW IN CONTROL
44     :RESPDM MUST BE CALLED TO RESPOND TO THEIR REQUESTS
45
46 014724 000207          RETURN
    
```

```

1      :RESPDM
2
3
4      :RESPOND TO DM REQUESTS. RETURN WHEN ALL DM PROGRAMS
5      :HAVE TERMINATED.
6 014726 013705 002212      RESPDM: MOV TSTTAB,R5      ;GET CONTROLLER TABLE ADDRESS
7 014732 013737 002232 002236  MOV URUN,UCNT      ;SET COUNTER OF UNITS
8 014740 016504 000016      RESPCT: MOV C.RING(R5),R4  ;GET HOST COMM AREA ADDRESS
9 014744 032765 000002 000014  BIT #CT.RN,C.FLG(R5)  ;CHECK IF PROGRAM RUNNING
10 014752 001446      BEQ RSPNXT      ;IF NOT, LOOK AT NEXT
11 014754 116537 000002 002074  MOVB C.UNIT(R5),L$LUN  ;STORE UNIT NUMBER UNDER TEST
12 014762 032765 000010 000014  BIT #CT.MSG,C.FLG(R5)  ;SEE IF INTERRUPT RECEIVED
13 014770 001071      BNE RSPIN      ;IF SO, LOOK AT PACKET
14 014772 032765 000004 000014  BIT #CT.CMD,C.FLG(R5)  ;SEE IF COMMAND HAS BEEN SENT
15 015000 001520      BEQ RSPDU      ;IF NOT, SEND ONE
16
17      :CHECK IF UDA STILL RUNNING
18
19 015002 011503      MOV (R5),R3      ;GET ADDRESS OF UDAIP
20 015004 016301 000002      MOV 2(R3),R1      ;LOOK AT UDASA REGISTER
21 015010 001405      BEQ RSPTM      ;IF ZERO, UDA STILL RUNNING
22 015012      ERRDF 30,,ERR030 ;REPORT UDA HAS FATAL ERROR
23 015012 104455      TRAP CSERDF
24 015014 000036      .WORD 30
25 015016 000000      .WORD 0
26 015020 013716      .WORD ERR030
27 015022 000445      BR RSPDRP      ;DROP CONTROLLER FROM TESTING
28
29      :CHECK FOR TIMEOUT OF RESPONSE
30
31 015024      RSPTM:
32 015024 005737 002354      TST KW.CSR      ;SEE IF A CLOCK ON SYSTEM
33 015030 001416      BEQ RSPNTO      ;DON'T TIME IF NO CLOCK
34 015032 023765 002366 000042  CMP KW.EL+2,C.TOH(R5) ;COMPARE TO TIMEOUT COUNTER
35 015040 101005      BHI RSPTMO
36 015042 001011      BNE RSPNTO
37 015044 023765 002364 000040  CMP KW.EL,C.TO(R5)
38 015052 103405      BLO RSPNTO
39 015054      RSPTMO: ERRDF 31,,ERR031 ;IF TOO MUCH TIME ELAPSED SINCE LAST INTERRUPT
40 015054 104455      ;REPORT TIMEOUT ERROR
41 015056 000037      TRAP CSERDF
42 015060 000000      .WORD 31
43 015062 013732      .WORD 0
44 015064 000424      .WORD ERR031
45 015066      BR RSPDRP      ;DROP CONTROLLER FROM TESTING
46 015066      RSPNTO: BREAK      ;ALLOW DRS TO SEE TERMINAL INPUT
47 015066 104422      TRAP CSBRK
    
```



```
1          ;CHECK FOR TIME TO PRINT STATISTICAL REPORT
2
3 015070 005737 002354      RSPNXT: TST KW.CSR          ;ANY CLOCK ON SYSTEM?
4 015074 001412              BEQ RSPNRP          ;BYPASS IF NOT
5 015076 023737 002366 002372  CMP KW.EL+2,STIME+2      ; A STATISTICAL REPORT
6 015104 101005              BHI RSPRPT
7 015106 001005              BNE RSPNRP
8 015110 023737 002364 002370  CMP KW.EL,STIME
9 015116 103401              BLO RSPNRP
10 015120              RSPRPT: DORPT          ;PRINT THE REPORT
    015120 104424              TRAP      C$DRPT
11
12          ;SWITCH TO NEXT CONTROLLER
13
14 015122 062705 000046      RSPNRP: ADD #C.SIZE,R5      ;MOVE TO NEXT TABLE
15 015126 005337 002236      DEC UCNT          ;CHECK IF MORE CONTROLLERS
16 015132 001302              BNE RESPCT      ;LOOK AT NEXT CONTROLLER
17 015134 000674              BR RESPDM        ;LOOK AT FIRST CONTROLLER AGAIN
18
19          ;REMOVE A CONTROLLER FROM TESTING
20
21 015136 042765 000012 000014 RSPDRP: BIC #CT.RN+CT.MSG,C.FLG(R5) ;CLEAR PROGRAM RUNNING
22 015144 005337 002234      DEC URNING      ;REDUCE RUNNING CONTROLLERS COUNT
23 015150 001347              BNE RSPNXT      ;IF ANY STILL RUNNING, LOOK AT THEM
24 015152 000207              RETURN          ;ELSE RETURN TO TEST SECTION
```



```

1          ;CONTROLLER HAS RESPONDED, LOOK AT MESSAGE PACKET
2
3          ;CHECK FOR PROPER OPCODE IN END PACKET
4
5 015154 012700 000204          RSPIN: MOV #OP.END+OP.SSD,R0          ;GET SEND DATA END PACKET OPCODE
6 015160 032765 000020 000014  BIT #CT.REQ,C.FLG(R5)          ;LOOK IF SEND DATA OR RECEIVE DATA
7 015166 001402          BEQ RSPMWR
8 015170 012700 000205          MOV #OP.END+OP.RSD,R0          ;CHANGE TO RECEIVE DATA END PACKET OPCODE
9 015174 120064 000030          RSPMWR: CMPB R0,HC.MPK+P.OPCD(R4)      ;COMPARE TO OPCODE IN END PACKET
10 015200 001010          BNE RSPERR
11
12          ;LOOK AT STATUS CODE
13
14 015202 032764 000037 000032  BIT #ST.MSK,HC.MPK+P.STS(R4)      ;CHECK FOR STATUS CODE ST.SUC (ZERO)
15 015210 001004          BNE RSPERR
16
17          ;CHECK FOR EXPECTED REFERENCE NUMBER
18
19 015212 026564 000044 000020  CMP C.REF(R5),HC.MPK+P.CRF(R4)    ;CHECK IF CORRECT REF NUMBER
20 015220 001405          BEQ RSPPTW
21 015222          RSPERR: ERRDF 33,,ERR033
22          TRAP C$ERDF
23          .WORD 33
24          .WORD 0
25          .WORD ERR033
26          BR RSPDRP          ;DROP UNIT FROM TESTING
27
28          ;CHECK IF RESPONSE FROM SEND OR RECEIVE DATA COMMAND
29
30 015234 032765 000020 000014  RSPPTW: BIT #CT.REQ,C.FLG(R5)      ;CHECK IF RESPONSE FROM DM PROGRAM
31 015242 001445          RSPOU: BEQ RSPOUT          ;LOOK AT REQUEST NUMBER IF SO
    
```

```

1      ;MAINTENANCE READ END PACKET RECEIVED, LOOK AT REQUEST FROM DM PROGRAM
2
3 015244 016401 000272      RSPPT2: MOV HC.BF2(R4),R1      ;GET REQUEST NUMBER
4 015250 042701 007777      BIC #007777,R1      ;CHECK TYPE
5 015254 022701 060000      CMP #DU.SPC,R1      ;IS SPECIAL TYPE SET?
6 015260 001010      BNE 1$      ;IF NOT, ERROR
7 015262 042764 170000 000272      BIC #^C007777,HC.BF2(R4)      ;CLEAR TYPE
8 015270 016401 000272      MOV HC.BF2(R4),R1      ;GET REQUEST NUMBER
9 015274 020127 000017      CMP R1,#DPSIZ      ;CHECK IF IN EXPECTED RANGE
10 015300 103405      BLO RSPPT3
11 015302      1$: ERRDF 32,,ERR032      ;BAD REQUEST NUMBER
      TRAP CSERDF
      .WORD 32
      .WORD 0
      .WORD ERR032
12 015312 000711      BR RSPDRP      ;DROP UNIT FROM TESTING
13
14 015314 012700 000004      RSPPT3: MOV #OP.SSD,R0      ;BUILD A SEND DATA COMMAND PACKET
15 015320 004737 023150      CALL BLDCMD      ; FOR ANSWER TO DM PROGRAM
16 015324 012700 000164      MOV #HC.BF1,R0      ;POINT TO BUFFER IN PACKET
17 015330 004737 023332      CALL CLRBUF      ; AND CLEAR BUFFER
18 015334 010403      MOV R4,R3      ;R3 POINTS TO COMMAND BUFFER
19 015336 062704 000106      ADD #HC.BSZ,R4      ;R4 POINTS TO MESSAGE BUFFER
20 015342 011401      MOV (R4),R1      ;GET REQUEST NUMBER
21 015344 012423      MOV (R4)+,(R3)+      ;PUT REQUEST NUMBER INTO COMMAND PACKET
22 015346 060101      ADD R1,R1      ;DOUBLE REQUEST NUMBER
23 015350 004771 015460      CALL @RSPDSP(R1)      ;CALL REQUESTED ROUTINE
24 015354 001270      BNE RSPDRP      ;ROUTINE RETURNS Z CLEAR TO DROP UNIT FROM TESTING
25      ; Z SET IF COMMAND READY TO SEND TO UNIT
26
27      ;SEND COMMAND BACK TO UDA
28
29 015356 042765 000010 000014      RSPOUT: BIC #CT.MSG,C.FLG(R5)      ;CLEAR MESSAGE RECEIVED FLAG
30 015364 032765 000020 000014      BIT #CT.REQ,C.FLG(R5)      ;CHECK WHICH COMMAND TO SEND
31 015372 001014      BNE RSPOU2      ;BRANCH IF RESPONSE TO REQUEST
32
33 015374 012700 000005      MOV #OP.RSD,R0      ;BUILD RECEIVE DATA COMMAND
34 015400 004737 023150      CALL BLDCMD
35 015404 012700 000272      MOV #HC.BF2,R0      ;POINT TO MESSAGE BUFFER
36 015410 004737 023332      CALL CLRBUF      ; AND CLEAR IT
37 015414 052765 000020 000014      BIS #CT.REQ,C.FLG(R5)      ;SET REQUEST BIT
38 015422 000403      BR RSPOU3
39
40 015424 042765 000020 000014      RSPOU2: BIC #CT.REQ,C.FLG(R5)      ;CLEAR REQUEST BIT
41 015432      RSPOU3:
42 015432 004737 023234      CALL SNDCMD      ;SEND COMMAND TO UDA
43 015436 012700 000264      MOV #3.*60.,R0      ;SET TIMEOUT FOR 3 MINUTES
44 015442 010501      MOV R5,R1
45 015444 062701 000040      ADD #C.TO,R1      ;PUT TIME IN CONTROLLER TABLE
46 015450 004737 023604      CALL SETTO
47 015454 000137 015070      JMP RSPNXT      ;NOW WAIT FOR END PACKET
    
```

```
1  
2  
3 015460 015516  
4 015462 015636  
5 015464 016002  
6 015466 016452  
7 015470 016474  
8 015472 016754  
9 015474 017004  
10 015476 017034  
11 015500 017062  
12 015502 017102  
13 015504 017244  
14 015506 017350  
15 015510 017564  
16 015512 017704  
17 015514 020016  
18  
19 000017
```

;RESPONSE REQUEST DISPATCH TABLE

```
RSPDSP: .WORD T1MSIZ  
          .WORD T2DLL  
          .WORD T2CMD  
          .WORD T4MPRM  
          .WORD T4UPRM  
          .WORD T4BB1  
          .WORD T4BB2  
          .WORD T4SOFT  
          .WORD T4SEEK  
          .WORD T4MXFR  
          .WORD UTOTST  
          .WORD ERRMES  
          .WORD ERRMC  
          .WORD MESSAG  
          .WORD DONE
```

DSPSIZ=<.-RSPDSP>/2

```
: 0. SET UP FREE MEMORY FOR ADDRESS TESTING  
: 1. PROVIDE DIAGNOSTIC PROGRAM FOR DISK DRIVE  
: 2. GET MANUAL INTERVENTION COMMAND  
: 3. TELL DATA PATTERN 16.  
: 4. TELL UNIT PARAMETERS, CLEAR CONTENTS  
: 5. TELL BAD BLOCKS (FIRST 14)  
: 6. TELL BAD BLOCKS (LAST TWO)  
: 7. ADD TO SOFT ERROR AND ECC COUNTS  
: 8. ADD 1000 TO SEEK COUNT  
: 9. ADD TO MEGABITS READ AND WRITE COUNTS  
:10. TELL WHICH DRIVES TO TEST  
:11. REPORT ERROR MESSAGE  
:12. REPORT ERROR MESSAGE AND COUNT HARD ERROR  
:13. PRINT A DESCRIPTIVE MESSAGE  
:14. MARK DM PROGRAM AS NO LONGER RUNNING
```

;LEGAL NUMBERS ARE LOWER THAN THIS

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38

:NORMAL MAINTENANCE READ BUFFER DESCRIPTION

:BYTE OFFSET FROM
:START OF BUFFER

0	REQUEST NUMBER
2	DATA ARGUMENT #1
4	DATA ARGUMENT #2
6	DATA ARGUMENT #3
8	DATA ARGUMENT #4
10	DATA ARGUMENT #5
12	DATA ARGUMENT #6
14	DATA ARGUMENT #7
16	DATA ARGUMENT #8
18	DATA ARGUMENT #9
20	DATA ARGUMENT #10
22	DATA ARGUMENT #11
.	.
.	.
.	.
68	DATA ARGUMENT #34

USED TO SELECT ROUTINE
R4 CONTAINS THIS ADDRESS

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38

:NORMAL PSEUDO-TERMINAL IN PACKET DESCRIPTION GIVEN IN RESPONSE TO ABOVE PACKET

:BYTE OFFSET FROM
:START OF PACKET

0	REQUEST NUMBER
2	DATA ARGUMENT #1
4	DATA ARGUMENT #2
6	DATA ARGUMENT #3
8	DATA ARGUMENT #4
10	DATA ARGUMENT #5
12	DATA ARGUMENT #6
14	DATA ARGUMENT #7
16	DATA ARGUMENT #8
18	DATA ARGUMENT #9
20	DATA ARGUMENT #10
22	DATA ARGUMENT #11
.	.
.	.
.	.
68	DATA ARGUMENT #34

ECHOED FROM REQUEST PACKET

R3 CONTAINS THIS ADDRESS

ALL DATA ARGUMENTS ARE RETURNED
CONTAINING ZEROS UNLESS
SPECIFICALLY INDICATED BY
RESPONSE ROUTINE.

```

1      :T1MSIZ - DM REQUEST 0
2      :
3      :SET UP MEMORY FOR ADDRESS TESTING FROM UDA.
4      :PLACE ADDRESS OF EACH LOCATION INTO EACH LOCATION IN FREE
5      :MEMORY. RETURN FIRST LOCATION OF FREE MEMORY IN CMD.02 (LOW BITS)
6      :AND CMD.03 (HIGH BITS). RETURN LAST LOCATION OF FREE MEMORY IN
7      :CMD.04 AND CMD.05. ALSO RETURN FIRST EXISTANT LOCATION IN CMD.06
8      :AND CMD.07; LAST EXISTANT LOCATION IN CMD.08 AND CMD.09.
9      :
10     :INPUTS:
11     :       R5 - CONTROLLER TABLE ADDRESS
12     :       R4 - MESSAGE PACKET DATA ADDRESS (POINTING TO MSG.02)
13     :       R3 - COMMAND PACKET DATA ADDRESS (POINTING TO CMD.02)
14     :OUTPUTS:
15     :       COMMAND PACKET CONTAINING:
16     :       (R3) LOW ADDRESS BITS OF FIRST WRITABLE ADDRESS
17     :       2.(R3) HIGH ADDRESS BITS OF FIRST WRITABLE ADDRESS
18     :       4.(R3) LOW ADDRESS BITS OF LAST WRITABLE ADDRESS
19     :       6.(R3) HIGH ADDRESS BITS OF LAST WRITABLE ADDRESS
20     :       8.(R3) LOW ADDRESS BITS OF FIRST READABLE ADDRESS
21     :       10.(R3) HIGH ADDRESS BITS OF FIRST READABLE ADDRESS
22     :       12.(R3) LOW ADDRESS BITS OF LAST READABLE ADDRESS
23     :       14.(R3) HIGH ADDRESS BITS OF LAST READABLE ADDRESS
24     :       Z SET
25     :
26     015516 013701 002176      T1MSIZ: MOV FFREE,R1          :GET FIRST ADDRESS OF FREE MEMORY
27     015522 013702 002200      MOV FSIZE,R2              :GET SIZE
28     :
29     :FILL MEMORY WITH ADDRESS PATTERN
30     :
31     015526 010111              MEMFIL: MOV R1,(R1)        :WRITE DATA INTO LOCATION
32     015530 062701 000002      ADD #2,R1                 :INCREASE ADDRESS TO NEXT LOCATION
33     015534 005302              DEC R2                     :COUNT THE WORDS
34     015536 001373              BNE MEMFIL                 :FILL ALL WORDS
35     :
36     :SEND LOCATION OF FREE MEMORY TO UDA
37     :
38     015540 013723 002176      MOV FFREE,(R3)+          :LOAD FIRST ADDRESS OF FREE MEMORY
39     015544 005023              CLR (R3)+                 : HIGH ORDER BITS ARE ZERO
40     015546 013700 002200      MOV FSIZE,R0             :GET SIZE OF FREE MEMORY
41     015552 006300              ASL R0                    :CONVERT TO BYTES
42     015554 063700 002176      ADD FFREE,R0             :COMPUTE LAST LOCATION
43     015560 162700 000002      SUB #2,R0
44     015564 010023              MOV R0,(R3)+            :LOAD LAST LOCATION
45     015566 005023              CLR (R3)+                :CLEAR HIGH ORDER BITS
    
```



```
1  
2          :SEND LOCATION OF READABLE MEMORY  
3 015570 005023          CLR (R3)+          ;SEND ZERO AS START OF READABLE MEMORY  
4 015572 005023          CLR (R3)+  
5 015574 013700 002120  MOV L$HIMEM,R0      ;GET HIGH MEMORY ADDRESS  
6 015600 005001          CLR R1          ;CLEAR HIGH BITS  
7 015602 006300          ASL R0          ;SHIFT LEFT 6 PLACES  
8 015604 006300          ASL R0  
9 015606 006300          ASL R0  
10 015610 006300         ASL R0  
11 015612 006300         ASL R0  
12 015614 006101         ROL R1  
13 015616 006300         ASL R0  
14 015620 006101         ROL R1  
15 015622 052700 000076  BIS #76,R0          ;SET LOW ORDER BITS  
16 015626 010023         MOV R0,(R3)+      ;PUT INTO BUFFER  
17 015630 010123         MOV R1,(R3)+  
18 015632 000264         SEZ  
19 015634 000207         RETURN
```

```

1      ;T2DLL - DM REQUEST 1
2
3      ;PROVIDE DIAGNOSTIC TO DOWNLINE LOAD INTO DISK DRIVE.
4
5      ;THE UDA MAY BE USED TO GET THE DIAGNOSTIC IF THE SYSTEM LOAD DEVICE
6      ;IS ON THE UDA. THIS ACTION WILL CAUSE A REINITIALIZATION OF THE UDA
7      ;AND THE RING STRUCTURE MOVED. SINCE THIS PROGRAM HAS NO WAY TO
8      ;DETERMINE IF THE UDA IS USED, IT WILL ALWAYS ASSUME IT IS USED AND
9      ;WILL INITIALIZE AND RELOAD THE DM PROGRAM AFTER READING THE
10     ;DIAGNOSTIC. THE OUTPUTS OF THIS ROUTINE ARE STORED AND SENT TO THE
11     ;DM PROGRAM IN THE UTOTST REQUEST.
12
13     ;INPUTS:
14     R5 - CONTROLLER TABLE ADDRESS
15     R4 - MESSAGE DATA ADDRESS
16         (R4) DRIVE NUMBER
17         2.(R4) A VALUE THE DM PROGRAM WISHES RETURNED
18         4.(R4) REGION TO WHICH PROGRAM IS TO BE LOADED IN DISK
19         6.(R4) 2 WORD PROGRAM NAME IN RAD50
20     R3 - COMMAND DATA ADDRESS
21
22     ;OUTPUTS:
23     COMMAND PACKET COULD CONTAIN THE FOLLOWING:
24         (R3) ONE IF PROGRAM PROVIDED, TWO IF PROGRAM NOT AVAILABLE
25         2.(R3) DRIVE NUMBER
26         4.(R3) COPY OF THE VALUE FROM DM PROGRAM
27         6.(R3) REGION TO WHICH PROGRAM IS TO BE LOADED
28         8.(R3) ADDRESS OF FIRST BYTE TO BE DOWNLINE LOADED
29         10.(R3) HIGH ORDER BITS OF ADDRESS
30         12.(R3) BYTE COUNT OF PROGRAM TO BE DOWNLINE LOADED
31         Z SET
32     ;THIS PROGRAM WILL NOT SEND A COMMAND PACKET IN RESPONSE TO THIS REQUEST.
33     ;THE UDA WILL BE REINITIALIZED AND THE DM PROGRAM RELOADED. THEN THIS DATA
34     ;WILL BE APPENDED TO THE NEXT UTOTST REQUEST.
35
36     ;COPY REQUEST DATA TO STORAGE
37     T2DLL: CLR DLL ;CLEAR CONTROL WORD
38             MOV (R4)+,DLLDR ;DRIVE NUMBER
39             MOV (R4)+,DLLV ;VALUE FROM DM
40             MOV (R4)+,DLLR ;REGION
41             MOV (R4)+,DLLNAM ;PROGRAM NAME
42             MOV (R4)+,DLLNAM+2 ; (TWO WORDS)
    
```

```
1  
2  
3  
4 015666 005075 000000  
5 015672 013737 002176 002424  
6 015700 005037 002426  
7 015704 013737 002200 002430  
8 015712 004737 024626  
9 015716 103002  
10 015720 005237 002414  
11 015724 005237 002414 1$:  
12 015730 013737 002430 002200  
13 015736 013737 002176 002430  
14 015744 163737 002424 002430  
15 015752 013737 002424 002176  
16 015760 005726  
17 015762 012701 000001  
18 015766 004737 014632  
19 015772 001402  
20 015774 000137 014726  
21 016000 000207 2$:
```

 :RESET UDA AND READ DM PROGRAM

 CLR @R5
 MOV FFREE,DLLADR
 CLR DLLADR+2
 MOV FSIZE,DLLSIZ
 CALL RDDLL
 BCC 1\$
 INC DLL
 INC DLL
 MOV DLLSIZ,FSIZE
 MOV FFREE,DLLSIZ
 SUB DLLADR,DLLSIZ
 MOV DLLADR,FFREE
 TST (SP)+
 MOV #1,R1
 CALL RUNDM
 BEQ 2\$
 JMP RESPDM
 RETURN

 :RESET THE UDA
 :GET ADDRESS WHERE PROGRAM
 : TO BE STORED
 :SAVE CURRENT SIZE OF MEMORY
 :READ DLL PROGRAM FROM DATA FILE
 :PROGRAM NOT FOUND IF CARRY SET
 :RETURN 1 IF PROGRAM FOUND
 :RETURN 2 IF PROGRAM NOT FOUND
 :COMPUTE SIZE OF DLL PROGRAM
 : AND RESTORE ORIGINAL FFREE
 : AND FSIZE VALUES

 :POP RETURN ADDRESS OFF STACK
 :RUN THE DM PROGRAM AGAIN

1	016054			1\$:	PNTF T2CMS1,D.UNIT(R4),(R5),(R4)		;PRINT DESCRIPTION
	016054	011446					MOV (R4),-(SP)
	016056	011546					MOV (R5),-(SP)
	016060	016446	000002				MOV D.UNIT(R4),-(SP)
	016064	004137	022266				JSR R1,LPNTF
	016070	005445					.WORD T2CMS1
	016072	000006					.WORD PNT.CT
2	016074	005037	002400		CLR T2WRR		
3	016100	005037	002402		CLR T2WRO		;CLEAR ALL STORAGE WORDS
4	016104	005037	002404		CLR T2DR		

GLOBAL SUBROUTINES SECTION

1	016110	022702	000002	T2CMD2: CMP #2,R2			
2	016114	001027		BNE T2CMDQ			:SEE IF LAST OPERATION WAS READ
3	016116			PRINT <#' >			:BRANCH IF NOT TO QUESTION
	016116	112700	000040				:PRINT ONE SPACE
	016122	004737	022114			MOVB #' ,R0	
4	016126	013701	002400	MOV T2WRR,R1		CALL CPNT	
5	016132	004737	021654	CALL T2PNTW			:PRINT REGION
6	016136	013701	002402	MOV T2WRO,R1			:PRINT OFFSET
7	016142	004737	021654	CALL T2PNTW			:PRINT A SLASH
8	016146			PRINT #' /			
	016146	112700	000057			MOVB #' / ,R0	
	016152	004737	022114			CALL CPNT	
9	016156	012401		MOV (R4)+,R1			:PRINT THE DATA
10	016160	004737	021704	CALL T2PNTB			
11	016164			PRINT #CR			:END THE LINE
	016164	112700	000015			MOVB #CR,R0	
	016170	004737	022114			CALL CPNT	


```

1
2
3 016174
   016174 104443
   016176 000406
   016200 002262
   016202 000142
   016204 003213
   016206 177777
   016210 000001
   016212 000024
   016214
4 016214 012701 002262
5 016220 112100
6 016222 022700 000105
7 016226 001415
8 016230 022700 000104
9 016234 001016
10 016236 012713 000003
11 016242 004737 021766
12 016246 001402
13 016250 010437 002404
14 016254 013763 002404
15 016262 004737 021766
16 016266 001064
17 016270 000461

;NOW ASK FOR COMMAND INPUT
T2CMDQ: GMANID T4OPT7,TEMP,A,-1,1,20.,NO ;ASK FOR COMMAND
TRAP CS$GMAN
BR 10000$
.WORD TEMP
.WORD T$CODE
.WORD T4OPT7
.WORD -1
.WORD T$LOLIM
.WORD T$HILIM

10000$:
;GET POINTER TO STRING
;GET COMMAND CHARACTER

;STORE DIAGNOSE OPERATION CODE
;GET REGION FROM COMMAND

000002 1$:
T2CMDV: CALL T2GNUM
        BNE T2CMDE
        BR T2CMDX

;MAKE SURE AT END OF LINE
    
```

```

1          ;COMMAND MUST BE EITHER READ OR WRITE
2
3 016272 012713 000002      T2CMD3: MOV #2,(R3)          ;CHECK IF READ
4 016276 022700 000122      CMP #'R,R0
5 016302 001415              BEQ T2CMDR
6 016304 022700 000127      CMP #'W,R0          ;CHECK IF WRITE
7 016310 001053              BNE T2CMDE          ; IF NOT - ERROR
8 016312 012713 000001      MOV #1,(R3)
9 016316 004737 021766      CALL T2GNUM        ;GET DATA BYTE
10 016322 001446             BEQ T2CMDE          ;ERROR IF NO DATA
11 016324 162700 000002      SUB #2,R0
12 016330 003043             BGT T2CMDE          ;OR GREATER THAN TWO DIGITS
13 016332 010463 000006      MOV R4,6(R3)        ;STORE DATA BYTES IN BUFFER
14 016336 013763 002400 000002 T2CMDR: MOV T2WRR,2(R3)    ;PUT REGION AND OFFSET
15 016344 013763 002402 000004      MOV T2WRO,4(R3)    ; INTO BUFFER
16 016352 021302             CMP (R3),R2          ;IF SO,
17 016354 001002             BNE T2CMDN          ; INCREMENT OFFSET
18 016356 005263 000004      INC 4(R3)
19 016362 004737 021766      T2CMDN: CALL T2GNUM
20 016366 001411             BEQ T2CMDW
21 016370 010463 000002      MOV R4,2(R3)
22 016374 005063 000004      CLR 4(R3)
23 016400 004737 021766      CALL T2GNUM
24 016404 001402             BEQ T2CMDW
25 016406 010463 000004      MOV R4,4(R3)
26 016412 004737 021766      T2CMDW: CALL T2GNUM
27 016416 001010             BNE T2CMDE
28 016420 016337 000002 002400      MOV 2(R3),T2WRR    ;SAVE REGION
29 016426 016337 000004 002402      MOV 4(R3),T2WRO    ;SAVE OFFSET
30 016434 000264             T2CMDX: SEZ
31 016436 000207             RETURN
32 016440             T2CMDE: PNTF T2CMS5      ;REPORT ERROR MESSAGE
    016440 004137 022266      JSR R1,LPNTF
    016444 006034              .WORD T2CMS5
    016446 000000              .WORD PNT.CT
33 016450 000651             BR T2CMDQ          ;GO ASK AGAIN
    
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

:T4MPRM - DM REQUEST 3
:REQUEST FOR TEST 4 CONTENTS OF DATA PATTERN 16.
:INPUTS:
:R5 - CONTROLLER TABLE ADDRESS
:R4 - MESSAGE DATA ADDRESS
: (NO DATA)
:R3 - COMMAND DATA ADDRESS
:OUTPUTS:
:COMMAND DATA FILLED WITH THE FOLLOWING:
: (R3) NUMBER OF WORDS IN DATA PATTERN 16
: 2.(R3) DATA IN PATTERN 16
: |
: 32.(R3) ..
: Z SET

18 016452 012701 000021
19 016456 012702 002312
20 016462 012223
21 016464 005301
22 016466 001375
23 016470 000264
24 016472 000207

T4MPRM: MOV #17,R1 ;GET COUNT
MOV #PAT16C,R2 ; AND ADDRESS OF PATTERN 16 PARAMETERS
1\$: MOV (R2)+,(R3)+ ;COPY THE DATA TO BUFFER
DEC R1
BNE 1\$
SEZ ;RETURN WITH Z SET
RETURN

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

```
:T4UPRM - DM REQUEST 4  
:REQUEST FOR TEST 4 UNIT PARAMETERS  
:INPUTS:  
R5 - CONTROLLER TABLE ADDRESS  
R4 - MESSAGE DATA ADDRESS  
  (R4) DRIVE NUMBER  
  2.(R4) DRIVE SERIAL NUMBER  
  :  
  6.(R4)  
  8.(R4) HDA SERIAL NUMBER  
  :  
 14.(R4)  
R3 - COMMAND DATA ADDRESS  
:OUTPUTS:  
COMMAND DATA FILLED WITH THE FOLLOWING:  
(R3) PARAMETER BITS (1 FOR TRUE)  
  BIT 14 - INITIAL WRITE  
  BIT 13 - DIAGNOSTIC CYLINDERS  
  BIT 12 - ECC CORRECTION  
  BIT 11 - READ ONLY  
  BIT 10 - WRITE ONLY  
  BIT 9 - RETRIES  
  BIT 8 - TRACK/GROUP AND CYLINDERS SPECIFIED  
  BIT 7 - (NOT USED)  
  BIT 6 - SEQUENTIAL SEEKS  
  BIT 5 - BEGIN/END SETS SPECIFIED  
  BIT 4 - TRACK SPECIFIED (0 - GROUPS SPECIFIED)  
           HAS MEANING ONLY WHEN BIT 5 IS ZERO  
  BIT 3 - WRITE CHECKS ENABLED  
  BIT 2 - WRITE CHECKS ALWAYS  
  BIT 1 - DATA COMPARES ENABLED  
  BIT 0 - DATA COMPARE ALWAYS  
2.(R3) DATA PATTERN NUMBER  
IF PARAMETER BIT 5 SET  
4.(R3) COUNT OF BEGIN/END SETS  
6.(R3) BEGIN BLOCK (2 WORDS) THEN END BLOCK (2 WORDS)  
  1 TO 4 SETS  
  OR  
  IF COUNT OF BEGIN/END BLOCKS = 0  
36.(R3) START CYLINDER (2 WORDS) THEN END CYLINDER (2 WORDS)  
  END CYLINDER A NEGATIVE VALUE IF TO TEST ENTIRE AREA  
IF PARAMETER BIT 5 CLEAR  
4.(R3) STARTING CYLINDER  
6.(R3) (2 WORDS)  
8.(R3) ENDING CYLINDER (2 WORDS)  
10.(R3) NEGATIVE FOR ALL CYLINDERS  
12.(R3) NUMBER OF TRACKS OR GROUPS SPECIFIED  
14.(R3) 1 TO 7 TRACK OR GROUP NUMBERS  
  DETERMINED BY PARAMETER BIT 4  
26.(R3)  
Z SET IF DATA RETURNED  
Z CLEAR IF UNIT NUMBER NOT ON THIS CONTROLLER
```

1	016474	012401		T4UPRM:	MOV (R4)+,R1		:GET DRIVE NUMBER
2	016476	010402			MOV R4,R2		:SAVE DATA ADDRESS
3	016500	004737	020734		CALL GIDRVT		:GET DRIVE TABLE ADDRESS
4	016504	001122			BNE T4UPRX		:CHECK IF DRIVE FOUND
5	016506	012264	000200		MOV (R2)+,D.SERN(R4)		:COPY DRIVE SERIAL NUMBER TO DRIVE TABLE
6	016512	012264	000202		MOV (R2)+,D.SERN+2(R4)		
7	016516	012264	000204		MOV (R2)+,D.SERN+4(R4)		
13	016522	016401	000004		MOV D.PRM(R4),R1		:GET PARAMETER BITS
14	016526	042701	140200		BIC #D.ZERO,R1		:CLEAR SOME BITS
15	016532	032737	000020	002226	BIT #ISTRTH,IFLAGS		:IF FIRST TIME TEST 4 BEING RUN
16	016540	001406			BEQ 2\$: AFTER A START COMMAND
17	016542	032737	040000	002160	BIT #SM.IW,SFPTBL+SO.BIT		:GET INITIAL WRITE BIT
18	016550	001402			BEQ 2\$		
19	016552	052701	040000		BIS #D.IW,R1		:MOVE INTO PARAMETER BITS
20	016556	010123		2\$:	MOV R1,(R3)+		:PUT INTO BUFFER
21	016560	016423	000006		MOV D.PAT(R4),(R3)+		:PUT PATTERN NUMBER IN BUFFER
22	016564	032701	000040		BIT #D.BE,R1		:CHECK BEGIN/END PARAMETER BIT
23	016570	001411			BEQ 10\$:BRANCH IF NOT SET
24							
25					:RETURN BEGIN/END SETS		
26							
27	016572	012701	000021		MOV #4*4+1,R1		:COUNT OF SETS TIMES WORDS PER SET PLUS COUNT WORD
28	016576	010402			MOV R4,R2		:GET INDEX INTO DRIVE TABLE
29	016600	062702	000112		ADD #D.BEC,R2		
30	016604	012223		1\$:	MOV (R2)+,(R3)+		:TRANSFER THE BEGIN/END SETS
31	016606	005301			DEC R1		
32	016610	001375			BNE 1\$		
33	016612	000457			BR T4UPRX		
34							
35	016614	032764	000400	000004	10\$:	BIT #D.CYL,D.PRM(R4)	:LOOK AT D.CYL BIT
36	016622	001441			BEQ 20\$:BRANCH IF NOT SET
37							
38					:RETURN TRACKS/GROUPS AND CYLINDERS		
39							
40	016624	005764	000112		TST D.BEC(R4)		:CHECK IF ANY TRACKS/GROUPS
41	016630	001421			BEQ 25\$:BRANCH IF NONE
42	016632	012701	000004		MOV #4,R1		:COUNT OF CYLINDER WORDS
43	016636	010402			MOV R4,R2		
44	016640	062702	000154		ADD #D.BCYL,R2		
45	016644	012223		11\$:	MOV (R2)+,(R3)+		:CYLINDERS
46	016646	005301			DEC R1		
47	016650	001375			BNE 11\$		
48	016652	012701	000010		MOV #8.,R1		
49	016656	010402			MOV R4,R2		
50	016660	062702	000112		ADD #D.BEC,R2		
51	016664	012223		12\$:	MOV (R2)+,(R3)+		:TRACKS/GROUPS
52	016666	005301			DEC R1		
53	016670	001375			BNE 12\$		
54	016672	000427			BR T4UPRX		

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
;RETURN CYLINDERS ONLY  
016674 052763 000040 177774 25$: BIS #D.BE,-4(R3) ;SET D.BE FOR DM PROGRAM  
016702 005023 CLR (R3)+ ;SEND ZERO BEGIN/END COUNT  
016704 012701 000004 MOV #4,R1  
016710 010402 MOV R4,R2  
016712 062702 000154 ADD #D.BCYL,R2  
016716 012223 26$: MOV (R2)+,(R3)+ ;CYLINDERS  
016720 005301 DEC R1  
016722 001375 BNE 26$  
016724 000412 BR T4UPRX  
;RETURN ENTIRE AREA  
016726 052763 000040 177774 20$: BIS #D.BE,-4(R3) ;SET D.BE FOR DM PROGRAM  
016734 005023 CLR (R3)+ ;BEGIN/END COUNT OF ZERO  
016736 005023 CLR (R3)+ ;START CYLINDER OF ZERO  
016740 005023 CLR (R3)+  
016742 005023 CLR (R3)+ ;END CYLINDER NEGATIVE  
016744 012723 177777 MOV #-1,(R3)+  
016750 000264 SEZ  
016752 000207 T4UPRX: RETURN
```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30

```

:T4BB1 - DM REQUEST 5
:REQUEST FOR FIRST 14 BAD BLOCKS
:INPUTS:
:   R5 - CONTROLLER TABLE ADDRESS
:   R4 - MESSAGE DATA ADDRESS
:         (R4) DRIVE NUMBER
:   R3 - COMMAND DATA ADDRESS
:OUTPUTS:
:   COMMAND DATA FILLED WITH BAD BLOCKS
:   (R3) COUNT OF BAD BLOCKS
:   2.(R3) BAD BLOCK 1 (LOW)
:   4.(R3)                (HIGH)
:
:   56.(R3) BAD BLOCK 14 (LOW)
:   58.(R3)                (HIGH)
:   Z SET IF DATA RETURNED
:   Z CLEAR IF DRIVE NUMBER NOT ON THIS CONTROLLER
    
```

```

22 016754 011401
23 016756 004737 020734
24 016762 001007
25 016764 062704 000010
26 016770 012701 000035
27 016774 012423
28 016776 005301
29 017000 001375
30 017002 000207
    
```

```

T4BB1:  MOV (R4),R1           :GET DRIVE NUMBER
        CALL GTDRVT         :GET DRIVE TABLE ADDRESS
        BNE T4BB1E         :CHECK IF DRIVE FOUND
        ADD #D.BB,R4       :INCREASE ADDRESS TO DATA TO COPY
        MOV #<1+<14.*2>>,R1 :GET COUNT OF WORDS
1$:     MOV (R4)+,(R3)+     :COPY THE WORDS
        DEC R1
        BNE 1$
T4BB1E: RETURN
    
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23

```
:T4BB2 - DM REQUEST 6  
:REQUEST LAST TWO BAD BLOCKS  
:INPUTS:  
:   R5 - CONTROLLER TABLE ADDRESS  
:   R4 - MESSAGE DATA ADDRESS  
:       (R4) DRIVE NUMBER  
:   R3 - COMMAND DATA ADDRESS  
:OUTPUTS:  
:   COMMAND DATA FILLED WITH BAD BLOCKS 15 AND 16  
:   Z SET IF DATA RETURNED  
:   Z CLEAR IF UNIT NUMBER NOT ON THIS CONTROLLER  
  
T4BB2:  MOV (R4),R1           ;GET DRIVE NUMBER  
        CALL GTDRVT         ;GET DRIVE TABLE ADDRESS  
        BNE T4BB2E         ;CHECK IF DRIVE FOUND  
        ADD #D.BB15,R4     ;INCREASE ADDRESS TO DATA TO COPY  
        MOV #4,R1          ;GET COUNT OF WORDS  
1$:     MOV (R4)+,(R3)+    ;COPY THE WORDS  
        DEC R1  
        BNE 1$  
T4BB2E: RETURN
```

```
15 017004 011401  
16 017006 004737 020734  
17 017012 001007  
18 017014 062704 000102  
19 017020 012701 000004  
20 017024 012423  
21 017026 005301  
22 017030 001375  
23 017032 000207
```

```
1      ;T4SOFT - DM REQUEST 7
2
3      ;ADD TO SOFT ERROR AND ECC COUNTS
4
5      ;INPUTS:
6      ;
7      ;   R5 - CONTROLLER TABLE ADDRESS
8      ;   R4 - MESSAGE DATA ADDRESS
9      ;         (R4) DRIVE NUMBER
10     ;   2.(R4) VALUE TO ADD TO SOFT ERROR COUNT
11     ;   4.(R4) VALUE TO ADD TO ECC COUNT
12     ;
13     ;   R3 - COMMAND DATA ADDRESS
14
15
16 017034 012401      T4SOFT: MOV (R4)+,R1      ;GET DRIVE NUMBER
17 017036 010402      MOV R4,R2      ;SAVE DATA ADDRESS
18 017040 004737 020734 CALL GTDRVT      ;GET DRIVE TABLE ADDRESS
19 017044 001005      BNE 1$      ;CHECK IF DRIVE FOUND
20 017046 062264 000172 ADD (R2)+,D.SERR(R4) ;ADD TO SOFT ERROR COUNT
21 017052 062264 000176 ADD (R2)+,D.ECCC(R4) ;ADD TO ECC COUNT
22 017056 000264      SEZ      ;EXIT
23 017060 000207      1$: RETURN
```


1 017062
2
3
4
5
6
7
8
9
10
11
12 017062 011401
13 017064 004737 020734
14 017070 001003
15 017072 005264 000174
16 017076 000264
17 017100 000207

T4SEEK:
: DM REQUEST 8.
: RECORD 1000 SEEKS COMPLETED ON DRIVE
: INPUTS:
: R5 - CONTROLLER TABLE ADDRESS
: R4 - MESSAGE DATA ADDRESS
: (R4) DRIVE NUMBER
: R3 - COMMAND DATA ADDRESS
:
: MOV (R4),R1 ; GET DRIVE NUMBER
: CALL GTDRVT ; GET DRIVE TABLE ADDRESS
: BNE SEKERE ; CHECK IF DRIVE FOUND
: INC D.SEEK(R4) ; COUNT THE BITS TRANSFERRED
: SEZ ; NORMAL RETURN
SEKERE: RETURN

```

1      ;T4MXFR - DM REQUEST 9.
2      ;
3      ;RECORD 1M BITS TRANSFERRED ON UNIT. COMPARE TO TRANSFER LIMIT AND
4      ;REPORT LIMIT REACHED.
5      ;
6      ;INPUTS:
7      ;
8      ;   R5 - CONTROLLER TABLE ADDRESS
9      ;   R4 - MESSAGE DATA ADDRESS
10     ;       (R4) DRIVE NUMBER
11     ;       2.(R4) VALUE TO ADD TO READ COUNT
12     ;       4.(R4) VALUE TO ADD TO WRITE COUNT
13     ;   R3 - COMMAND DATA ADDRESS
14     ;
15     ;OUTPUTS:
16     ;   (R3) BIT 15 SET IF TRANSFER LIMIT REACHED
17     ;   MESSAGE PRINTED IF TRANSFER LIMIT REACHED
18     ;   Z CLEAR IF DRIVE NUMBER NOT ON THIS CONTROLLER
19
20     T4MXFR: MOV R4,R2           ;GET MESSAGE DATA ADDRESS
21             MOV (R4),R1       ;GET DRIVE NUMBER
22             CALL GTDRV       ;GET DRIVE TABLE ADDRESS
23             BNE MXFERE       ;CHECK IF DRIVE FOUND
24             TST D.UNIT(R4)   ;SEE IF UNIT HAS BEEN DROPPED
25             BPL 1$          ;CONTINUE IF STILL TO BE TESTED
26
27             ASSUME DT.AVL EQ BIT15
28             BIS #BIT15,(R3)   ;TELL DM PROGRAM TO STOP TESTING THIS UNIT
29             BR MXFERX        ; AND EXIT WITHOUT ADDING TO ADDING TO COUNTS
30
31     1$:
32     ADD 2(R2),D.XFRR(R4)      ;ADD MEGABITS READ
33     ADD 4(R2),D.XFRW(R4)     ;ADD MEGABITS WRITTEN
34     TST SFPTBL+SO.XL         ;SEE IF LIMIT SPECIFIED
35     BEQ MXFERX               ;BRANCH IF NOT
36     CMP D.XFRR(R4),SFPTBL+SO.XL ;CHECK IF LIMIT REACHED
37     BLO MXFERX               ;BRANCH IF LIMIT NOT REACHED
38     RFLAGS R0                ;CHECK FLAGS
39
40     BIT #IDU,R0               ;SEE IF DROPPING UNITS IS INHIBITED
41     BNE MXFERX
42     BIS #BIT15,(R3)          ;SET DROP UNIT BIT
43     BIC #CT.MSG,C.FLG(R5)    ;CLEAR MESSAGE RECEIVED FLAG
44     PNTX MESSG,D.UNIT(R4),(R5),(R4) ;PRINT TESTING DONE
45
46     MOV (R4),-(SP)
47     MOV (R5),-(SP)
48     MOV D.UNIT(R4),-(SP)
49     JSR R1,LPNTX
50     .WORD MESSG
51     .WORD PNT.CT
52
53     CALL RNTIME              ;PRINT RUNTIME
54     PNTX MXFERP
55
56     JSR R1,LPNTX
57     .WORD MXFERP
58     .WORD PNT.CT
59
60     MXFERX: SEZ                ;NORMAL RETURN
61     MXFERE: RETURN
    
```



```

1      ;UTOTST - DM REQUEST 10
2
3      ;TELL DM PROGRAM WHICH DRIVES ARE SELECTED FOR TESTING
4      ;AND CLEAR STATISTICS IN DRIVE TABLE
5
6      ;INPUTS:
7          R5 - CONTROLLER TABLE ADDRESS
8          R4 - MESSAGE DATA ADDRESS
9              (NO DATA)
10         R3 - COMMAND DATA ADDRESS
11
12     ;OUTPUTS:
13         COMMAND PACKET CONTAINING UP TO 8 DRIVE NUMBERS.
14         LIST IS ENDED BY A WORD WITH BIT 15 SET.
15         FOLLOWING LIST IS THE INFORMATION FROM T2DLL REQUEST IF APPLICABLE.
16         D.XFRW, D.XFRR, D.HERR, D.SERR, D.SEEK AND D.ECC CLEARED IN DRIVE TABLE
17         Z SET
18 017244 010504      UTOTST: MOV R5,R4          ;GET ADDRESS OF CONTROLLER TABLE
19 017246 062704 000020 ADD #C.DR0,R4      ;BUMP TO DRIVE TABLE POINTERS
20 017252 012702 000010 MOV #8,R2          ;GET COUNT OF PORTS
21 017256 012400      UTOT1: MOV (R4)+,R0      ;SEE IF DRIVE TABLE POINTER EXISTS
22 017260 001415      BEQ UTOT2          ;BRANCH IF NOT
23 017262 005760 000002 TST D.UNIT(R0)    ;LOOK IF UNIT AVAILABLE FOR TESTING
24 017266 100410      BMI UTOT1A
25 017270      ASSUME DT.AVL EQ BIT15
26 017270 011023      MOV (R0),(R3)+      ;LOAD DRIVE NUMBER FROM TABLE
27 017272 062700 000164 ADD #D.XFRW,R0    ;CLEAR STATISTICS IN DRIVE TABLE
28 017276 012701 000011 MOV #D.SIZE-D.XFRW/2,R1
29 017302 005020      1$: CLR (R0)+
30 017304 005301      DEC R1
31 017306 001375      BNE 1$
32 017310 005302      UTOT1A: DEC R2          ;COUNT THE DRIVE TABLES
33 017312 001361      BNE UTOT1          ;REPEAT FOR EACH TABLE
34 017314 012723 100000 UTOT2: MOV #BIT15,(R3)+ ;TERMINATE LIST
35 017320 013723 002414 MOV DLL,(R3)+    ;GET DLL CONTROL WORD
36 017324 001407      BEQ UTOT4          ; IF NON-ZERO
37 017326 012701 002416 MOV #DLLDR,R1    ; TRANSFER ALL DLL WORDS INTO BUFFER
38 017332 012702 000020 MOV #<DLLNAM+4-DLLDR>,R2
39 017336 012123      UTOT3: MOV (R1)+,(R3)+
40 017340 005302      DEC R2
41 017342 001375      BNE UTOT3
42 017344 000264      UTOT4: SEZ
43 017346 000207      RETURN          ;RETURN WITH Z SET
    
```



```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25 017350
36 017350 005764 000002
37 017354 100406
38 017356
   017356 104421
39 017360 032700 000040
40 017364 001014
55 017366 052713 100000
56 017372 016400 000002
57 017376 005100
58 017400 032700 140000
59 017404 001004
60 017406 032737 000400 002160
61 017414 001061
62 017416
63 017416 042765 000010 000014
65 017424 022737 000004 002230
66 017432 001004
68 017434 032737 001000 002160
69 017442 001005
70 017444 004737 022372
71 017450 103043
75 017452 000244
76 017454 000207

:ERRMES - DM REQUEST 11
:PRINT AN ERROR MESSAGE
:INPUTS:
R5 - CONTROLLER TABLE ADDRESS
R4 - MESSAGE DATA ADDRESS
   (R4) ERROR PC IN DM PROGRAM
   2.(R4) <15:14> ERROR TYPE
   <13:0 > ERROR NUMBER
   4.(R4) DRIVE NUMBER (-1 IF NOT GIVEN)
   6.(R4) MESSAGE POINTER
   8.(R4) OPTIONAL PARAMETERS FOR ERROR PRINT ROUTINE
   10.(R4) ..
   ..
   58.(R4) ..
R3 - COMMAND DATA ADDRESS
:OUTPUTS:
COMMAND PACKET CONTAINING THE FOLLOWING:
   (R3) - BIT 15 SET IF FATAL ERROR TO INDICATE DRIVE SHOULD NO LONGER BE TESTED
Z SET TO INDICATE DATA RETURNED
Z CLEAR IF DRIVE NUMBER NOT ON THIS CONTROLLER

ERRMES:
TST 2(R4)           :CHECK IF FATAL ERROR
BMI 5$              :BRANCH IF NOT
RFLAGS R0          :LOOK AT FLAGS
                                TRAP   CSRFLA
BIT #IDU,R0        :SEE IF ALLOWED TO DROP UNITS
BNE 6$              :BRANCH IF NOT
BIS #BIT15,(R3)    :SET DROP DRIVE BIT
5$: MOV 2(R4),R0    :SEE IF SOFT ERROR
COM R0
BIT #14000,R0
BNE 6$              :BRANCH IF NOT
BIT #SM.SSF,SO.BIT+SFPTBL
BNE ERRMSX         :SEE IF SOFT ERRORS SUPPRESSED
                                :DON'T PRINT IF SO
6$: BIC #CT.MSG,C.FLG(R5) :CLEAR MESSAGE RECEIVED FLAG
CMP #4,TNUM        :IF TEST # 4,
BNE 7$
BIT #SM.LOG,SFPTBL+SO.BIT
BNE ERRMSL         :SEE IF LOG BEING USED
7$: CALL PNTERR     :IF NOT, PRINT THE ERROR MESSAGE
BCC ERRMSX         :IF DRIVE HASN'T BEEN DROPPED, PRINT
CLZ                :ELSE RETURN
RETURN
    
```

1	017456	005737	002406		ERRMSL: TST LBUFS		
2	017462	001014			BNE 1\$:SEE IF LOG BUFFER ESTABLISHED
3	017464	013701	002176		MOV FFREE,R1		: LBUFS CONTAINS ADDRESS IF ESTABLISHED
4	017470	010137	002406		MOV R1,LBUFS		:GET START ADDRESS OF BUFFER
5	017474	010137	002410		MOV R1,LBUFN		:INITIALIZE BUFFER STORAGE
6	017500	063701	002200		ADD FSIZE,R1		:SAVE ADDRESS WHERE TO ADD
7	017504	063701	002200		ADD FSIZE,R1		:COMPUTE END OF STORAGE AREA
8	017510	010137	002412		MOV R1,LBUFE		
9	017514	013701	002410		1\$: MOV LBUFN,R1		:GET ADDRESS OF DATA STORAGE AREA
10	017520	062737	000106	002410	ADD #HC.BSZ,LBUFN		:ADD BYTES OF STORAGE NEEDED
11	017526	023737	002410	002412	CMP LBUFN,LBUFE		:SEE IF ENOUGH ROOM
12	017534	103007			BHIS 3\$: BRANCH IF NOT
13	017536	010521			MOV R5,(R1)+		:STORE CONTROLLER TABLE ADDRESS
14	017540	012700	000042		MOV #<HC.BSZ-2>/2,R0		:GET COUNT OF REST OF DATA IN WORDS
15	017544	012421			2\$: MOV (R4)+,(R1)+		:STORE DATA
16	017546	005300			DEC R0		
17	017550	001375			BNE 2\$		
18	017552	000402				BR ERRMSX	
19	017554	010137	002410		3\$: MOV R1,LBUFN		:RESTORE OLD VALUE OF LBUFN
20	017560	000264			ERRMSX: SEZ		
21	017562	000207			RETURN		

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29 017564
30 017566 010446 017350
31 017572 012604
32 017574 005713
33 017576 100436
34 017600 016401 000004
35 017604 016402 000002
36 017610 004737 020734
37 017614 001031
38 017616 042702 037777
39 017622 022702 100000
40 017626 001022
41 017630 005264 000170
42 017634 026437 000170 002154
43 017642 103414
44 017644 104421
45 017646 032700 000040
46 017652 001010

```

:ERRMC - DM REQUEST 12.
:REPORT AN ERROR MESSAGE IDENTICAL TO DM REQUEST ERRMES
:THEN ADD ONE TO THE ERROR COUNT FOR THE DRIVE AND SEE IF
:ERROR LIMIT REACHED.
:INPUTS:
R5 - CONTROLLER TABLE ADDRESS
R4 - MESSAGE DATA ADDRESS
(R4) ERROR PC IN DM PROGRAM
2.(R4) < 9:8 > ERROR TYPE
      < 7:0 > ERROR NUMBER
4.(R4) DRIVE NUMBER (-1 IF NOT GIVEN)
6.(R4) <15:12> TYPE
      <11:0 > MESSAGE POINTER
8.(R4) OPTIONAL PARAMETERS FOR ERROR PRINT ROUTINE
10.(R4) ..
      ..
      ..
58.(R4) ..
R3 - COMMAND DATA ADDRESS
:OUTPUTS:
COMMAND PACKET CONTAINING THE FOLLOWING:
(R3) BIT 15 SET IF ERROR COUNT REACHED
      TO INDICATE DRIVE SHOULD NO LONGER BE TESTED.
Z CLEAR IF DRIVE NUMBER NOT ON THIS CONTROLLER
Z SET TO INDICATE DATA RETURNED

ERRMC: PUSH R4
      CALL ERRMES ; CALL REQUEST ERRMES
      POP R4
      MOV R4,-(SP)
      TST (R3) ;SEE IF UNIT ALREADY TO BE DROPPED
      BMI 3$ ; IF SO, JUST EXIT NOW
      MOV 4(R4),R1 ; GET DRIVE NUMBER
      MOV 2(R4),R2 ; GET ERROR TYPE
      CALL GTDRVT ; GET DRIVE TABLE
      BNE 5$ ; EXIT IF NO TABLE FOR UNIT
      BIC #^C140000,R2
      CMP #100000,R2 ;CHECK IF HARD ERROR
      BNE 3$ ;BRANCH IF NOT
      INC D.HERR(R4) ; COUNT THE ERROR
      CMP D.HERR(R4),SFPTBL+SO.EL ; CHECK IF AT LIMIT
      BLO 3$ ; IF LIMIT REACHED, BRANCH
      RFLAGS R0 ;LOOK AT THE FLAGS
      BIT #IDU,R0 TRAP CSRFLA
      BNE 3$ ;SEE IF DROPPING UNITS INHIBITED
      ;BRANCH IF SO
    
```


8	017654			PNTX ERR LIM,D.UNIT(R4)		: PRINT LIMIT REACHED	
	017654	016446	000002				MOV D.UNIT(R4),-(SP)
	017660	004137	022306				JSR R1,LPNTX
	017664	004046					.WORD ERR LIM
	017666	000002					.WORD PNT.CT
9	017670	052713	100000				
13	017674	000264		3\$:	BIS #BIT15,(R3)		:SET STOP TESTING BIT
14	017676	000207			SEZ		: SET Z FOR NORMAL RETURN
15					RETURN		: RETURN TO CALLING PROGRAM
16	017700	000244		5\$:	CLZ		: FLAG AS ERROR
17	017702	000207			RETURN		: RETURN TO CALLING PROGRAM

```

1      :MESSAG - DM REQUEST 13.
2
3      :PRINT A MESSAGE WITH HEADER AS FOLLOWS:
4      :   'UNIT XX UDA AT XXXXXX DRIVE XXX RUNTIME HH:MM:SS '
5      :ENTIRE MESSAGE IS PRINTED WITH PRINTX CALLS.
6
7      :INPUTS:
8      :   R5 - CONTROLLER TABLE ADDRESS
9      :   R4 - MESSAGE DATA ADDRESS
10     :   (R4) DRIVE NUMBER
11     :   2.(R4) MESSAGE POINTER
12     :   2.(R4) MESSAGE POINTER
13     :   4.(R4) OPTIONAL MESSAGE PARAMETERS
14
15     :
16     :
17     :   58.(R4) COMMAND DATA ADDRESS
18 017704 042765 000010 000014 MESSAG: BIC #CT.MSG,C.FLG(R5)      ;CLEAR MESSAGE RECEIVED FLAG
19 017712 012401                MOV (R4)+,R1              ;GET DRIVE NUMBER
20 017714                PUSH R4                ;SAVE DATA POINTER
21 017714 010446                ;
22 017716 004737 020734        CALL GTDRV              ;GET DRIVE TABLE ADDRESS
23 017722 001033                BNE 1$                ;CHECK IF DRIVE FOUND
24 017724 005764 000002        TST D.UNIT(R4)         ;IF UNIT DROPPED FROM TESTING
25 017730 100430                BMI 1$                ;DON'T PRINT ANYTHING
26 017732                PNTX MESSG,D.UNIT(R4),(R5),(R4) ;PRINT HEADER
27 017732 011446                ;
28 017734 011546                ;
29 017736 016446 000002        MOV (R4),-(SP)
30 017742 004137 022306        MOV (R5),-(SP)
31 017746 005302                MOV D.UNIT(R4),-(SP)
32 017750 000006                JSR R1,LPNTX
33 017752 004737 025356        .WORD MESSG
34 017756 012604                .WORD PNT.CT
35 017760 012402                CALL RNTIME           ;GET RUNTIME PARAMETERS
36 017762 006302                POP R4
37 017764 063702 002214        MOV (R4)+,R2         ;GET MESSAGE POINTER
38 017770 067702 162220        ASL R2              ;DOUBLE TO MAKE BYTE OFFSET
39 017774 105712                ADD DMPROG,R2       ;ADD TO START OF MESSAGE STRINGS
40 017776 001001                ADD @DMPROG,R2     ;ADD SIZE OF MAIN PROGRAM
41 020000 005202                TSTB (R2)          ;CHECK FIRST BYTE
42 020002 004737 020022        BNE 2$              ;IF ZERO
43 020006 000264                INC R2              ;INCREMENT TO NEXT BYTE
44 020010 000207                CALL OSTRNG         ;OUTPUT ACCORDING TO STRING
45 020012 012604                SEZ
46 020014 000207                RETURN
47
48     2$:
49     1$:
50     RETURN
51
52     MOV (SP)+,R4
    
```

1
2
3
4
5
6
7
8
9
10
11
12
13 020016 000244
14 020020 000207

:DONE - DM REQUEST 14
:MARK DM PROGRAM AS NO LONGER RUNNING
:INPUTS:
:R5 - CONTROLLER TABLE ADDRESS
:R4 - MESSAGE DATA ADDRESS
: (NO DATA)
:R3 - COMMAND DATA ADDRESS
:OUTPUTS:
:Z CLEAR TO DROP UNIT FROM TESTING
:DONE: CLZ ;DROP UNIT FROM TESTING
:RETURN

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46

```

:OSTRNG
:OUTPUT A MESSAGE ACCORDING TO A FORMAT STRING
:FORMAT OF THE ASCIZ STRING IS AS FOLLOWS:
:CHARACTERS ENCLOSED IN QUOTES ARE TO BE PRINTED AS THEY ARE.
:OTHERWISE CODE IS A SINGLE LETTER FOLLOWED BY AN OPTIONAL DECIMAL
:NUMBER:
:  ON - PRINT OCTAL NUMBER. N REPRESENTS SIZE OF BINARY NUMBER PASSED
:       IN PARAMETER IN BITS. MAY BE IN RANGE 1 TO 32. IF N>16, TWO PARAMETER
:       WORDS ARE USED, OTHERWISE ONLY ONE WORD. LEADING ZEROS ARE PRINTED.
:       N IS ALWAYS SPECIFIED.
:  DN - PRINT UNSIGNED DECIMAL NUMBER FROM N BIT PARAMETER. LEADING ZEROS
:       ARE NOT PRINTED. A 16 BIT NUMBER EQUAL TO ZERO WILL PRINT '0'.
:  HN - PRINT HEX NUMBER FROM PARAMETER OF N BITS. IF N>16 TWO PARAMETERS
:       ARE USED, OTHERWISE ONLY ONE PARAMETER. LEADING ZEROS ARE PRINTED.
:  SN - PRINT N SPACES. N ASSUMED TO BE 1.
:  NN - START NEW LINE (CR-LF SEQUENCE). N ASSUMED TO BE 1.
:  AN - PRINT N ASCII CHARACTERS FROM PARAMETERS, N ASSUMED TO BE 1.
:       N/2 PARAMETER WORDS USED.
:  RN - EXECUTE ROUTINE #N. N MUST BE GIVEN AND DEFINED IN HOST PROGRAM.
:A NULL CHARACTER MEANS END OF MESSAGE. A NULL AS FIRST CHARACTER IN STRING
:MUST BE IGNORED.

```

```

:INPUTS:
:  R2 - ADDRESS OF START OF FORMAT STRING
:  R4 - ADDRESS OF PARAMETERS
:OUTPUTS:
:  R2 AND R4 UPDATED TO END OF STRING AND PARAMETERS

```

```

33 020022 112201
34 020024 001421
35 020026 012700 020374
36 020032 120110
37 020034 001407
38 020036 105720
39 020040 001374
40 020042
   020042 004137 022266
   020046 003702
   020050 000000
41 020052 000406
42 020054 162700 020334
43 020060 006300
44 020062 004770 020346
45 020066 000755
46 020070 000207

```

```

OSTRNG: MOVB (R2)+,R1           :GET CONTROL CHARACTER
        BEQ OSTRE             :EXIT IF NULL CHARACTER
        MOV #ERRC,R0         :GET POINTER TO CHARACTER TABLE
NCONS:  CMPB R1,(R0)         :COMPARE CHARACTER WITH TABLE ENTRY
        BEQ NCONF           :BRANCH IF MATCH FOUND
        TSTB (R0)+          :INCREMENT POINTER
        BNE NCONS           :CONTINUE SEARCH IF NOT END OF TABLE
        PNTF ERRME1         :REPORT BAD CONTROL CHARACTER
                               JSR R1,LPNTF
                               .WORD ERRME1
                               .WORD PNT.CT
NCONF:  BR OSTRE
        SUB #ERRC,R0         :GET INCREMENT INTO TABLE
        ASL R0              :DOUBLE TO WORD COUNT
        CALL @ERRD(R0)      :DISPATCH TO PRINT ROUTINE
        BR OSTRNG          :GET NEXT
OSTRE:  RETURN

```

```

1          ;CONTROL CHARACTER WAS A QUOTE. PRINT ALL CHARACTERS TO THE NEXT QUOTE.
2
3 020072 112200          CON.QU: MOVB (R2)+,R0          ;GET CHARACTER
4 020074 120027 000042  CMPB R0,#"'"          ;CHECK IF ENDING QUOTE
5 020100 001403          BEQ CON.QX          ;IF SO, GO GET NEXT CONTROL CHARACTER
6 020102          PRINT R0          ;PRINT THE CHARACTER
7 020102 004737 022114          BR CON.QU          CALL CPNT
8 020106 000771          CON.QX: RETURN          ;CONTINUE PRINTING
9 020110 000207
10         ;CONTROL CHARACTER WAS AN A. PRINT ASCII CHARACTERS FROM PARAMETERS.
11
12 020112 004737 021024  CON.A: CALL GETCNT          ;GET COUNT OF CHARACTERS
13 020116          CON.A1: PRINT (R4)+          ;PRINT THE CHARACTER
14 020116 112400          MOVB (R4)+,R0          ;COUNT THE CHARACTERS
15 020120 004737 022114          CALL CPNT
16 020124 005301          DEC R1
17 020126 001373          BNE CON.A1          ;PRINT UNTIL COUNT REACHES ZERO
18 020130 032704 000001  BIT #1,R4          ;CHECK IF R4 NOW ODD
19 020134 001401          BEQ CON.A2
20 020136 005204          INC R4          ;IF SO, INCREMENT TO NEXT EVEN ADDRESS
21 020140 000207  CON.A2: RETURN          ;NOW GET NEXT CONTROL CHARACTER
22
23         ;CONTROL CHARACTER WAS A D. PRINT DECIMAL NUMBER.
24
25 020142 012701 000012  CON.D: MOV #10.,R1          ;LOAD RADIX
26 020146 004737 021102  CALL PNTNUM          ;PRINT NUMBER
27 020152 000207          RETURN          ;NOW GET NEXT CONTROL CHARACTER
28
29         ;CONTROL CHARACTER WAS AN H. PRINT HEX NUMBER.
30
31 020154 012701 000020  CON.H: MOV #16.,R1          ;LOAD RADIX
32 020160 004737 021102  CALL PNTNUM          ;PRINT NUMBER
33 020164 000207          RETURN          ;NOW GET NEXT CONTROL CHARACTER
    
```



```

1          ;CONTROL CHARACTER WAS AN O. PRINT OCTAL NUMBER.
2
3 020166 012701 000010  CON.O: MOV #8.,R1          ;LOAD RADIX
4 020172 004737 021102  CALL PNTNUM          ;PRINT NUMBER
5 020176 000207          RETURN          ;NOW GET NEXT CONTROL CHARACTER
6
7          ;CONTROL CHARACTER WAS AN N. PRINT NEW LINE SEQUENCE.
8
9 020200 004737 021024  CON.N: CALL GETCNT          ;GET COUNT
10 020204          CON.N1: PRINT #CR          ;PRINT NEW LINE SEQUENCE
    020204 112700 000015          ;
    020210 004737 022114          ;
11 020214 005301          DEC R1          ;COUNT THE SEQUENCES          MOVB #CR,R0
12 020216 001372          BNE CON.N1          ;
13 020220 000207          RETURN          ;NOW GET NEXT CONTROL CHARACTER
14
15          ;CONTROL CHARACTER WAS AN R. CALL A PRE-PROGRAMMED ROUTINE.
16
17 020222 004737 021024  CON.R: CALL GETCNT          ;GET ROUTINE NUMBER
18 020226 020127 000011  CMP R1,#ERRRSZ          ;CHECK IF DEFINED ROUTINE NUMBER
19 020232 101004          BHI CON.R1
20 020234 060101          ADD R1,R1          ;DOUBLE COUNT TO GET WORD INDEX
21 020236 004771 020300  CALL @ERRRTB-2(R1)          ;CALL ROUTINE
22 020242 000207          RETURN          ;NOW GET NEXT CONTROL CHARACTER
23 020244          CON.R1: PNTF ERRME1          ;REPORT BAD MESSAGE STRING
    020244 004137 022266          ;
    020250 003702          JSR R1,LPNTF
    020252 000000          .WORD ERRME1
24 020254          POP R1          ;FIX THE STACK          .WORD PNT.CT
    020254 012601          ;
25 020256 000207          RETURN          MOV (SP)+,R1
26
27          ;CONTROL CHARACTER WAS AN S. PRINT SPACES.
28
29 020260 004737 021024  CON.S: CALL GETCNT          ;GET COUNT
30 020264          CON.S1: PRINT '<#>'          ;PRINT A SPACE
    020264 112700 000040          ;
    020270 004737 022114          ;
31 020274 005301          DEC R1          ;COUNT THE SPACES          MOVB #',R0
32 020276 001372          BNE CON.S1          ;
33 020300 000207          RETURN          ;NOW GET NEXT CONTROL CHARACTER
    
```



```
1          ;ERROR ROUTINE DISPATCH TABLE
2
3 020302 020366      ERRRTB: .WORD CALR1          ;CALL ALTERNATE PRINT STRING IN DM MEMORY IMAGE
4 020304 020414          .WORD CALR2          ;PRINT AN SDI DIAGNOSE RESPONSE
5 020306 020512          .WORD CALR3          ;DECIDE WHETHER TO PRINT RBN
6 020310 020526          .WORD CALR4          ;PRINT BASIC LINE WITHOUT UDA ADDRESS
7 020312 020602          .WORD CALR5          ;PRINT BASIC LINE WITH UDA ADDRESS
8 020314 020660          .WORD CALR6          ;CALL ALTERNATE PRINT STRING IN PDP-11 MEMORY
9 020316 020674          .WORD CALR7          ;PRINT 'REPLACE UDA MODULE M7161'
10 020320 020712         .WORD CALR8          ;PRINT ' UDASA CONTAINS 'XXXXXX''
11 020322 020730         .WORD CALR9          ;REPRINT LAST NUMBER
12          000011      ERRRSZ=<.-ERRRTB>/2
13
14 020324          Tnames: .WORD BASN1
16 020324 006257          .WORD BASN2
17 020326 006303          .WORD BASN3
18 020330 006323          .WORD BASN4
21 020332 006343
23
24          ;BUILD TWO TABLES
25          ; FIRST CONTAINING CONTROL CHARACTERS
26          ; SECOND CONTAINING ROUTINE ADDRESSES
27
28          .MACRO BUILD
29          ENTRY ",CON.QU
30          ENTRY A,CON.A
31          ENTRY D,CON.D
32          ENTRY H,CON.H
33          ENTRY O,CON.O
34          ENTRY N,CON.N
35          ENTRY R,CON.R
36          ENTRY S,CON.S
37          .ENDM
```

1
2
3
4
5
6
7
8
9 020334
020334 042
020335 101
020336 104
020337 110
020340 117
020341 116
020342 122
020343 123
10 020344 000
11
12
13
14
15
16
17
18
19
20
21 020346
020346 020072
020350 020112
020352 020142
020354 020154
020356 020166
020360 020200
020362 020222
020364 020260

;HERE IS FIRST TABLE
.MACRO ENTRY ARG1,ARG2
.LIST
.BYTE ''ARG1
.NLIST
.ENDM
ERRC: BUILD
.BYTE ''
.BYTE 'A
.BYTE 'D
.BYTE 'H
.BYTE 'O
.BYTE 'N
.BYTE 'R
.BYTE 'S
.BYTE 0
.EVEN

;FOLLOW WITH A NULL BYTE

;HERE IS SECOND TABLE
.MACRO ENTRY ARG1,ARG2
.LIST
.WORD ARG2
.NLIST
.ENDM
ERRD: BUILD
.WORD CON.GU
.WORD CON.A
.WORD CON.D
.WORD CON.H
.WORD CON.O
.WORD CON.N
.WORD CON.R
.WORD CON.S

```
1  
2  
3  
4 020366          :PRE-PROGRAMMED ROUTINE 1  
   020366 010246  :CALL ALTERNATE PRINT STRING IN DM PROGRAM IMAGE  
5 020370 012402  
6 020372 006302  
7 020374 063702 002214  
8 020400 067702 161610  
9 020404 004737 020022  
10 020410  
   020410 012602  
11 020412 000207  
  
CALR1: PUSH R2          :SAVE CURRENT STRING POINTER  
                        MOV R2,-(SP)  
                        :GET NEW STRING POINTER  
MOV (R4)+,R2           :DOUBLE FOR WORD COUNT  
ASL R2                :ADD START OF STRING STORAGE  
ADD DMPROG,R2         :ADD SIZE OF MAIN PROGRAM  
ADD @DMPROG,R2       :OUTPUT USING THIS STRING  
CALL OSTRNG          :GET OLD POINTER BACK  
POP R2               :NOW CONTINUE THE OLD STRING  
                        MOV (SP)+,R2  
RETURN
```


1
2
3
4
5
6
7
8
9 020512 032724 000200
10 020516 001323
11 020520 062704 000006
12 020524 000207

;PRE-PROGRAMMED ROUTINE 3
;DECIDE WHETHER TO PRINT RBN
;
;FOUR PARAMETERS ARE PROVIDED FOR THIS ROUTINE. THE FIRST PARAMETER
;SHOULD BE CHECKED TO SEE IF BIT 7 IS SET:
; IF SET - TURN INTO A CALL TO ROUTINE 1 (WHICH WILL USE OTHER 3 PARAMETERS)
; IF CLEAR - SKIP OVER NEXT 3 PARAMETERS AND END ROUTINE

CALR3: BIT #BIT7,(R4)+ ;CHECK BIT 7 IN FIRST PARAMETER WORD
BNE CALR1 ;IF SET, TURN INTO A CALR1
ADD #6,R4 ;ELSE, SKIP OVER NEXT 3 PARAMETERS
RETURN


```
1 ;PRE-PROGRAMMED ROUTINE 4
2 ;PRINT BASIC LINE FOR HOST PROGRAM ERROR WITHOUT UDA ADDRESS
3 ;THEN SWITCH TO EXTENDED FORMAT
4
5 CALR4: PNTB BASLN,#BASNO,#BAS,#BAS,#BAS
6
7 020526 012746 006436 MOV #BAS,-(SP)
8 020532 012746 006436 MOV #BAS,-(SP)
9 020536 012746 006436 MOV #BAS,-(SP)
020542 012746 006240 MOV #BASNO,-(SP)
020546 004137 022276 JSR R1,LPNTB
020552 006437 .WORD BASLN
020554 000010 .WORD PNT.CT
6 020556 004737 025356 CALL RNTIME
7 020562 PRINT #CR
020562 112700 000015 MOV #CR,R0
020566 004737 022114 CALL CPNT
8 020572 012737 022214 002172 MOV #PX,PTYPE
9 020600 000207 RETURN
```


1
2
3
4
5
6
7
8
9

:PRE-PROGRAMMED ROUTINE 5
:PRINT BASIC LINE FOR HOST PROGRAM ERROR WITH UDA ADDRESS
:THEN SWITCH TO EXTENDED FORMAT

CALR5: PNTB BASLN,#BASNO,#BASL2,(R5),#BAS,#BAS

020602 012746 006436
020602 012746 006436
020606 012746 006436
020612 011546
020614 012746 006402
020620 012746 006240
020624 004137 022276
020630 006437
020632 000012
020634 004737 025356
020640 112700 000015
020644 004737 022114
020650 012737 022214 002172
020656 000207

CALL RNTIME
PRINT #CR

MOV #PX,PType
RETURN

MOV #BAS,-(SP)
MOV #BAS,-(SP)
MOV (R5),-(SP)
MOV #BASL2,-(SP)
MOV #BASNO,-(SP)
JSR R1,LPNTB
.WORD BASLN
.WORD PNT.CT

MOVB #CR,R0
CALL CPNT

```
1  
2  
3  
4 020660  
   020660 010246  
5 020662 012402  
6 020664 004737 020022  
7 020670  
   020670 012602  
8 020672 000207  
  
;PRE-PROGRAMMED ROUTINE 6  
;CALL ALTERNATE PRINT ROUTINE IN PDP-11 MEMORY  
  
CALR6:  PUSH R2  
        MOV (R4)+,R2  
        CALL OSTRNG  
        POP R2  
        RETURN  
  
;SAVE CURRENT STRING POINTER  
        MOV R2,-(SP)  
;GET NEW STRING POINTER  
;OUTPUT USING THIS STRING  
;GET OLD POINTER BACK  
        MOV (SP)+,R2  
;NOW CONTINUE THE OLD STRING
```



```
1  
2  
3  
4 020674 ;PRE-PROGRAMMED ROUTINE 7  
020674 010246 ;PRINT 'REPLACE UDA MODULE M7161'  
5 020676 012702 013205 CALR7: PUSH R2  
6 020702 004737 020022 MOV #XFRU,R2  
7 020706 CALL OSTRNG  
020706 012602 POP R2  
8 020710 000207 RETURN  
MOV R2,-(SP)  
MOV (SP)+,R2
```



```
1  
2  
3  
4 020712          ;PRE-PROGRAMMED ROUTINE 8  
   020712 010246  ;PRINT " UDASA CONTAINS XXXXXX"  
5 020714 012702 013154 CALR8: PUSH R2  
6 020720 004737 020022   MOV #XSA,R2          MOV P2,-(SP)  
7 020724          CALL OSTRNG  
   020724 012602   POP R2  
8 020726 000207   RETURN          MOV (SP)+,R2
```

1
2
3 020730 005744
4 020732 000207

: REPRINT LAST NUMBER
: R4 -> TABLE
CALR9: TST -(R4)
RETURN


```

1      ;GDRVT
2
3      ;GET DRIVE TABLE POINTER
4
5      ;INPUTS:
6      ;       R5 - CONTROLLER TABLE ADDRESS
7      ;       R1 - DRIVE NUMBER
8
9      ;OUTPUTS:
10     ;       R4 - DRIVE TABLE ADDRESS
11     ;       L$LUN - LOADED WITH UNIT NUMBER OF DRIVE
12     ;       Z CLEAR IF DRIVE TABLE NOT FOUND AFTER ERROR PRINTED
13
14 020734 010246      GDRVT: PUSH R2
15 020736 010504      ;MOV R5,R4      ;GET CONTROLLER TABLE ADDRESS      MOV R2,-(SP)
16 020740 062704 000020  ADD #C.DR0,R4      ;ADD OFFSET TO DRIVE TABLE ADDRESS
17 020744 012702 000010  MOV #8,,R2      ;GET COUNT OF DRIVES
18 020750 005714      1$: TST (R4)      ;CHECK IF AN ADDRESS HERE
19 020752 001406      BEQ 3$
20 020754 027401 000000  CMP @ (R4),R1      ;COMPARE DRIVE NUMBERS
21 020760 001412      BEQ 10$      ;BRANCH IF A MATCH
22 020762 005724      2$: TST (R4)+      ;BUMP ADDRESS
23 020764 005302      DEC R2
24 020766 001370      BNE 1$      ;LOOK AT ALL OF THEM
25 020770      3$: ERRDF 35,,ERR035      ;UNIT NUMBER NOT FOUND
26 020770 104455      TRAP C$ERDF
27 020772 000043      .WORD 35
28 020774 000000      .WORD 0
29 020776 013776      .WORD ERR035
30
31 021000      POP R2
32 021000 012602      CLZ      ;CLEAR Z AS ERROR FLAG      MOV (SP)+,R2
33 021002 000244      RETURN
34 021004 000207      10$: MOV (R4),R4      ;GET ADDRESS OF TABLE
35 021006 011404      MOV B D.UNIT(R4),L$LUN      ;GET UNIT NUMBER
36 021010 116437 000002 002074  POP R2
37 021016 012602      SEZ      ;SET Z FLAG      MOV (SP)+,R2
38 021020 000264      RETURN
39 021022 000207
    
```


1									
2									
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13	021024								
	021024	010046							
14	021026	005001							
15	021030	121227	000060						
16	021034	103415							
17	021036	121227	000071						
18	021042	101012							
19	021044	006301							
20	021046	010100							
21	021050	006301							
22	021052	006301							
23	021054	060001							
24	021056	112200							
25	021060	162700	000060						
26	021064	060001							
27	021066	000760							
28	021070	005701							
29	021072	001001							
30	021074	005201							
31	021076								
	021076	012600							
32	021100	000207							

```

;GETCNT
;GET COUNT IN NEXT CHARACTERS OF STRING POINTED TO BY R2.
;NUMBER WILL BE IN DECIMAL. IF NO NUMBER, RETURN A
;DEFAULT OF 1.
;INPUTS:
;R2 - POINTER TO ASCII STRING
;OUTPUTS:
;R1 - NUMBER READ OR A ONE
;R2 - POINTING TO CHARACTER AFTER NUMBER

GETCNT: PUSH R0
;START WITH ZERO COUNT
MOV R0,-(SP)
;CHECK IF CHARACTER A DIGIT
GETCNX: CLRB R1
;BRANCH IF LOWER THAN ZERO
BLO GETCDN
;BRANCH IF HIGHER THAN NINE
CMPB (R2),#'9
BHI GETCDN
;MULTIPLY NUMBER BY 10
ASL R1
;SAVE 2N
MOV R1,R0
;COMPUTE 4N
ASL R1
;COMPUTE 8N
ASL R1
;8N + 2N = 10N
ADD R0,R1
;GET DIGIT FROM STING
MOVB (R2)+,R0
;GET RID OF ASCII
SUB #'0,R0
;ADD TO NUMBER
ADD R0,R1
;GO TO NEXT CHARACTER
BR GETCNX
;CHECK IF NUMBER IS ZERO
GETCDN: TST R1
;IF ZERO, CHANGE
BNE GETCXX
; TO DEFAULT OF ONE
; TO DEFAULT OF ONE
GETCXX: INC R1
POP R0
MOV (SP)+,R0
RETURN
    
```

1			:PNTNUM	
2			:	
3			:PRINT A NUMBER	
4			:	
5			:INPUTS:	
6			: R1 - RADIX OF NUMBER	
7			: R2 - ASCII STRING TO COUNT OF BITS IN NUMBER	
8			: R4 - POINTER TO NUMBER (LOW WORD)	
9			:	
10			:OUTPUTS:	
11			: NUMBER IS PRINTED. LEADING ZEROS ARE PRINTED EXCEPT FOR	
12			: DECIMAL NUMBERS.	
13			: R0 - CONTENTS DESTROYED	
14	021102	010100	PNTNUM: MOV R1,R0	:SAVE RADIX
15	021104	004737 021024	CALL GETCNT	:GET COUNT OF BITS
16	021110		PNTNUS: PUSH <R2,R3,R5>	
	021110	010246		MOV R2,-(SP)
	021112	010346		MOV R3,-(SP)
	021114	010546		MOV R5,-(SP)
17	021116	012403	MOV (R4)+,R3	:GET ONE PARAMETER WORD
18	021120	005005	CLR R5	:CLEAR STORAGE FOR OTHER
19	021122	020127 000020	CMP R1,#16.	:MORE THAN 16 BITS IN NUMBER?
20	021126	003401	BLE 1\$	
21	021130	012405	MOV (R4)+,R5	:YES, GET SECOND PARAMETER WORD
22	021132		1\$: PUSH R4	
	021132	010446		MOV R4,-(SP)
23	021134	010504	MOV R5,R4	:PUT HIGH WORD IN R4
24	021136	012702 000020	MOV #16.,R2	:COMPUTE BITS NOT WANTED
25	021142	160102	SUB R1,R2	:BY SUBTRACTING BITS TO USE
26	021144	002002	BGE 2\$:FROM 16.
27	021146	062702 000020	ADD #16.,R2	:IF NEGATIVE, ADD 16 FOR FIRST WORD
28	021152	001414	2\$: BEQ 6\$:IF ZERO, NO BITS NEED BE CLEARED
29	021154	012705 100000	MOV #BIT15,R5	:START MASK WITH SIGN BIT SET
30	021160	005302	3\$: DEC R2	:COUNT BITS IN MASK
31	021162	001402	BEQ 4\$	
32	021164	006205	ASR R5	:SHIFT MORE BITS TO RIGHT
33	021166	000774	BR 3\$	
34	021170	020127 000020	4\$: CMP R1,#16.	:MORE THAN 16 BITS IN NUMBER?
35	021174	003402	BLE 5\$	
36	021176	040504	BIC R5,R4	:YES, CLEAR IN HIGH WORD
37	021200	000401	BR 6\$	
38	021202	040503	5\$: BIC R5,R3	:NO, CLEAR IN LOW WORD
39	021204	004737 021344	6\$: CALL DIVIDE	:DIVIDE BY RADIX IN R0
40	021210		PUSH R5	:PUSH REMAINDER ON STACK
	021210	010546		MOV R5,-(SP)
41	021212	005202	INC R2	:COUNT DIGITS ON STACK
42	021214	005703	TST R3	:CHECK IF QUOTIENT IS ZERO
43	021216	001372	BNE 6\$	
44	021220	005704	TST R4	
45	021222	001370	BNE 6\$	

1	021224	020027	000012		CMP R0,#10.		
2	021230	001423			BEQ 10\$		
3	021232	010103			MOV R1,R3		
4	021234	162700	000014		SUB #12.,R0		
5	021240	003002			BGT 7\$		
6	021242	012700	000003		MOV #3,R0		
7	021246	004737	021344	7\$:	CALL DIVIDE		
8	021252	005705			TST R5		
9	021254	001401			BEQ 8\$		
10	021256	005203			INC R3		
11	021260	160203		8\$:	SUB R2,R3		
12	021262	001406			BEQ 10\$		
13	021264			9\$:	PRINT #'0		
	021264	112700	000060				
	021270	004737	022114				MOVB #'0,R0
14	021274	005303			DEC R3		CALL CPNT
15	021276	001372			BNE 9\$		
16							
17	021300			10\$:	POP R5		
	021300	012605					
18	021302	062705	000060		ADD #'0,R5		
19	021306	020527	000071		CMP R5,#'9		
20	021312	003402			BLE 11\$		
21	021314	062705	000007		ADD #'<'A-'9-1>,R5		
22	021320			11\$:	PRINT R5		
	021320	110500					
	021322	004737	022114				MOVB R5,R0
23	021326	005302			DEC R2		CALL CPNT
24	021330	001363			BNE 10\$		
25	021332				POP <R4,R5,R3,R2>		
	021332	012604					MOV (SP)+,R4
	021334	012605					MOV (SP)+,R5
	021336	012603					MOV (SP)+,R3
	021340	012602					MOV (SP)+,R2
26	021342	000207			RETURN		

;IF RADIX IS DECIMAL
 ; JUST GO PRINT DIGITS ON STACK
 ;OTHERWISE COMPUTE NUMBER OF LEADING ZEROS
 ;DIVIDEND IS BITS IN NUMBER
 ;DIVISOR IS BITS PER DIGIT PRINTED
 ; (3 OR 4)

;IF REMAINDER NOT ZERO
 ;INCREMENT QUOTIENT

;SUBTRACT DIGITS ON STACK
 ;NO LEADING ZEROS IF ZERO
 ;PRINT A ZERO

;REPEAT UNTIL COUNT REACHES ZERO

;GET CHACACTER FROM STACK
 MOV (SP)+,R5

;CNVERT TO ASCII DIGIT
 ;IF GREATER THAN A 9
 ; CONVERT TO A OR HIGHER
 ; FOR HEX DIGIT
 ;PRINT THE CHARACTER

;REPEAT FOR ALL DIGITS
 ; ON STACK


```

1      ;DIVIDE
2      :
3      :DIVIDE A 32 BIT UNSIGNED NUMBER BY A 16 BIT UNSIGNED NUMBER.
4      :REPLACE DIVIDEND WITH QUOTIENT AND RETURN REMAINDER.
5      :WILL NOT CHECK FOR DIVIDE BY ZERO.
6      :
7      :INPUTS:
8      :       R3 - LOW 16 BITS OF DIVIDEND
9      :       R4 - HIGH 16 BITS OF DIVIDEND
10     :       R0 - DIVISOR
11     :
12     :OUTPUTS:
13     :       R3 - LOW 16 BITS OF QUOTIENT
14     :       R4 - HIGH 16 BITS OF QUOTIENT
15     :       R5 - REMAINDER
16     DIVIDE: PUSH R2
17     021344 010246          MOV R2,-(SP)
18     021346 012702 000040  ;SET UP SHIFT COUNT
19     021352 005005          CLR R5
20     021354 006303          ;START WITH ZERO REMAINDER
21     021356 006104          ;SHIFT LEFT INTO R5
22     021360 006105          1$: ASL R3
23     021362 020005          ROL R4
24     021364 101002          ROL R5
25     021366 160005          CMP R0,R5
26     021370 005203          ;WILL DIVISOR GO INTO REMAINDER
27     021372 005302          ;ONLY SUBTRACT IF IT WILL
28     021374 001367          ;SUBTRACT DIVISOR
29     021376 012602          ;PUT A ONE INTO QUOTIENT
30     021400 000207          ;COUNT THE SHIFTS
31     021376 012602          2$: DEC R2
32     021400 000207          BNE 1$
33     021400 000207          POP R2
34     021400 000207          MOV (SP)+,R2
35     021400 000207          RETURN
    
```

```

1      ;BUILD DEFAULT 28-BIT NUMBER
2
3      ;INPUT:
4      R4 - POINTER TO 2 WORD DEFAULT NUMBER
5
6      ;OUTPUT:
7      TEMP - ASCIZ STRING REPRESENTING DEFAULT NUMBER
8      BLD28: PUSH <R0,R1,R3,R4,R5>
          MOV R0,-(SP)
          MOV R1,-(SP)
          MOV R3,-(SP)
          MOV R4,-(SP)
          MOV R5,-(SP)
9      021402 010046
10     021404 010146
11     021406 010346
12     021410 010446
13     021412 010546
14     021414 011403
15     021416 016404 000002
16     021422 012700 000012
17     021426 005001
18     021430 004737 021344
19     021434 062705 000060
20     021440
21     021442 010546
22     021444 005201
23     021446 010305
24     021448 050405
25     021450 001367
26     021452 012700 002262
27     021456
28     021458 012605
29     021460 110520
30     021462 005301
31     021464 001374
32     021466 105020
33     021470
34     021472 012605
35     021474 012604
36     021476 012603
37     021478 012601
38     021500 012600
39     021502 000207
          MOV (R4),R3 ;GET NUMBER
          MOV 2(R4),R4
          MOV #10,,R0 ;DIVISOR IS 10.
          CLR R1 ;CLEAR CHARACTER COUNT
          CALL DIVIDE
          ADD #'0',R5 ;CONVERT REMAINDER TO ASCII CHARACTER
          PUSH R5 ;STORE ON STACK
          MOV R5,-(SP)
          INC R1 ;COUNT THE CHARACTER
          MOV R3,R5 ;REPEAT UNTIL QUOTIENT IS ZERO
          BIS R4,R5
          BNE 1$
          MOV #TEMP,R0 ;GET POINTER TO STRING
          POP R5 ;PUT CHARACTERS INTO STRING
          MOV (SP)+,R5
          MOVB R5,(R0)+
          DEC R1
          BNE 2$
          CLRB (R0)+
          POP <R5,R4,R3,R1,R0> ;END WITH NULL
          MOV (SP)+,R5
          MOV (SP)+,R4
          MOV (SP)+,R3
          MOV (SP)+,R1
          MOV (SP)+,R0
          RETURN
    
```



```
1 021604          PNTF INP28A          ;PRINT PROPER RANGE
  021604 004137 022266          JSR R1,LPNTF
  021610 005155          .WORD INP28A
  021612 000000          .WORD PNT.CT
2 021614 000261          SEC          ;SET CARRY TO ASK AGAIN
3 021616 000411          BR 4$
4
5 021620          2$: PNTF INP28B          ;PRINT ILLEGAL CHARACTER
  021620 004137 022266          JSR R1,LPNTF
  021624 005216          .WORD INP28B
  021626 000000          .WORD PNT.CT
6 021630 000261          SEC
7 021632 000403          BR 4$
8
9 021634 010024          3$: MOV R0,(R4)+          ;MOVE NUMBER TO STORAGE AREA
10 021636 010124          MOV R1,(R4)+
11 021640 000241          CLC          ;CLEAR CARRY TO INDICATE ALL IS WELL
12 021642          4$: POP <R3,R2,R1,R0>
  021642 012603          MOV (SP)+,R3
  021644 012602          MOV (SP)+,R2
  021646 012601          MOV (SP)+,R1
  021650 012600          MOV (SP)+,R0
13 021652 000207          RETURN
```

```

1
2
3 021654      ;PRINT HEX NUMBERS WITH LEADING SPACE
   021654 112700 000040
   021660 004737 022114
4 021664      T2PNTW: PRINT <#>
   021664 010146      ;PRINT A SPACE
   021666 000301      MOVB #' ,R0
   021670 004737 021714      CALL CPNT
   021674 012601      PUSH R1
   021676 004737 021714      SWAB R1
   021702 000207      CALL T2PNT
   021702 000207      POP R1
   021702 000207      ;PRINT HIGH TWO DIGITS
   021702 000207      ;PRINT LOW TWO DIGITS
   021702 000207      MOV (SP)+,R1
10
11 021704      T2PNTB: PRINT <#>
   021704 112700 000040      ;PRINT A SPACE
   021710 004737 022114      MOVB #' ,R0
   021710 004737 022114      CALL CPNT
12
13
14
15 021714      ;PRINT TWO HEX DIGITS FROM NUMBER IN R1
   021714 010146      T2PNT: PUSH R1
   021716 006001      ;SAVE NUMBER
   021720 006001      ROR R1
   021722 006001      ;SHIFT TO GET HIGH DIGIT
   021724 006001      ROR R1
   021726 004737 021734      ROR R1
   021732 012601      CALL T2PNTD
   021732 012601      POP R1
   021734 042701 177760      ;PRINT TWO DIGITS
   021740 062701 000060      ;GET LOW DIGIT AGAIN
   021744 020127 000071      MOV (SP)+,R1
   021750 003402      T2PNTD: BIC #^C17,R1
   021752 062701 000007      ADD #'0,R1
   021756 110100      CMP R1,#'9
   021760 004737 022114      BLE T2PNTD
   021764 000207      ADD #<'A-'9-1>,R1
   021764 000207      T2PNTD: PRINT R1
   021764 000207      ;CLEAR OTHER BITS
   021764 000207      ;CONVERT TO ASCII CHARACTER
   021764 000207      ;IF GREATER THAN A 9
   021764 000207      ; CONVERT TO A OR HIGHER
   021764 000207      ; FOR HEX DIGIT
   021764 000207      ;PRINT THE DIGIT
   021764 000207      MOVB R1,R0
   021764 000207      CALL CPNT
28 021764      RETURN
    
```



```

1      :T2GNUM
2
3      :GET A HEX DIGIT FROM AN ASCII INPUT STRING
4
5      :INPUTS:
6          R1 - STRING POINTER
7
8      :OUTPUTS:
9          R4 - NUMBER
10         R1 - UPDATED STRING TO CHARACTER AFTER NUMBER
11         R0 - COUNT OF DIGITS (0 IF END OF LINE FOUND)
12
12 021766 005000      T2GNUM: CLR R0                ;CLEAR DIGIT COUNT
13 021770 105711      TSTB (R1)                ;CHECK IF END OF LINE
14 021772 001442      BEQ T2GNX                ;REPORT NULL CHARACTER FOUND
15 021774 121127 000040  CMPB (R1),#'          ;CHECK IF A SPACE
16 022000 001002      BNE T2GND1              ;IF SO, IGNORE IT
17 022002 005201      INC R1
18 022004 000770      BR T2GNUM
19 022006 005004      T2GND1: CLR R4            ;CLEAR NUMBER STORAGE
20 022010 000000      T2GND2: PUSH R2          ;SAVE REGISTER
21 022010 010246      MOV R2,-(SP)
22 022012 112102      MOVB (R1)+,R2           ;GET CHARACTER
23 022014 162702 000060  SUB #'0,R2          ;CONVERT TO HEX DIGIT
24 022020 100431      BMI T2GNE
25 022022 020227 000011  CMP R2,#9
26 022026 003410      BLE T2GND3
27 022030 020227 000021  CMP R2,#<'A-'0>
28 022034 103423      BLO T2GNE
29 022036 020227 000026  CMP R2,#<'F-'0>
30 022042 101020      BHI T2GNE
31 022044 162702 000007  SUB #'A-'9-1>,R2
32 022050 006304      T2GND3: ASL R4
33 022052 006304      ASL R4
34 022054 006304      ASL R4
35 022056 006304      ASL R4
36 022060 050204      BIS R2,R4
37 022062 005200      INC R0
38 022064 012602      POP R2
39 022066 105711      MOV (SP)+,R2
40 022070 001403      TSTB (R1)
41 022072 121127 000040  BEQ T2GNX
42 022074 001344      CMPB (R1),#'
43 022100 005700      BNE T2GND2
44 022102 000207      T2GNX: TST R0
45 022104 000000      RETURN
46
46 022104 012602      T2GNE: POP R2           ;CLEAN UP THE STACK
47 022106 000000      POP R0
47 022106 012600      MOV (SP)+,R2
47 022110 000137 016440  JMP T2CMDE
47 022110 000137 016440  MOV (SP)+,R0
    
```



```

1      ;PRINT ONE CHARACTER
2
3      ;CALL WITH MACRO PRINT
4
5 022114 110037 002174      CPNT:  MOV B R0,ERRCHR
6 022120                                PUSH R1
7 022120 010146                                MOV R1,-(SP)
8 022122 012701 003640      MOV #ERRONE,R1
9 022126 120027 000015      CMPB R0,#CR
10 022132 001002      BNE 1$
11 022134 012701 003643      MOV #ERRNL,R1
12 022140 000177 160026      1$:  JMP @PTYPE
13 022144                                PF:  PRINTF R1,#ERRCHR
14 022144 012746 002174      MOV #ERRCHR,-(SP)
15 022150 010146      MOV R1,-(SP)
16 022152 012746 000002      MOV #2,-(SP)
17 022156 010600      MOV SP,R0
18 022160 104417      TRAP C$PNTF
19 022162 062706 000006      ADD #6,SP
20 022166 000435
21 022170                                PB:  BR CPNTX
22 022170 012746 002174      PRINTB R1,#ERRCHR
23 022174 010146      MOV #ERRCHR,-(SP)
24 022176 012746 000002      MOV R1,-(SP)
25 022202 010600      MOV #2,-(SP)
26 022204 104414      MOV SP,R0
27 022206 062706 000006      TRAP C$PNTB
28 022212 000423      ADD #6,SP
29 022214                                PX:  BR CPNTX
30 022214 012746 002174      PRINTX R1,#ERRCHR
31 022220 010146      MOV #ERRCHR,-(SP)
32 022222 012746 000002      MOV R1,-(SP)
33 022226 010600      MOV #2,-(SP)
34 022230 104415      MOV SP,R0
35 022232 062706 000006      TRAP C$PNTX
36 022236 000411      ADD #6,SP
37 022240                                PS:  BR CPNTX
38 022240 012746 002174      PRINTS R1,#ERRCHR
39 022244 010146      MOV #ERRCHR,-(SP)
40 022246 012746 000002      MOV R1,-(SP)
41 022252 010600      MOV #2,-(SP)
42 022254 104416      MOV SP,R0
43 022256 062706 000006      TRAP C$PNTS
44 022262 000411      ADD #6,SP
45 022262 012601      CPNTX: POP R1
46 022264 000207      RETURN
47                                MOV (SP)+,R1
    
```

```

1          ;PRINT FORMATTED MESSAGE
2
3          ;CALL WITH MACRO PNT, PNTF, PNTB, PNTX, OR PNTS
4
5 022266 012737 022144 002172 LPNTF: MOV #PF,PTYPE
6 022274 000413                BR LPNT
7 022276 012737 022170 002172 LPNTB: MOV #PB,PTYPE
8 022304 000407                BR LPNT
9 022306 012737 022214 002172 LPNTX: MOV #PX,PTYPE
10 022314 000403               BR LPNT
11 022316 012737 022240 002172 LPNTS: MOV #FS,PTYPE
12 022324                LPNT:  PUSH <R2,R3,R4,R5>
    022324 010246                MOV R2,-(SP)
    022326 010346                MOV R3,-(SP)
    022330 010446                MOV R4,-(SP)
    022332 010546                MOV R5,-(SP)
13 022334 012102                MOV (R1)+,R2          ;GET ADDRESS OF STRING
14 022336 010604                MOV SP,R4           ;COMPUTE ADDRESS OF ARGUMENTS
15 022340 062704 000012        ADD #10.,R4        ; WHICH ARE NOW ON STACK (IF ANY)
16 022344                PUSH R1           ;SAVE RETURN ADDRESS
    022344 010146                MOV R1,-(SP)
17 022346 004737 020022        CALL OSTRNG        ;PRINT THE FORMATTED MESSAGE
18 022352                POP <R0,R5,R4,R3,R2,R1> ;RESTORE ALL REGISTERS
    022352 012600                MOV (SP)+,R0
    022354 012605                MOV (SP)+,R5
    022356 012604                MOV (SP)+,R4
    022360 012603                MOV (SP)+,R3
    022362 012602                MOV (SP)+,R2
    022364 012601                MOV (SP)+,R1
19 022366 062006                ADD (R0)+,SP      ;ADJUST STACK POINTER OVER ARGUMENTS
20 022370 000110                JMP @R0           ;RETURN
    
```



```

1      :PNTERR
2
3
4
5
6
7
8
9
10     :PRINT ERROR MESSAGE FROM DM PROGRAM REQUEST 11 OR 12.
11
12     :INPUTS:
13     :       R5 - CONTROLLER TABLE ADDRESS
14     :       R4 - MESSAGE DATA ADDRESS
15     :       R3 - COMMAND DATA ADDRESS
16     :OUTPUTS:
17     :       ERROR MESSAGE PRINTED
18     :       BIT 15 SET IN COMMAND DATA IF DRIVE HAS BEEN DROPPED
19
20     PNTERR: PUSH <R0,R1,R2>
21
22     022372 010046
23     022374 010146
24     022376 010246
25     14 022400 005764 000004
26     15 022404 002004
27     16 022406 116537 000002 002074
28     17 022414 000416
29     18 022416 010446
30     19 022420 016401 000004
31     20 022424 004737 020734
32     21 022430 001036
33     22 022432 005764 000002
34     23 022436 100004
35     24 022440 052713 100000
36     28 022444
37     29 022446 012604
38     30 022450 000423
39     31 022452 012604
40     32 022456 012702 002162
41     33 022462 016412 000002
42     34 022464 006112
43     35 022466 006112
44     36 022470 042722 177774
45     37 022474 016412 000002
46     38 022500 042722 140000
47     39 022504 005022
48     40 022506 012712 014246
49     41 022512
50     42 022514 104460
51     43 022516 000241
52     44 022524 012602
53     45 022526 012601
54     46 022530 012600
55     47 022534 000207
56     48 022536 000261
57     49 022538 000772
58
59     :GET DRIVE NUMBER
60     :CHECK IF BIT 15 SET
61     :IF SO, GET UNIT FROM CONTROLLER TABLE
62
63     :SAVE DATA ADDRESS
64     :GET DRIVE NUMBER
65     :GET DRIVE TABLE ADDRESS
66     :IF UNIT DROPPED, EXIT
67     :SEE IF UNIT HAS BEEN DROPPED FROM TESTING
68     : PROCEED IF STILL TO BE TESTED
69     :TELL DM PROGRAM TO STOP TESTING THIS UNIT
70
71     :RESTORE DATA ADDRESS
72     :GET POINTER TO ERROR TABLE
73     :GET ERROR TYPE
74
75     :CLEAR LOW 2 BITS
76
77     :MASK LOW 14 BITS
78     :CLEAR MESSAGE POINTER
79     :GET ROUTINE NUMBER
80     :PRINT THE ERROR MESSAGE
81     TRAP C$ERROR
82     :DRIVE HAS NOT BEEN DROPPED
83
84     MOV R0,-(SP)
85     MOV R1,-(SP)
86     MOV R2,-(SP)
87
88     TST 4(R4)
89     BGE 1$
90     MOV B,C.UNIT(R5),L$LUN
91     BR 2$
92     1$: PUSH R4
93
94     MOV 4(R4),R1
95     CALL GTDRV
96     BNE 5$
97     TST D.UNIT(R4)
98     BPL 3$
99     BIS #BIT15,(R3)
100    POP R4
101
102    BR 4$
103    3$: POP R4
104
105    MOV #ERRTYP,R2
106    MOV 2(R4),(R2)
107    ROL (R2)
108    ROL (R2)
109    ROL (R2)
110    BIC #^C3,(R2)+
111    MOV 2(R4),(R2)
112    BIC #140000,(R2)+
113    CLR (R2)+
114    MOV #ERRRTN,(R2)
115    ERROR
116
117    CLC
118    4$: POP <R2,R1,R0>
119
120    RETURN
121    SEC
122    BR 4$
123
124    MOV (SP)+,R2
125    MOV (SP)+,R1
126    MOV (SP)+,R0
127
128    :DRIVE HAS BEEN DROPPED
    
```


1
2
3
4
5
6
7
8
9
10
11
12
13
14
28
29
30
31
32
33
34
35
36
37
38

```

:LOADDM
:LOAD AND START A DM PROGRAM INTO A CONTROLLER
:INPUTS:
:   R5 - CONTROLLER TABLE ADDRESS
:IMPLICIT INPUTS:
:   DMPROG - POINTER TO START OF DM PROGRAM IN MEMORY
:OUTPUTS:
:   IF LOAD SUCCEEDS - Z CLEAR
:   CONTROLLER TAPE MARKED LOADED
:   IF ERROR - Z SET
    
```

```

14 022532
28 022532 016504 000004
29 022536
   022536 042704 177000
30 022542 010501
31 022544 062701 000010
32 022550
   022550 012746 000340
   022554 010146
   022556 010446
   022560 012746 000003
   022564 104437
   022566 062706 000010
33
34 022572 006204
35 022574 006204
36 022576 004737 023666
37 022602 001560
38 022604 004737 014444
    
```

```

LOADDM:
MOV C.VEC(R5),R4           ;GET VECTOR OF UDA
AND CT.VEC,R4
MOV R5,R1                 BIC #^C<CT.VEC>,R4
ADD #C.JSR,R1            ;GET INTERRUPT SERVICE LINK
SETVEC R4,R1,#PRI07      ;SET UP INTERRUPT VECTOR
                          MOV #PRI07,-(SP)
                          MOV R1,-(SP)
                          MOV R4,-(SP)
                          MOV #3,-(SP)
                          TRAP CSSVEC
                          ADD #10,SP
ASR R4                    ;INITIALIZE UDA WITH SMALLEST
ASR R4                    ;POSITION VECTOR FOR UDA
CALL UDAINT              ; RING BUFFER AND INTERRUPTS ENABLED
BEQ LOADER               ;BRANCH IF AN ERROR
CALL HCOMM                ;ALLOCATE SPACE FOR HOST COMM AREA
    
```

2	022610	023727	002230	000001		CMP TNUM,#1	:IF TEST NUMBER 1
3	022616	001440				BEQ LOADT1	: DO SPECIAL LOAD
5	022620	017701	157370			MOV @DMPROG,R1	:GET SIZE OF PROGRAM
6	022624	012700	000002		LOADB:	MOV #OP.ESP,R0	:BUILD EXECUTE SUPPLIED PROGRAM COMMAND PACKET
7	022630	004737	023150			CALL BLDCMD	
8	022634	013764	002214	000124		MOV DMPROG,HC.CPK+P.UADR(R4)	:LOAD MAIN PROGRAM ADDRESS
9	022642	010164	000120			MOV R1,HC.CPK+P.BCNT(R4)	: AND SIZE
10	022646	013764	002214	000140		MOV DMPROG,HC.CPK+P.OVRL(R4)	:LOAD OVERLAY ADDRESS
11	022654	067764	157334	000140		ADD @DMPROG,HC.CPK+P.OVRL(R4)	
15	022662	004737	023234			CALL SNDCMD	:SEND COMMAND TO UDA
16	022666	004737	023374			CALL WAITMS	:WAIT FOR MESSAGE RESPONSE
17	022672	032764	000037	000032		BIT #ST.MSK,HC.MPK+P.STS(R4)	:CHECK FOR ERRORS
18	022700	001115				BNE LOADE1	
19	022702	042765	000024	000014		BIC #CT.CMD+CT.REQ,C.FLG(R5)	:CLEAR COMMAND OUTSTANDING FLAG
20	022710	052765	000002	000014		BIS #CT.RN,C.FLG(R5)	:SET DM PROGRAM RUNNING FLAG
21	022716	000207				RETURN	

```

1          ;LOAD DM PROGRAM FROM MEMORY SPACE TESTED DURING
2          ;INITIALIZATION IN TEST 1
3
4 022720 017704 157270   LOADT1: MOV @DMPROG,R4           ;GET SIZE OF DM PROGRAM IN BYTES
5 022724 162704 000040   SUB #DMMAIN,R4
6 022730 013700 002214   MOV DMPROG,R0           ;GET ADDRESS OF DM PROGRAM
7 022734 062700 000040   ADD #DMMAIN,R0
8 022740 005001         CLR R1                   ;START WITH OFFSET OF ZERO
9
10 022742 012703 000214  LT1L1: MOV #<HC.BSZ*2>,R3      ;GET SIZE OF BOTH BUFFERS
11 022746 020403        CMP R4,R3                ;IF FEWER BYTES REMAINING IN PROGRAM
12 022750 103001        BHIS LT11
13 022752 010403        MOV R4,R3                ;USE ACTUAL BYTE COUNT
14 022754         LT11:  PUSH R3                   ;SAVE THE BYTE COUNT
15 022756 010346         MOV R3,-(SP)           ;GET ADDRESS OF BUFFER
16 022762 013702 002176   MOV FFREE,R2
17 022766 162702 000214   SUB #<HC.BSZ*2>,R2      ;SAVE BUFFER ADDRESS
18 022766 010246         PUSH R2
19 022770 012022        MOV (R0)+,(R2)+          ;MOVE DATA TO BUFFER
20 022772 162703 000002   SUB #2,R3                ;COUNT BYTES
21 023000 001374        BNE LT1L2
22 023002 012602        POP R2                   ;RESTORE BUFFER ADDRESS
23 023004 012603        POP R3                   ;RESTORE BYTE COUNT
24 023010 004737 023032   CALL LOAD                ;LOAD INTO UDA
25 023012 006203        BEQ LOADER            ;IF ERROR, GET OUT NOW
26 023014 060301        ASR R3                ;CONVERT BYTES TO WORDS
27 023016 006303        ADD R3,R1              ;INCREASE OFFSET FOR NEXT BUFFER
28 023020 160304        ASL R3                ;CONVERT WORDS TO BYTES
29 023022 001347        SUB R3,R4              ;REDUCE REMAINING BYTE COUNT
30 023024 012701 000040   BNE LT1L1                ;GET NEXT BUFFER
31 023030 000675        MOV #DMMAIN,R1        ;GET A BYTE COUNT OF HEADER ONLY
                          BR LOADB                ;NOW START
    
```



```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15 023032
    023032 010046
    023034 010346
    023036 010446
16 023040 012700 000031
17 023044 004737 023150
18 023050 010264 000124
19 023054 010364 000120
20 023060 010164 000144
21 023064 012764 000001 000140
22 023072 004737 023234
23 023076 004737 023374
24 023102 001420
25 023104 032764 000037 000032
26 023112 001010
27 023114 042765 000004 000014
28 023122
    023122 012604
    023124 012603
    023126 012600
29 023130 000244
30 023132 000207

:LOAD
:ISSUE DOWNLINE LOAD COMMAND TO UDA. CHECK THAT LOAD
:HAPPENS WITHOUT ERROR.
:INPUTS:
:   R1 - OFFSET FOR DM PROGRAM
:   R2 - ADDRESS OF BUFFER CONTAINING PROGRAM
:   R3 - SIZE OF BUFFER IN BYTES
:   R5 - CONTROLLER TABLE ADDRESS
:OUTPUTS:
:   Z CLEAR IF NO ERROR
:   Z SET IF ERROR AND ERROR REPORTED
LOAD:  PUSH <R0,R3,R4>

MOV #OP.MWR,R0
CALL BLDCMD
MOV R2,HC.CPK+P.UADR(R4)
MOV R3,HC.CPK+P.BCNT(R4)
MOV R1,HC.CPK+P.RGOF(R4)
MOV #1,HC.CPK+P.RGID(R4)
CALL SNDCMD
CALL WAITMS
BEQ  LOADER
BIT #ST.MSK,HC.MPK+P.STS(R4)
BNE LOADE1
BIC #CT.CMD,C.FLG(R5)
POP <R4,R3,R0>

:GET DOWNLINE LOAD COMMAND
:BUILD COMMAND PACKET
:STUFF IN BUFFER ADDRESS
:STUFF IN BYTE COUNT
:STUFF IN OFFSET
:STUFF IN REGION ID 1
:SEND COMMAND TO UDA
:WAIT FOR MESSAGE RESPONSE
: IF FAILED, EXIT
:LOOK FOR ANY ERROR

:CLEAR COMMAND ISSUED

MOV (SP)+,R4
MOV (SP)+,R3
MOV (SP)+,R0

CLZ
RETURN

;CLEAR Z TO INDICATE NO ERROR
    
```

1
2
3 023134
023134 104455
023136 000042
023140 000000
023142 013770
4 023144 000264
5 023146 000207

;UDA FAILED TO DOWNLINE LOAD DM PROGRAM
LOADE1: ERRDF 34,,ERR034

LOADER: SEZ
RETURN

TRAP CSERDF
.WORD 34
.WORD 0
.WORD ERR034
;SET Z TO INDICATE ERROR OCCURRED

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31

```

:BLDCMD
:BUILD A COMMAND IN COMMAND PACKET
:INPUTS:
:   R5 - CONTROLLER TABLE ADDRESS
:   R0 - COMMAND CODE
:OUTPUTS:
:   R4 - ADDRESS OF HOST COMM AREA
:   COMMAND PACKET CONTAINING REF NUMBER AND OPCODE. ALL OTHER FIELDS CLEARED.
:   CMD REFERENCE NUMBER IN CONTROLLER TABLE INCREMENTED AND RESULT
:   IN COMMAND PACKET.
:   R0 - CONTENTS DESTROYED
    
```

```

15 023150
   023150 010146
   023152 010046
16 023154 016504 000016
17 023160 010400
18 023162 062700 000100
19 023166 012720 000060
20 023172 012701 001000
21 023176 022716 000031
22 023202 001002
23 023204 012701 177777
24 023210 010120
25 023212 012701 000030
26 023216 005020
27 023220 005301
28 023222 001375
29 023224
   023224 012664 000114
30 023230
   023230 012601
31 023232 000207
    
```

```

BLDCMD: PUSH <R1,R0>
                                           MOV R1,-(SP)
                                           MOV R0,-(SP)
                                           ;GET ADDRESS OF HOST COMM AREA
                                           ;COPY TO R0
                                           ;COMPUTE ADDRESS OF COMMAND ENVELOPE
                                           ;LOAD PACKET LENGTH
                                           ;LOAD DIAG CIRCUIT IDENTIFIER
                                           ;IF CODE IS MAINTENANCE WRITE
                                           ; GET OTHER CIRCUIT IDENTIFIER
BLDC0: MOV C.RING(R5),R4
      MOV R4,R0
      ADD #HC.CEV,R0
      MOV #HC.PSZ,(R0)+
      MOV #DUP,R1
      CMP #OP.MWR,(SP)
      BNE BLDC0
      MOV #DIAG,R1
      ;PUT IDENTIFIER INTO PACKET
      MOV R1,(R0)+
      ;GET WORDS TO CLEAR
      MOV #<HC.PSZ>/2,R1
      ;CLEAR PACKET
BLDC1: CLR (R0)+
      DEC R1
      BNE BLDC1
      POP HC.CPK+P.OPCD(R4)
      ;PUT OPCODE IN PACKET
      POP R1
      ;RESTORE R1
      MOV (SP)+,HC.CPK+P.OPCD(R4)
      MOV (SP)+,R1
      RETURN
    
```



```

1      :SNDCMD
2
3      :SEND A COMMAND TO THE UDA.
4      :CLEAR THE RESPONSE PACKET. MARK BOTH PACKETS AVAILABLE TO THE
5      :UDA. SET COMMAND ISSUED BIT IN CONTROLLER TABLE AND INITIALIZE
6      :TIMEOUT COUNTER.
7
8      :INPUTS:
9      :   R5 - CONTROLLER TABLE ADDRESS
10     :OUTPUTS:
11     :   R4 - ADDRESS OF HOST COMM AREA
12
13
14     023234      SNDCMD: PUSH <R0,R1>
15     023234      010046
16     023236      010146
17     023240      016504 000016
18     023244      005265 000044
19     023250      016564 000044 000104
20     023256      012700 000014
21     023262      060400
22     023264      012701 000032
23     023270      005020
24     023272      005301
25     023274      001375
26     023276      012764 140000 000006
27     023304      012764 100000 000012
28     023312      005775 000000
29     023316      052765 000004 000014
30     023324      012601
31     023326      012600
32     023330      000207
33
34     SNDC1:
35     CLR (R0)+
36     DEC R1
37     BNE SNDC1
38     MOV #RG.CWN+RG.FLG,HC.MCT(R4)
39     MOV #RG.OWN,HC.CCT(R4)
40     TST @ (R5)
41     BIS #CT.CMD,C.FLG(R5)
42     POP <R1,R0>
43
44     MOV R0,-(SP)
45     MOV R1,-(SP)
46     :LOAD R4 WITH HOST COMM AREA ADDRESS
47     :INCREMENT CMD REFERENCE NUMBER
48     :PUT IN PACKET
49     :POINT TO MESSAGE ENVELOPE
50     :SIZE OF MESSAGE PACKET
51     :CLEAR ENTIRE MESSAGE PACKET
52
53     :MARK MESSAGE PACKET AVAILABLE
54     :MARK COMMAND TO UDA
55     :TELL UDA COMMAND IS THERE
56     :MARK COMMAND ISSUED
57
58     MOV (SP)+,R1
59     MOV (SP)+,R0
60
61     RETURN
    
```

```

1      :CLRBUF
2      :
3      :CLEAR THE SPECIFIED DATA BUFFER IN THE HOST COMM AREA
4      :AND LOAD BUFFER DESCRIPTOR IN COMMAND PACKET TO THE BUFFER
5      :
6      :INPUTS:
7      :
8      :       R5 - CONTROLLER TABLE ADDRESS
9      :       R4 - ADDRESS OF HOST COMM AREA
10     :       R0 - OFFSET INTO HOST COMM AREA TO DATA BUFFER
11     :
12     :OUTPUTS:
13     :       DATA BUFFER CLEARED
14     :       COMMAND PACKET POINTING TO BUFFER
15     :       BYTE COUNT SET TO SIZE OF BUFFER
16     :       R4 - ADDRESS OF DATA BUFFER
17     CLRBUF: PUSH <R0,R1>
18
19     023332 010046
20     023332 010146
21     023334 010146
22     023336 060400
23     023340 010064 000124 000120
24     023344 012764 000106 000120
25     023352 010004
26     023354 012701 000043
27     023360 005020
28     023362 005301
29     023364 001375
30     023366 012601
31     023370 012600
32     023372 000207
33
34     ADD R4,R0
35     MOV R0,HC.CPK+P.UADR(R4)
36     MOV #HC.BSZ,HC.CPK+P.BCNT(R4)
37     MOV R0,R4
38     MOV #HC.BSZ/2,R1
39     CLRBFL: CLR (R0)+
40             DEC R1
41             BNE CLRBFL
42             POP <R1,R0>
43
44     ;ADD START OF HOST COMM AREA TO OFFSET
45     ;PUT BUFFER ADDRESS IN COMMAND PACKET
46     ;PUT SIZE OF BUFFER IN COMMAND PACKET
47     ;PUT BUFFER ADDRESS IN R4
48     ;GET SIZE OF BUFFER IN WORDS
49     ;CLEAR ALL THE WORDS
50
51     MOV R0,-(SP)
52     MOV R1,-(SP)
53
54     MOV (SP)+,R1
55     MOV (SP)+,R0
56
57     RETURN
    
```



```

1      :WAITMS
2      :
3      :WAIT FOR UDA TO RESPOND WITH A MESSAGE PACKET
4      :
5      :INPUTS:
6      :      R5 - ADDRESS OF CONTROLLER TABLE
7      :OUTPUTS:
8      :      Z CLEAR IF NO ERROR
9      :      Z SET IF ERROR, MESSAGE PRINTED
10     :
11     023374      WAITMS: PUSH <R0,R1>
12     023374      010046
13     023376      010146
14     023400      012700      000036
15     023404      016501
16     023406      062701      000040
17     023412      004737      023604
18     023416      011500
19     023420      032765      000010      000014      1$:
20     023426      001030
21     023430      016001      000002
22     023434      001034
23     023436      104422
24     023436      104422
25     023440      005737      002354
26     023444      001764
27     023446      023765      002366      000042
28     023454      101005
29     023456      001357
30     023460      023765      002364      000040
31     023466      103753
32     023470      104455
33     023472      000044
34     023474      000000
35     023476      014014
36     023500      012601
37     023502      012600
38     023504      000264
39     023506      000207
    
```

```

:WAITMS
:
:WAIT FOR UDA TO RESPOND WITH A MESSAGE PACKET
:
:INPUTS:
:      R5 - ADDRESS OF CONTROLLER TABLE
:OUTPUTS:
:      Z CLEAR IF NO ERROR
:      Z SET IF ERROR, MESSAGE PRINTED
:
WAITMS: PUSH <R0,R1>
                                MOV R0,-(SP)
                                MOV R1,-(SP)
MOV #30,,R0                      ;SET TIME OUT VALUE OF 30 SECONDS
MOV R5,R1                        ;POINT TO TIME OUT COUNTER
ADD #C.TO,R1
CALL SETTO
MOV (R5),R0                      ;GET ADDRESS OF UDAIP REGISTER
BIT #CT.MSG,C.FLG(R5)           ;LOOK IF INTERRUPT OCCURRED
BNE 3$                          ;BRANCH IF SO
MOV 2(R0),R1                    ;LOOK AT UDASA REGISTER
BNE 4$                          ;BRANCH IF ERROR CODE PRESENT
BREAK
                                TRAP      CSBRK
TST KW.CSR                      ;SEE IF A CLOCK ON SYSTEM
BEQ 1$
CMP KW.EL+2,C.TO(R5)           ;CHECK IF TIMEOUT HAS HAPPENED
BHI 2$
BNE 1$
CMP KW.EL,C.TO(R5)
BLO 1$
ERRDF 36,,ERR036
                                TRAP      CSERDF
                                .WORD    36
                                .WORD    0
                                .WORD    ERR036
POP <R1,R0>
SEZ
RETURN
MOV (SP)+,R1
MOV (SP)+,R0
    
```


1	023510	042765	000010	000014	3\$:	BIC #CT.MSG,C.FLG(R5)		:CLEAR MESSAGE RECEIVED FLAG	
2	023516					POP <R1,R0>			
	023516	012601							MOV (SP)+,R1
	023520	012600							MOV (SP)+,R0
3	023522	000244				CLZ		:GIVE NO ERROR RETURN	
4	023524	000207				RETURN			
5	023526				4\$:	ERRDF 37,,ERR037			
	023526	104455							TRAP C\$ERDF
	023530	000045							.WORD 37
	023532	000000							.WORD 0
	023534	014026							.WORD ERR037
6	023536					POP <R1,R0>			
	023536	012601							MOV (SP)+,R1
	023540	012600							MOV (SP)+,R0
7	023542	000264				SEZ			
8	023544	000207				RETURN			

```
1      :APRINT
2      :
3      :CONVERT AN 18 BIT ADDRESS STORED IN TWO WORDS INTO A FORMAT
4      :THAT WILL ALLOW PRINTING OF THE 18 BIT NUMBER.
5      :
6      :INPUTS:
7      :      R0 - ADDRESS OF TWO WORD BLOCK CONTAINING ADDRESS.
8      :              FIRST WORD CONTAINING LOW 16 BITS.
9      :              SECOND WORD CONTAINING HIGH 2 BITS.
10     :
11     :OUTPUTS:
12     :      R1 - HIGH 3 BITS OF ADDRESS
13     :      R2 - LOW 15 BITS OF ADDRESS
14 023546 016001 000002      APRINT: MOV 2(R0),R1      :GET HIGH 2 BITS
15 023552 006301              ASL R1              :SHIFT LEFT
16 023554 011002              MOV (R0),R2          :GET LOW 16 BITS
17 023556 100001              BPL APRIZ           :IF 16TH BIT SET
18 023560 005201              INC R1              :PLACE IT IN WITH HIGH 2 BITS
19 023562 000207      APRIZ: RETURN
```



```

1      :UDASRV
2
3      :UDA INTERRUPT SERVICE ROUTINE. MARKS UDA CONTROLLER TABLE THAT AN
4      :INTERRUPT HAS BEEN RECEIVED.
5
6      :THIS ROUTINE IS CALLED BY A [JSR R0,UDASRV] INSTRUCTION FROM WITHIN
7      :THE CONTROLLER TABLE. THE PC STORED IN R0 IS THE ADDRESS OF THE C.FLG
8      :WORD IN THE CONTROLLER TABLE. THE STACK CONTAINS THE SAVED CONTENTS
9      :OF R0 FOLLOWED BY THE INTERRUPTED PC AND PS.
10
11     :INPUTS:
12     :   R0 - ADDRESS OF C.FLG WORD IN CONTROLLER TABLE
13     :   STACK - SAVED CONTENTS OF R0
14     :OUTPUTS:
15     :   CT.CMD CLEARED AND CT.MSG SET IN C.FLG WORD OF CONTROLLER TABLE
16     :   R0 - RESTORED FROM STACK
17
18     BGNSRV UDASRV
19     023574 052710 000010      BIS #CT.MSG,(R0)      :SET CT.MSG      UDASRV::
20     023600 012600              POP R0              :RESTORE R0
21     023602                      ENDsrv                      MOV (SP)+,R0
22     023602 000002                      L10037:
23                                     RTI
    
```

```

1      ;SETTO
2
3      ;SET TIMEOUT COUNTER TO SOME NUMBER OF SECONDS FROM CURRENT TIME.
4
5      ;INPUTS:
6      ;   R0 - NUMBER OF SECONDS FOR TIMEOUT
7      ;   R1 - ADDRESS WHERE TWO WORD TIME TO BE PUT
8
9      ;OUTPUTS:
10     ;   R0 - CONTENTS DESTROYED
11     ;   R1 - INCREMENTED BY 2
12
13
14
15     ;COMPUTE CLOCK TICKS TIL TIMEOUT
16
17
18
19
20
21
22
23
24
25
26
27 023604      ;SETTO: PUSH <R2,R3>
    023604 010246
    023606 010346
    29 023610 005002
    30 023612 013703 002362
    45 023616 006200
    46 023620 103001
    47 023622 060302
    48 023624 006303
    49 023626 005700
    50 023630 001372
    52
    53
    54 023632 013700 002364
    55 023636 013703 002366
    56 023642 020037 002364
    57 023646 001371
    58
    59
    60
    61 023650 060200
    62 023652 005503
    66
    67
    68
    69 023654 010021
    70 023656 010311
    71
    75 023660
    023660 012603
    023662 012602
    77 023664 000207

        CLR R2
        MOV KW.HZ,R3
        ASR R0
        BCC SET01
        ADD R3,R2
        ASL R3
        TST R0
        BNE SET00
        MOV KW.EL,R0
        MOV KW.EL+2,R3
        CMP R0,KW.EL
        BNE SET02
        ADD R2,R0
        ADC R3
        MOV R0,(R1)+
        MOV R3,(R1)
        POP <R3,R2>
        RETURN

        ;CLEAR PRODUCT
        ;GET MULTIPLICAND
        ;SHIFT MULTIPLIER TO RIGHT
        ;IF A ONE BIT SHIFTED OUT
        ; ADD MULTIPLICAND TO PRODUCT
        ;DOUBLE THE MULTIPLICAND
        ;CONTINUE UNTIL MULTIPLIER IS ZERO

        ;GET TIME
        ;IF CHANGED DURING RETRIEVAL
        ; GET IT AGAIN

        ;ADD
        MOV R2,-(SP)
        MOV R3,-(SP)
        MOV (SP)+,R3
        MOV (SP)+,R2
    
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40

```

:UDAIN
:FUNCTIONAL DESCRIPTION:
:   SUBROUTINE TO INITIALIZE A UDA AND BRING IT ON-LINE.
:   ALL STEPS ARE CHECKED. AN ERROR MESSAGE IS REPORTED IF ANY ERROR
:   DETECTED.
:INPUTS:
:   R5 - ADDRESS OF CONTROLLER TABLE.
:   R4 - LEN, INTI AND VECTOR FIELDS TO SEND TO UDA
:IMPLICIT INPUTS:
:   FFREE - FIRST FREE ADDRESS OF MEMORY. THIS ADDRESS IS GIVEN TO UDA
:           AS START OF RING BUFFER.
:   FSIZE - SIZE OF FREE MEMORY AVAILABLE IN WORDS.
:OUTPUTS:
:   CONDITION Z - SET IF ANY ERROR REPORTED. CLEAR IF NO ERROR.
:   R1 - SIZE OF RING BUFFER IN WORDS IF NO ERROR.
:   R4 - ADDRESS OF UDAIP REGISTER IN UDA
:   R5 - UNCHANGED.
:CHECK IF ENOUGH FREE MEMORY FOR RING BUFFER
UDAIN: MOV R4,R0           ;GET MESSAGE LENGTH
      SWAB R0
      BIC #177770,R0
      CALL CLOG           ;COMPUTE LOGARITHMIC VALUE
      MOV R1,R2          ;SAVE RESULT IN R2
      MOV R4,R0          ;GET COMMAND LENGTH
      SWAB R0
      ROR R0
      ROR R0
      ROR R0
      BIC #177770,R0
      CALL CLOG           ;COMPUTE LOGARITHMIC VALUE
      ADD R2,R1          ;ADD THE TWO RESULTS
      ASL R1             ;MULTIPLY BY 2 WORDS PER RING
      ADD #HC.ISZ/2,R1  ;ADD SPACE FOR INTERRUPT INDICATORS
      CMP R1,FSIZE      ;COMPARE WITH SIZE OF FREE MEMORY
      BLOS UDAI1
      JMP FMERR          ;FATAL ERROR IF NOT ENOUGH MEMORY
    
```

023666	010400	
023670	000300	
023672	042700	177770
023676	004737	024606
023702	010102	
023704	010400	
023706	000300	
023710	006000	
023712	006000	
023714	006000	
023716	042700	177770
023722	004737	024606
023726	060201	
023730	006301	
023732	062701	000002
023736	020137	002200
023742	101402	
023744	000137	014406


```

1          ;FILL HOST COMMUNICATION AREA WITH ALL ONES
2
3 023750 013702 002176   UDAI1:  MOV FFREE,R2          ;GET FIRST ADDRESS OF RING BUFFER
4 023754 010103          MOV R1,R3          ;GET SIZE OF RING BUFFER
5 023756 012722 177777   UDAI1L: MOV #-1,(R2)+  ;WRITE ONES TO BUFFER
6 023762 005303          DEC R3            ;COUNT THE WORDS IN BUFFER
7 023764 003374          BGT UDAI1L        ;LOOP UNTIL ENTIRE BUFFER WRITTEN
8
9          ;DO THE INITIALIZATION
10
11 023766 004737 024136   CALL UDAIST        ;DO FIRST THREE STEPS
12 023772 103457          BCS UDAIEX        ;GET OUT IF UDA MICROCODE REPORTED FAILURE
13 023774 012364 000002   MOV (R3)+,2(R4)   ;WRITE NEXT WORD TO UDASA REGISTER
14 024000 012700 000310   MOV #200.,R0     ;GET TRY COUNTER
15 024004 016402 000002   UDAI1A: MOV 2(R4),R2 ;LOOK AT UDASA
16 024010 001410          BEQ UDAI1C
17 024012 100005          BPL UDAI1B
18 024014          ERRDF 24,,ERR024
19 024014 104455          TRAP      CSERDF
20 024016 000030          .WORD    24
21 024020 000000          .WORD    0
22 024022 013600          .WORD   ERR024
23 024024 000442
24 024026 005300   UDAI1B: BR UDAIEX
25 024030 001365   UDAI1B: DEC R0
26 024032 010264 000002   BNE UDAI1A
27 024036 011402   UDAI1C: MOV R2,2(R4) ;WRITE 0 TO UDASA (PURGE)
28 024040 004737 024446   MOV (R4),R2     ;READ FROM UDAIP (POLL)
29 024044 103432   CALL UDARSP    ;WAIT FOR STEP OR ERROR BIT
30 024046          BCS UDAIEX ;GET OUT IF UDA MICROCODE REPORTED FAILURE
31 024046 010146   PUSH R1
32 024050 004733   CALL @ (R3)+    ; CALL LAST ROUTINE
33 024052          POP R1
34 024052 012601   MOV (SP)+,R1
35
36          ;CHECK HOST COMMUNICATION AREA FOR ALL ZEROS
37
38 024054 013702 002176   UDAI2:  MOV FFREE,R2          ;GET FIRST ADDRESS OF RING BUFFER
39 024060 010103          MOV R1,R3          ;GET SIZE OF RING BUFFER
40 024062 005722   UDAI2L: TST (R2)+          ;CHECK WORD IN BUFFER
41 024064 001003          BNE UDAI2E        ;GO TO ERROR REPORTER IF NOT ZERO
42 024066 005303          DEC R3            ;COUNT THE WORDS IN BUFFER
43 024070 003374          BGT UDAI2L        ;LOOP UNTIL ALL WORDS CHECKED
44 024072 000405          BR UDAI3
45
46 024074          UDAI2E: ERRDF 23,,ERR023 ;REPORT BUFFER NOT CLEARED
47 024074 104455          TRAP      CSERDF
48 024076 000027          .WORD    23
49 024100 000000          .WORD    0
50 024102 013514          .WORD   ERR023
51 024104 000412          BR UDAIEX
    
```

```
1  
2  
3 024106  
12 024106 016500 000006  
13 024112 006300  
14 024114 006300  
15 024116 052700 000001  
16 024122 010064 000002  
17 024126 000244  
18 024130 000207  
19  
20  
21  
22 024132 000264  
23 024134 000207
```

;
:SEND GO BIT TO UDASA REGISTER TO END INITIALIZATION

UDAI3:
MOV C,BST(R5),R0 ;GET BURST VALUE
ASL R0 ;SHIFT TO POSITION
ASL R0
BIS #SA.GO,R0 ;SET THE GO BIT
MOV R0,2(R4) ;SEND TO UDA
CLZ ;CLEAR Z AS NO ERROR INDICATION
RETURN

;
:ERROR RETURN

UDAIEX: SEZ ;SET Z TO INDICATE ERROR OCCURRED
RETURN


```

1      :UDAIST
2      :
3      :START THE INITIALIZATION PROCESS ON THE SELECTED UDA.
4      :STOP BEFORE WRITING THE THIRD WORD SO UDA DOES NOT
5      :ATTEMPT ANY UNIBUS TRANSFERS.
6      :
7      :INPUTS:
8      :
9      :   R5 - ADDRESS OF CONTROLLER TABLE
10     :   R4 - LEN, INTI AND VECTOR FIELDS TO SEND TO UDA
11     :
12     :LOAD TABLE OF DATA TO SEND TO UDASA REGISTER
13     UDAIST: BREAK
14     024136 104422                                TRAP    C$BRK
15     024140                                PUSH R1
16     024140 010146                                MOV R1,-(SP)
17     024142 052704 100000                        BIS #SA.STP,R4          ;SET STEP BIT IN DATA WORD
18     024146 010437 024340                        MOV R4,UDAID1          ;LOAD LENGTH AND INTERRUPT VECTOR
19     024152 013737 002176 024344                MOV FFREE,UDAID2      ;LOAD MEMORY ADDRESS
20     024160 062737 000004 024344                ADD #HC.MSG,UDAID2    ; OF FIRST RESPONSE RING
21     :START THE INITIALIZATION BY WRITING TO UDAIP REGISTER
22     024166 016504 000000                        MOV C.UADR(R5),R4     ;GET ADDRESS OF UDAIP REGISTER
23     024172 005037 002374                        CLR NXMAD             ;CLEAR MEMORY ERROR FLAG
24     024176                                SETVEC #4,#NXMI,#PRI07 ;SET UP VECTOR 4
25     024176 012746 000340                                MOV    #PRI07,-(SP)
26     024202 012746 023564                                MOV    #NXMI,-(SP)
27     024206 012746 000004                                MOV    #4,-(SP)
28     024212 012746 000003                                MOV    #3,-(SP)
29     024216 104437                                TRAP   C$SVEC
30     024220 062706 000010                                ADD    #10,SP
31     024224 005764 000002                        TST 2(R4)             ;ACCESS UDASA REGISTER
32     024230 005014                        CLR (R4)              ;WRITE TO UDAIP
33     024232                                CLRVEC #4             ;GIVE UP THE VECTOR
34     024232 012700 000004                                MOV    #4,R0
35     024236 104436                                TRAP   C$CVEC
36     024240 005737 002374                        TST NXMAD             ;SEE IF A MEMORY ERROR OCCURRED
37     024244 001406                        BEQ UDAISG
38     024246                                ERRDF 20,,ERR020
39     024246 104455                                TRAP   C$ERDF
40     024250 000024                                .WORD 20
41     024252 000000                                .WORD 0
42     024254 013402                                .WORD ERRO20
43     024256 000261                                SEC
44     024260 000424                                BR UDAISE
    
```



```
1 ;SET UP LOOP PARAMETERS TO EXECUTE THE FOUR STEPS OF INITIALIZATION
2
3 024262 012737 004000 024604 UDAISG: MOV #SA.S1,UDARSD ;STORE RESPONSE MASK
4 024270 012703 024336 MOV #UDAIDT,R3 ;AND INDEX TO TABLE
5
6 ;WAIT FOR AND CHECK RESPONSE DATA
7
8 024274 004737 024446 UDAISL: CALL UDARSP ;WAIT FOR STEP OR ERROR BITS
9 024300 103414 BCS UDAISE ;EXIT IF ERROR
10 024302 004733 CALL @(R3)+ ;CALL RESPONSE CHECKER FOR STEP
11 024304 103412 BCS UDAISE ;GET OUT IF ERROR
12 024306 006337 024604 ASL UDARSD ;SHIFT TO NEXT STEP BIT
13 024312 032737 040000 024604 BIT #SA.S4,UDARSD ;CHECK IF NOW AT STEP 4
14 024320 001003 BNE UDAISX ;GET OUT IF SO
15 024322 012364 000002 MOV (R3)+,2(R4) ;WRITE DATA TO UDASA REGISTER
16 024326 000762 BR UDAISL ;STAY IN LOOP
17
18 024330 000241 UDAISX: CLC ;CLEAR CARRY FOR NO ERROR INDICATION
19 024332 UDAISE: POP R1
20 024334 012601 000207 MOV (SP)+,R1
RETURN
```

```

1          ;DATA TO BE SENT AND RECEIVED BY UDA INITIALIZATION
2
3 024336 024354      UDAIDT: .WORD UDAIR1          ;FIRST WORD RESPONSE CHECK ROUTINE
4 024340 000000      UDAID1: .WORD 0              ;FIRST WORD TO SEND TO UDASA
5 024342 024362      UDAID2: .WORD UDAIR2          ;SECOND WORD RESPONSE CHECK ROUTINE
6 024344 000000      UDAID2: .WORD 0              ;SECOND WORD TO SEND TO UDASA
7 024346 024402      UDAID3: .WORD UDAIR3          ;THIRD WORD RESPONSE CHECK ROUTINE
8 024350 100000      UDAID3: .WORD SA.TST         ;THIRD WORD TO SEND TO UDASA
9 024352 024420      UDAID3: .WORD UDAIR4          ;FOURTH WORD RESPONSE CHECK ROUTINE
10
11          ;RESPONSE CHECK FOR FIRST WORD FROM UDASA
12          ;CHECK FOR PROPER CONTROLLER TYPE
13
14 024354 012701 004400  UDAIR1: MOV #SA.S1+SA.DI,R1      ;SET STEP ONE BIT
15 024360 000422          BR UDAIRC                      ;NOW COMPARE
16
17          ;RESPONSE CHECK FOR SECOND WORD FROM UDASA
18          ;CHECK FOR ECHO OF INTI AND VECTOR
19
20 024362 013701 024340  UDAIR2: MOV UDAID1,R1          ;GET WORD SENT TO UDASA
21 024366 000301          SWAB R1                       ;GET HIGH 8 BITS
22 024370 042701 177400          BIC #177400,R1
23 024374 052701 010000          BIS #SA.S2,R1          ;SET STEP 2 BIT
24 024400 000412          BR UDAIRC                      ;NOW COMPARE
25
26          ;RESPONSE CHECK FOR THIRD WORD FROM UDASA
27          ;CHECK FOR ECHO OF MESSAGE AND COMMAND RING LENGTHS
28
29 024402 013701 024340  UDAIR3: MOV UDAID1,R1          ;GET WORD SENT TO UDASA
30 024406 042701 177400          BIC #177400,R1          ;JUST LOW 8 BITS
31 024412 052701 020000          BIS #SA.S3,R1          ;SET STEP 3 BIT
32 024416 000403          BR UDAIRC                      ;NOW COMPARE
33
34          ;RESPONSE CHECK FOR FOURTH WORD FROM UDASA
35          ;CHECK FOR ECHO OF PURGE AND LFAIL BITS
36
37 024420 010201          UDAIR4: MOV R2,R1              ;GET RESPONSE FROM UDA
38 024422 042701 137400          BIC #^C<SA.S4+SA.MCV>,R1 ;KEEP MICROCODE VERSION AND STEP 4
39
40          ;COMPARE EXPECTED DATA IN R1 WITH ACTUAL DATA IN R2
41
42 024426 020102      UDAIRC: CMP R1,R2                ;COMPARE THE DATA
43 024430 001405      BEQ UDAIRX                       ;EXIT IF COMPARED CORRECTLY
44 024432          ERRDF 25,,ERR025                    ;REPORT ERROR
45 024432 104455      TRAP CSERDF
46 024434 000031      .WORD 25
47 024436 000000      .WORD 0
48 024440 013614      .WORD ERR025
49 024442 000261
50 024444 000207
51          SEC
52          UDAIRX: RETURN
    
```



```

1      ;UDARSP
2      ;
3      ;WAIT FOR UDA TO RESPOND WITH DATA IN UDASA REGISTER.
4      ;EITHER STEP BIT FROM MASK IN LOCATION UDARSD OR ERROR BIT
5      ;WILL CAUSE A TERMINATION.
6      ;AN ERROR MESSAGE WILL BE PRINTED IF THE UDA DOES NOT RESPOND
7      ;IN 10 SECONDS OR IF ERROR SETS.
8      ;
9      ;INPUTS:
10     ;       UDASRD - MASK OF STEP BIT TO LOOK FOR
11     ;       R5 - ADDRESS OF CONTROLLER TABLE
12     ;       R4 - ADDRESS OF UDAIP REGISTER
13     ;OUTPUTS:
14     ;       ERROR MESSAGE IF TIME OUT ON RESPONSE OR ERROR BIT SETS
15     ;       R2 - DATA FROM UDASA REGISTER
16     ;       CARRY SET IF ERROR BIT SETS OR TIME OUT
17
18 024446 UDARSP: PUSH R1
19 024446 010146
20 024450 052737 100000 024604      BIS #SA.ERR,UDARSD      ;SET ERROR BIT IN MASK WORD      MOV R1,-(SP)
21 024462 010501 000012      MOV #10.,R0      ;SET UP FOR 10 SECOND TIMEOUT
22 024464 062701 000040      MOV R5,R1      ;POINT TO COUNTER IN CONTROLLER TABLE
23 024470 004737 023604      ADD #C.TO,R1
24 024474      CALL SETTO
25 024474 012601
26 024476 033764 024604 000002 UDARS1: BIT UDARSD,2(R4)      ;LOOK AT ERROR AND STEP BIT      MOV (SP)+,R1
27 024504 001024      BNE UDARS2      ;BRANCH IF EITHER SET
28 024506 104422      BREAK
29 024510 005737 002354      TST KW.CSR      TRAP CSBRK
30 024514 001770      BEQ UDARS1      ;SEE IF CLOCK ON SYSTEM
31 024516 023765 002366 000042      CMP KW.EL+2,C.TO(R5)      ;CHECK IF TIME OUT OCCURRED
32 024524 101005      BHI 1$
33 024526 001363      BNE UDARS1
34 024530 023765 002364 000040      CMP KW.EL,C.TO(R5)
35 024540 016402 000002      BLO UDARS1
36 024544      1$: MOV 2(R4),R2      ;GET REGISTER CONTENTS
37 024544 104455      ERRDF 22,,ERR022      ;REPORT TIME OUT ERROR
38 024546 000026      TRAP CSERDF
39 024550 000000      .WORD 22
40 024552 013446      .WORD 0
41 024554 000407      .WORD ERR022
42      BR UDARSE
    
```



```
1  
2  
3 024556 016402 000002  
4 024562 100006  
5 024564  
   024564 104455  
   024566 000025  
   024570 000000  
   024572 013414  
6 024574 000261  
7 024576 000207  
8  
9  
10  
11 024600 000241  
12 024602 000207  
13  
14  
15  
16 024604 000000
```

;
:CHECK IF ERROR BIT SET
UDARS2: MOV 2(R4),R2
 BPL UDARSX
 ERRDF 21,,ERR021
;
UDARSE: SEC
 RETURN
;
:NORMAL EXIT
UDARSX: CLC
 RETURN
;
:LOCATION FOR STEP BIT MASK
UDARSD: .WORD 0

;
:GET REGISTER CONTENTS
:EXIT IF ERROR NOT SET
:REPORT ERROR INFO
TRAP CSERDF
.WORD 21
.WORD 0
.WORD ERR021
;
:CLEAR CARRY AS NO ERROR INDICATION
;
:LOAD BY CALLING ROUTINE

1
2
3
4
5
6
7
8
9
10 024606 010046
11 024606 005001
12 024610 000261
13 024614 006101
14 024616 005300
15 024620 100375
16 024622 012600
17 024624 000207

```
:CLOG  
:COMPUTE LOGARITHMIC VALUE OF NUMBER TO BASE 2.  
:INPUTS:  
:      R0 - LOGARITHM TO BE CONVERTED  
:OUTPUTS:  
:      R1 - VALUE OF 2 RAISED TO POWER OF INPUT NUMBER  
CLOG:  PUSH R0  
      CLR R1  
      SEC  
CLOGLP: ROL R1  
      DEC R0  
      BPL CLOGLP  
      POP R0  
      RETURN  
      MOV R0,-(SP)  
      :SET UP ZERO START VALUE  
      :WITH CARRY READY TO SHIFT IN  
      :SHIFT TO LEFT  
      : UNTIL R0  
      : GOES NEGATIVE  
      MOV (SP)+,R0
```

1
2
3
4
5
6
7
8
9
10
11 024626 012701 000005
12 024632 004737 024666
13 024636 006101
14 024640 004737 024650
15 024644 006001
16 024646 000207

:RDDLL
:READ DISK DRIVE DOWNLINE LOAD PROGRAM INTO MEMORY
:INPUTS: DLLNAM - NAME OF PROGRAM IN RAD50 (TWO WORDS)
:OUTPUTS: FREE MEMORY CONTAINING PROGRAM
: CARRY CLEAR IF NO ERROR, CARRY SET IF PROGRAM NOT FOUND
:TYPE OF PROGRAM IN DATA FILE
:READ PROGRAM INTO MEMORY
:PRESERVE CARRY STATE IN R1
: WHILE CLOSING THE DATA FILE
: AS NORMAL POSITION IS LOST

RDDLL: MOV #5,R1
CALL RDREC
ROL R1
CALL CLOSEF
ROR R1
RETURN


```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10 024650 005737 002260  
11 024654 001403  
12 024656 104435  
13 024660 005037 002260  
14 024664 000207  
:CLOSEF  
:CLOSE DATA FILE FOR DM PROGRAMS  
:INPUTS: FILEOPN - ZERO IF FILE NOT OPEN  
:OUTPUTS:  
: NONE  
CLOSEF: TST FILEOPN ;SEE IF FILE CURRENTLY OPEN  
BEQ 1$ ; IF SO, CLOSE IT  
CLOSE ;AND MARK AS SO TRAP CSCLOS  
1$: CLR FILEOPN  
RETURN
```

```

1      :RDREC
2
3      :READ A RECORD FROM THE INPUT FILE. PLACE DATA INTO FREE MEMORY.
4
5      :INPUTS:
6          R1 - FILE TYPE
7              1 - TEST 1 DM PROGRAM
8              2 - TEST 2 DM PROGRAM
9              3 - TEST 3 DM PROGRAM
10             4 - TEST 4 DM PROGRAM
11             5 - DRIVE DIAGNOSTIC DOWNLINE LOAD PROGRAM
12
13         DLLNAM - IF R1 CONTAINS 5, TWO WORDS AT THIS ADDRESS CONTAIN
14             NAME OF PROGRAM IN RAD50.
15
16     :OUTPUTS:
17         DATA FROM RECORD IN MEMORY
18         CARRY CLEAR IF NO ERROR, CARRY SET IF ERROR
19
20 RDREC:  PUSH <R0,R1,R2,R3,R4,R5>
21
22         MOV R0,-(SP)
23         MOV R1,-(SP)
24         MOV R2,-(SP)
25         MOV R3,-(SP)
26         MOV R4,-(SP)
27         MOV R5,-(SP)
28
29         TST FILOPN          ;SEE IF FILE ALREADY OPEN
30         BNE RDSTS          ;IF NOT, OPEN FILE NOW
31         OPEN #FNAME
32
33             MOV #FNAME,R0
34             TRAP CSOPEN
35
36         ;AND MARK AS OPEN
37         ;CLEAR LOAD ADDRESS (SEARCH MODE)
38         ;ALLOW PROGRAM TO BE INTERRUPTED
39             TRAP CSBRK
40
41         ;GETBYTE CALLS DON'T SEEM TO BREAK ON CONTROL-C!
42         ;GET A BYTE
43             TRAP CSGETB
44             MOV R0,R4
45
46         ;IF ZERO
47         ;KEEP READING
48         ;WHEN NOT ZERO
49         ;IT BETTER BE A ONE
50         ;AND THE NEXT BYTE
51             TRAP CSGETB
52
53         ADD R0,R4
54         TST R0
55         BEQ RDDAT
56
57         ; IF ZERO, PROCESS DATA
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
    
```


1	025040	004737	025224	RDDAT:	CALL FWORD	:READ BYTE COUNT	
2	025044	013703	002256		MOV FDATA,R3	:SAVE IN R3	
3	025050	004737	025224		CALL FWORD	:READ LOAD ADDRESS	
4	025054	162703	000006		SUB #6,R3	:SUBTRACT BYTES ALREADY READ FROM BYTE COUNT	
5	025060	001443			BEQ RWORDT	:IF RESULT IS ZERO, THIS IS A	
6						:TRANSFER BLOCK	
7	025062	005705			TST R5	:IF IN SEARCH MODE,	
8	025064	001425			BEQ 3\$: BYPASS TRANSFER ADDRESS COMPUTATION	
9	025066	013701	002256		MOV FDATA,R1	:GET LOAD ADDRESS	
10	025072	060301			ADD R3,R1	:ADD BYTES IN RECORD	
11	025074	162701	001000		SUB #DMFRST,R1		
12	025100	060501			ADD R5,R1	:COMPUTE STORAGE ADDRESS	
13	025102	032701	000001		BIT #1,R1	:CHECK IF ODD BYTE ADDRESS	
14	025106	001401			BEQ 1\$:IF SO,	
15	025110	005201			INC R1	: INCREMENT	
16	025112	163701	002176	1\$:	SUB FFREE,R1	:SEE IF MORE MEMORY NEEDS TO BE ALLOCATED	
17	025116	101403			BLOS 2\$		
18	025120	006001			ROR R1	:REDUCE TO WORDS	
19	025122	004737	014420		CALL ALOCM	:ALLOCATE THE MEMORY	
20	025126	013702	002256	2\$:	MOV FDATA,R2	:GET LOAD ADDRESS	
21	025132	162702	001000		SUB #DMFRST,R2		
22	025136	060502			ADD R5,R2		
23	025140			3\$:	GETBYTE R0	:GET DATA BYTE	
24	025142	104426					TRAP CSGETB
25	025144	005705			TST R5	:IF IN SEARCH MODE,	
26	025146	001401			BEQ 4\$: BYPASS DATA STORAGE	
27	025150	110022			MOVB R0,(R2)+	:STORE IN MEMORY	
28	025152	060004		4\$:	ADD R0,R4	:UPDATE CHECKSUM	
29	025154	005303			DEC R3	:COUNT THE BYTE	
30	025156	001371			BNE 3\$:GET THEM ALL	
31	025160	104426			GETBYTE R0	:GET CHECKSUM	
32	025162	060004					TRAP CSGETB
33	025164	105704			ADD R0,R4	:ADD	
34	025166	001657			TSTB R4	:IF CHECKSUM CORRECT,	
		000441			BEQ RDST	: THEN GO READ NEXT RECORD	
					BR RWRDE1	: ELSE REPORT ERROR	

K 3

```

1 025170          RWORDT: GETBYTE R0          ;READ CHECKSUM BYTE
   025170 104426          ;ADD TO COMPUTED CHECKSUM      TRAP   CSGETB
2 025172 060004          ADD R0,R4          ;CHECK LOW BYTE OF SUM
3 025174 105704          TSTB R4          ;BRANCH IF CHECKSUM ERROR
4 025176 001035          BNE RWRDE1        ;IF IN SEARCH MODE,
5 025200 005705          TST R5          ; KEEP ON SEARCHING
6 025202 001650          BEQ RDST
7 025204          POP <R5,R4,R3,R2,R1,R0>
   025204 012605          MOV (SP)+,R5
   025206 012604          MOV (SP)+,R4
   025210 012603          MOV (SP)+,R3
   025212 012602          MOV (SP)+,R2
   025214 012601          MOV (SP)+,R1
   025216 012600          MOV (SP)+,R0
8 025220 000241          CLC
9 025222 000207          RETURN
10
11 025224          FWORD: GETBYTE R0          ;READ A BYTE FROM FILE
   025224 104426          ;UPDATE CHECKSUM ERROR      TRAP   CSGETB
12 025226 060004          ADD R0,R4          ;START TO BUILD WORD
13 025230 110037 002256  MOVB R0,FDATA
14 025234          GETBYTE R0          ;READ ANOTHER BYTE FROM FILE
   025234 104426          ;UPDATE CHECKSUM      TRAP   CSGETB
15 025236 060004          MOV R0,FDATA+1
16 025240 110037 002257  RETURN
17 025244 000207          RETURN
    
```

```
1 025246 004737 024650      RDERR: CALL CLOSEF          ;CLOSE FILE AS POSITION IS LOST
2 025252                                POP <R5,R4,R3,R2,R1,R0>
   025252 012605                                MOV (SP)+,R5
   025254 012604                                MOV (SP)+,R4
   025256 012603                                MOV (SP)+,R3
   025260 012602                                MOV (SP)+,R2
   025262 012601                                MOV (SP)+,R1
   025264 012600                                MOV (SP)+,R0
3 025266 000261      SEC          ;ERROR RETURN, FILE NOT FOUND
4 025270 000207      RETURN
5
6 025272                                RWRDE1: ERRSF 5,,ERR005
   025272 104454
   025274 000005                                TRAP C$ERSF
   025276 000000                                .WORD 5
   025300 013326                                .WORD 0
7 025302                                DOCLN                                .WORD ERR005
   025302 104444                                TRAP C$DCLN
```



```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10 025334  
025334 104422  
11 025336  
025336 104433  
12 025340 005737 002354  
13 025344 001403  
14 025346 012777 000105 155000  
15 025354 000207  
:RESET  
:RESET ALL UNIBUS DEVICES THEN RESTART THE CLOCK IF IT IS PRESENT  
:INPUTS:  
: KW.CSR = ADDRESS OF CLOCK REGISTER. ZERO IF NO CLOCK PRESENT.  
: KWOUT. = DATA TO SEND TO CLOCK REGISTER TO RESTART IT.  
:OUTPUTS:  
: NONE  
RESET: BREAK  
BRESE  
:RESET ALL DEVICES TRAP CSBRK  
:SEE IF A CLOCK IS PRESENT TRAP CSRESET  
:START UP THE CLOCK  
TST KW.CSR  
BEQ RESETX  
MOV #KWOUT.,@KW.CSR  
RESETX: RETURN
```

```

1
2
3
4
5
6
7
8
9
10
11
12
13 025356 005737 002354      :RNTIME
14 025362 001465              :PRINT RUNTIME
15 025364                      :INPUTS:
    025364 010046              KW.EL - CONTAINS ELAPSED TIME
    025366 010346              KW.HZ - HERTZ OF CLOCK
    025370 010446              :OUTPUTS:
    025372 010546              IF CLOCK ON SYSTEM:
    025374 013703 002364      " RNTIME HH:MM:SS " PRINTED
    025400 013704 002366      :
    025404 013700 002362      IF NO CLOCK: ONE SPACE IS PRINTED
    025410 004737 021344
    025414 012700 000074
    025420 004737 021344
    025424 010546
    025426 004737 021344
    025432 010346
    025434 004137 022324
    025440 003646
    025442 000002
    025444 020527 000011
    025450 003004
    025452 112700 000060
    025456 004737 022114
28 025462                      1$: PNT RNTIM1,R5
    025462 010546
    025464 004137 022324
    025470 003671
    025472 000002
29 025474                      POP R5
    025474 012605
    025476 020527 000011
    025502 003004
    025504 112700 000060
    025510 004737 022114
33 025514                      2$: PNT RNTIM2,R5
    025514 010546
    025516 004137 022324
    025522 003677
    025524 000002
34 025526                      POP <R5,R4,R3,R0>
    025526 012605
    
```

```

:CHECK IF A CLOCK PRESENT
:BRANCH IF NOT
MOV R0,-(SP)
MOV R3,-(SP)
MOV R4,-(SP)
MOV R5,-(SP)
:GET ELAPSED TIME
MOV KW.EL,R3
MOV KW.EL+2,R4
:GET SPEED OF CLOCK
MOV KW.HZ,R0
:COMPUTE SECONDS OF ELAPSED TIME
CALL DIVIDE
:NOW DIVIDE BY 60
MOV #60,R0
: TO COMPUTE MINUTES
CALL DIVIDE
:SAVE REMAINDER AS SECONDS
PUSH R5
MOV R5,-(SP)
:DIVIDE BY 60 AGAIN
CALL DIVIDE
:PRINT HOURS
PNT RNTIM,R3
MOV R3,-(SP)
JSR R1,LPNT
.WORD RNTIM
.WORD PNT.CT
:IF MINUTES 9 OR LESS
CMP R5,#9.
BGT 1$
:PRINT A LEADING ZERO
PRINT #'0
MOVB #'0,R0
CALL CPNT
:NOW PRINT MINUTES
1$: PNT RNTIM1,R5
MOV R5,-(SP)
JSR R1,LPNT
.WORD RNTIM1
.WORD PNT.CT
:GET SECONDS
POP R5
MOV (SP)+,R5
:IF 9 OR LESS
CMP R5,#9.
BGT 2$
:PRINT A LEADING ZERO
PRINT #'0
MOVB #'0,R0
CALL CPNT
:NOW PRINT SECONDS
2$: PNT RNTIM2,R5
MOV R5,-(SP)
JSR R1,LPNT
.WORD RNTIM2
.WORD PNT.CT
:HOURS IN R3
POP <R5,R4,R3,R0>
MOV (SP)+,R5
    
```



```
025530 012604
025532 012603
025534 012600
35 025536 112700 000040      RNTIMX: PRINT '<#>'
025536 004737 022114      ;PRINT A SPACE
025542 000207
36 025546
37
38 025550      RETURN
      ENDMOD
      MOV (SP)+,R4
      MOV (SP)+,R3
      MOV (SP)+,R0
      MOVB #',R0
      CALL CPNT
```

```
1          .SBTTL REPORT CODING SECTION
2
3 025550          BGNMOD
4
5
6          :++
7          : THE REPORT CODING SECTION CONTAINS THE
8          : "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
9          :--
10         BGNRPT
11
12         LSRPT::
13         PUSH   <R0,R1,R2,R3,R4,R5>
14         MOV R0,-(SP)
15         MOV R1,-(SP)
16         MOV R2,-(SP)
17         MOV R3,-(SP)
18         MOV R4,-(SP)
19         MOV R5,-(SP)
20         PNTS RPTMSG,TNUM          :PRINT TEST NUMBER
21         MOV TNUM,-(SP)
22         JSR R1,LPNTS
23         .WORD RPTMSG
24         .WORD PNT.CT
25         CALL RNTIME          :GET RUNTIME PARAMETERS
26         PRINT #CR          :END THE LINE
27         MOV #CR,R0
28         CALL CPNT
29         MOV #STIME,R1          :AT 15 MINUTES FROM NOW
30         MOV #15.*60.,R0      :SET TIME FOR NEXT REPORT
31         CALL SETTO
```



```

1 025712          PUSH <R3,R4,R5,R1>
  025712 010346
  025714 010446
  025716 010546
  025720 010146
2 025722 012700 002262
3 025726 012701 000022
4 025732 112720 000040      1$:
5 025736 005301
6 025740 001374
7 025742 005010
8 025744 011605
9 025746 016501 000200
10 025752 016502 000202
11 025756 016503 000204
12 025762 005004
13 025764 004737 026156      2$:
14 025770 062705 000060
15 025774 110540
16 025776 010146
17 026000 050216
18 026002 050316
19 026004 050426
20 026006 001366
21 026010
  026010 012601
22 026012          PRINTS #RPTMSD,D.UNIT(R1),(R1),#TEMP,D.SEEK(R1),D.XFRR(R1),D.XFRW(R1)
  026012 016146 000164
  026016 016146 000166
  026022 016146 000174
  026026 012746 002262
  026032 011146
  026034 016146 000002
  026040 012746 026476
  026044 012746 000007
  026050 010600
  026052 104416
  026054 062706 000020
23 026060          ASSUME D.DRV EQ 0
24 026060          PRINTS #RPTMD2,D.HERR(R1),D.SERR(R1),D.ECCC(R1)
  026060 016146 000176
  026064 016146 000172
  026070 016146 000170
  026074 012746 026545
  026100 012746 000004
  026104 010600
  026106 104416
  026110 062706 000012
48 026114          POP <R5,R4,R3>
  026114 012605
  026116 012604
  026120 012603
    
```

```

MOV R3,-(SP)
MOV R4,-(SP)
MOV R5,-(SP)
MOV R1,-(SP)
    
```

```

:PLACE 18 SPACE CHARACTERS INTO
: TEMP STORAGE
    
```

```

:THEN A NULL CHARACTER
:GET DRIVE TABLE STORAGE ADDRESS
:GET SERIAL NUMBER
    
```

```

:DIVIDE BY 10
:CONVERT TO ASCII CHARACTER
:PUT DIGIT INTO TEMP STORAGE
    
```

```

:SEE IF QUOTIENT IS ZERO
    
```

```

:IF NOT, DIVIDE AGAIN
    
```

```

MOV (SP)+,R1
MOV D.XFRW(R1),-(SP)
MOV D.XFRR(R1),-(SP)
MOV D.SEEK(R1),-(SP)
MOV #TEMP,-(SP)
MOV (R1),-(SP)
MOV D.UNIT(R1),-(SP)
MOV #RPTMSD,-(SP)
MOV #7,-(SP)
MOV SP,R0
TRAP C$PNTS
ADD #20,SP
    
```

```

MOV D.ECCC(R1),-(SP)
MOV D.SERR(R1),-(SP)
MOV D.HERR(R1),-(SP)
MOV #RPTMD2,-(SP)
MOV #4,-(SP)
MOV SP,R0
TRAP C$PNTS
ADD #12,SP
    
```

```

MOV (SP)+,R5
MOV (SP)+,R4
MOV (SP)+,R3
    
```

```

1 026122 005303          RPTDTN: DEC R3          ;COUNT THE DRIVE TABLES
2 026124 003265          BGT RPTDT          ;REPEAT FOR ALL DRIVE TABLES
3 026126 062705 000046  RPTCTN: ADD #C.SIZE,R5 ;GO TO NEXT CONTROLLER TABLE
4 026132 005715          TST (R5)
9 026134 001251          BNE RPTCT
11 026136          RPTXX: POP <R5,R4,R3,R2,R1,R0>
    026136 012605          MOV (SP)+,R5
    026140 012604          MOV (SP)+,R4
    026142 012603          MOV (SP)+,R3
    026144 012602          MOV (SP)+,R2
    026146 012601          MOV (SP)+,R1
    026150 012600          MOV (SP)+,R0
12 026152          EXIT RPT
    026152 000167          .WORD JSJMP
    026154 000412          .WORD L10042-2-.
13
14 026156          DIV10: PUSH R0          ;DIVIDEND IS IN <R4,R3,R2,R1>
    026156 010046          MOV R0,-(SP)
15 026160 012700 000100  MOV #64.,R0          ;SET UP SHIFT COUNT
16 026164 005005          CLR R5          ;START WITH ZERO REMAINDER
17 026166 006301 1$: ASL R1
18 026170 006102          ROL R2          ;SHIFT LEFT INTO R5
19 026172 006103          ROL R3
20 026174 006104          ROL R4
21 026176 006105          ROL R5
22 026200 022705 000012  CMP #10.,R5          ;SILL DIVISOR GO INTO REMAINDER?
23 026204 101003          BHI 2$          ;ONLY SUBTRACT IF IT WILL
24 026206 162705 000012  SUB #10.,R5          ;SUBTRACT DIVISOR
25 026212 005201          INC R1          ;PUT A ONE INTO QUOTIENT
26 026214 005300 2$: DEC R0          ;COUNT THE SHIFTS
27 026216 001363          BNE 1$
28 026220          POP R0          ;RETURN WITH QUOTIENT IN
    026220 012600          MOV (SP)+,R0
29 026222 000207          RETURN          ; <R4,R3,R2,R1> AND REMAINDER IN R5
30
31 026224 116 042 124 RPTMSG: .ASCIZ\N'TEST 'D3' IN PROGRESS. '\
32 026260 116 042 125 RPTMSH: .ASCII\N'UNIT DRIVE SERIAL-NUMBER SEEKS MBYTES MBYTES HARD SOFT ECC'\N
33 026372 042 040 040 .ASCIZ '\
34 026476 045 123 062 RPTMSD: .ASCIZ\%S2%D2%S3%D3%S1%T%S1%D5%S2%D5%S3%D5%S2\
35 026545 045 104 065 RPTMD2: .ASCIZ\%D5%S2%D5%S1%D5%\
41 .EVEN
42
43 026570          ENDRPT
    026570          L10042: TRAP CSRPT
    026570 104425
    
```

.SBTTL PROTECTION TABLE

..++
.. THIS TABLE IS USED BY THE RUNTIME SERVICES
.. TO PROTECT THE LOAD MEDIA.
..--

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

026572
026572

BGNPROT

LSPROT::

026572 177777
026574 177777
026576 177777

-1
-1
-1

;OFFSET INTO P-TABLE FOR CSR ADDRESS
;OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
;OFFSET INTO P-TABLE FOR DRIVE NUMBER

ENDPROT

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35

.SBTTL INITIALIZE SECTION

:++
: THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
: AT THE BEGINNING OF EACH PASS.
:--

026600
026600

BGNINIT

LSINIT::

.REM &
IF HERE FROM CONTINUE COMMAND
THEN
SET ICONT BIT IN IFLAGS
ENDIF
IF HERE FROM RESTART COMMAND
THEN
SET IREST BIT IN IFLAGS
ENDIF
IF HERE FROM POWER FAIL RESTART
THEN
RESET ALL UNITS
PRINT STATISTICAL REPORT
ENDIF
IF HERE FROM START COMMAND
THEN
RESET ALL UNITS
ESTABLISH FREE MEMORY
CLEAR TNUM
SET ISTRT BIT IN IFLAGS, CLEAR OTHER BITS
INITIALIZE CLOCK
BUILD TABLES
ELSE
CHECK TABLES FOR ADDED OR DROPPED UNITS
.ENDF

&

1	026600				READEF #EF.CONTINUE		:HERE FROM CONTINUE COMMAND?	
	026600	012700	000036				MOV	#EF.CONTINUE,RO
	026604	104447					TRAP	CSREFG
2	026606				BNCOMPLETE INIT1		:JUMP IF NOT	
	026606	103006					BCC	INIT1
3	026610	042737	000020	002226	BIC #ISTRTH,IFLAGS		:SET CONTINUE BIT IN FLAG RECORD	
4	026616	052737	000002	002226	BIS #ICONT,IFLAGS		:LOOK AT EVENT FLAGS	
5	026624				INIT1: READEF #EF.RESTART			
	026624	012700	000037				MOV	#EF.RESTART,RO
	026630	104447					TRAP	CSREFG
6	026632				BNCOMPLETE INIT1A		:SET IREST BIT IN IFLAGS	
	026632	103003					BCC	INIT1A
7	026634	052737	000004	002226	BIS #IREST,IFLAGS		: IF HERE FROM RESTART COMMAND	
8	026642				INIT1A: READEF #EF.PWR		:HERE FROM POWER RESTART?	
	026642	012700	000034				MOV	#EF.PWR,RO
	026646	104447					TRAP	CSREFG
9	026650				BNCOMPLETE INIT2		:JUMP IF SET	
	026650	103003					BCC	INIT2
10	026652	004737	025334		CALL RESET		:RESET ALL UNITS	
11	026656				DORPT		:PRINT A STATISTICAL REPORT	
	026656	104424					TRAP	CSDRPT
12	026660				INIT2: READEF #EF.START		:HERE FROM START COMMAND?	
	026660	012700	000040				MOV	#EF.START,RO
	026664	104447					TRAP	CSREFG
13	026666				BCOMPLETE INIT3		:JUMP IF NOT	
	026666	103467					BCS	INIT3

```

1          ;SET NOT AVAILABLE BITS IN ALL CONTROLLER AND DRIVE TABLES
2
3 026670 013705 002206          MOV CTABS,R5          ;GET FIRST CONTROLLER TABLE ADDRESS
4
5 026674 052765 100000 000002 INITC1: BIS #CT.AVL,C.UNIT(R5)      ;SET BIT IN CONTROLLER TABLE
6 026702 010502          MOV R5,R2          ;GET POINTER TO DRIVE TABLES
7 026704 062702 000020          ADD #C.DR0,R2
8 026710 012703 000010          MOV #8,R3          ;GET COUNT OF DRIVE TABLES
9 026714 012200          INITC2: MOV (R2)+,R0      ;CHECK IF ANY MORE DRIVE TABLES
10 026716 001405          BEQ INITC3
11 026720 052760 100000 000002  BIS #DT.AVL,D.UNIT(R0)      ;SET BIT IN DRIVE TABLE
12 026726 005303          DEC R3
13 026730 003371          BGT INITC2
14 026732 062705 000046          INITC3: ADD #C.SIZE,R5      ;MOVE TO NEXT CONTROLLER TABLE
15 026736 005715          TST (R5)          ;IS THERE A NEXT ONE?
16 026740 001355          BNE INITC1        ;IF SO, CLEAR THE BITS THERE
17
18          ;NOW GET EACH P-TABLE AND CLEAR NOT AVAILABLE BITS
19
20 026742 005003          CLR R3          ;START WITH UNIT 0
21 026744          INITC4: GPHARD R3,R0      ;GET HW P-TABLE
22 026744 010300          MOV R3,R0
23 026746 104442          TRAP CS$GPHRD
24 026750          BNCOMPLETE INITC7      ;GO AROUND IF NOT AVAILABLE
25 026750 103030          BCC INITC7
26 026752 013705 002206          INITC5: MOV CTABS,R5      ;GET FIRST CONTROLLER TABLE
27 026756 021015          CMP (R0),(R5)    ;COMPARE UDA ADDRESSES
28 026760 001411          BEQ INITCC
29 026762 062705 000046          ADD #C.SIZE,R5      ;LOOK AT NEXT CONTROLLER TABLE
30 026766 005715          TST (R5)          ;IF THERE IS ANY
31 026770 001372          BNE INITC5
32 026772          INITE1: ERRSF 6,,ERR006
33 026772 104454          TRAP CS$ERSF
34 026774 000006          .WORD 6
35 026776 000000          .WORD 0
36 027000 013340          .WORD ERR006
37 027002          DOCLN          TRAP CS$DCLN
38 027002 104444
39 027004 016001 000010          INITCC: MOV HO.LDR(R0),R1
40 027010 004737 020734          CALL GTDRVT
41 027014 001366          BNE INITE1
42 027016 042765 100000 000002 INITC6: BIC #CT.AVL,C.UNIT(R5)      ;CLEAR BIT IN CONTROLLER TABLE
43 027024 042764 100000 000002  BIC #DT.AVL,D.UNIT(R4)      ;CLEAR BIT IN DRIVE TABLE
44 027032 005203          INITC7: INC R3      ;INCREMENT UNIT NUMBER
45 027034 023703 002012          CMP L$UNIT,R3      ;CHECK IF GOT ALL TABLES
46 027040 003341          BGT INITC4        ;IF NOT, GO GET ANOTHER
47 027042 000137 030040          JMP INITXX        ;EXIT THE INIT CODE

```



```

1 027046          INIT3: BRESET          ;RESET ALL UNITS
  027046 104433          MEMORY FFREE      ;RESET START OF FREE MEMORY
2 027050          TRAP CSRESET
  027050 104431          TRAP CSMEM
  027052 010037 002176          MOV RO,FFREE
3 027056 017737 153114 002200          ;RESET SIZE OF FREE MEMORY
4 027064 005037 002230          CLR TNUM      ;INITIALIZE TEST NUMBER TO NO TEST RUNNING
5 027070 012737 000010 002226          MOV #ISTR,IFLAGS ;SET START FLAG FOR TEST 4
6
7
8          ;INITIALIZE CLOCK
9
10 027076 000105          KWOUT.=105      ;DATA TO SEND TO KW11 TO START CLOCK
11 027102 005037 002364          CLR KW.EL      ;CLEAR ELAPSED TIME
12 027106 005037 002366          CLR KW.EL+2
13 027106 012700 000114          CLOCK L,RO      ;SEE IF AN L CLOCK PRESENT
14 027112 104462          MOV #L,RO
15 027114          TRAP CSCLCK
16 027114 103413          BCOMPLETE KYES
17 027116          BCS KYES
18 027116 103413          CLOCK P,RO      ;SEE IF A P CLOCK PRESENT
19 027116 012700 000120          MOV #P,RO
20 027122 104462          TRAP CSCLCK
21 027124          BCOMPLETE KYES
22 027124 103407          BCS KYES
23 027126 005037 002354          CLR KW.CSR      ;IF NEITHER, CLEAR CSR STORAGE WORD
24 027132 004137 022266          PNTF NOCLOCK
25 027132 004137 022266          JSR R1,LPNTF
26 027136 006055          .WORD NOCLOCK
27 027140 000000          .WORD PNT.CT
28 027142 000434          BR KNO
29 027144 012037 002354          KYES: MOV (RO)+,KW.CSR      ;STORE DATA RETURNED
30 027150 012037 002356          MOV (RO)+,KW.BRL
31 027154 012037 002360          MOV (RO)+,KW.VEC
32 027160 012037 002362          MOV (RO)+,KW.HZ
33 027164          SETVEC KW.VEC,#KW11I,KW.BRL ;SET THE VECTOR
34 027164 013746 002356          MOV KW.BRL,-(SP)
35 027170 012746 025304          MOV #KW11I,-(SP)
36 027174 013746 002360          MOV KW.VEC,-(SP)
37 027200 012746 000003          MOV #3,-(SP)
38 027204 104437          TRAP CSSVEC
39 027206 062706 000010          ADD #10,SP
40 027212 012777 000105 153134          MOV #KWOUT,@KW.CSR ;START THE CLOCK
41 027220 012701 002370          MOV #STIME,R1
42 027224 012700 001604          MOV #15.*60.,RO
43 027230 004737 023604          CALL SETTO ;SET TIME FOR NEXT REPORT
44 027234          KNO:
    
```

```

1
2
3 027234 013737 002176 002206      MOV FFREE,CTABS      ;STORE START OF CONTROLLER TABLES
4 027242 005077 152740              CLR @CTABS          ;ZEROS MARKS END CONTROLLER TABLES
5 027246 005037 002210              CLR CTRLRS         ;CLEAR CONTROLLER COUNT
6
7
8
9 027252 005002                      ;GET A P-TABLE FROM DRS
10 027254 005002                      INIT4: CLR R2        ;LOGICAL UNIT NUMBER IN R2
    027254 010200                      GPHARD R2,R0        ;GET POINTER TO A P-TABLE
    027256 104442                      ;
11 027260 005002                      BNCOMPLETE NXTTAB  ;IGNORE IF NO TABLE RETURNED
    027260 103075                      ;
12
13
14
15 027262 013703 002206              ;SEE IF A CONTROLLER TABLE ALREADY EXISTS FOR CONTROLLER IN P-TABLE
16 027266 005713                      INITS: MOV CTABS,R3 ;GET ADDRESS OF CONTROLLER TABLES
17 027270 001416                      TST (R3)           ;CHECK IF ANY MORE TABLES
18 027272 021013                      BEQ NEWTAB         ;BUILD NEW TABLE IF FOUND ZERO WORD
19 027274 001416                      CMP (R0),(R3)      ;CHECK IF SAME UNIBUS ADDRESS
20 027274 001416                      ;
    ;
    ASSUME C.UADR EQ 0
    ASSUME HO.UBA EQ 0
21 027274 001450                      BEQ SAMTAB        ;CHECK TABLE IF ALREADY EXISTS
22 027276 016301 000004              MOV C.VEC(R3),R1  ;GET VECTOR FROM EXISING CONTROLLER TABLE
23 027302 042701 177000              BIC #^C<CT.VEC>,R1
24 027306 026001 000002              CMP HO.VEC(R0),R1 ;SEE IF DEFFERENT VECTOR
25 027312 001002                      BNE 1$
26 027314 000137 030166              JMP SAMVEC
27 027320 062703 000046              1$: ADD #C.SIZE,R3  ;ERROR, CAN'T HAVE TWO UDA'S WITH SAME VECTOR
28 027324 000760                      BR INITS          ;MOVE TO NEXT TABLE
    
```



```
1  
2 ;SHOULD BE SAME CONTROLLER, CHECK THAT OTHER PARAMETERS MATCH  
3 027416 016004 000004 SAMTAB: MOV HO,BRL(R0),R4 ;GET BR LEVEL FROM P-TABLE  
4 027422 000304 SWAB R4 ;SWAP TO HIGH BYTE  
5 027424 006104 ROL R4 ;SHIFT ONE MORE TO LEFT  
6 027426 056004 000002 B: S HO.VEC(R0),R4 ;ADD VECTOR ADDRESS  
7 027432 020463 000004 CMP R4,C.VEC(R3) ;COMPARE WITH CONTROLLER TABLE  
8 027436 001004 BNE 1$  
9 027440 026063 000006 000006 CMP HO.BST(R0),C.BST(R3) ;COMPARE BURST RATES  
10 027446 001402 BEQ NXTTAB  
11 027450 000137 030116 1$: JMP CTABER ;FATAL ERROR IF NOT SAME  
12  
13 ;GET NEXT P-TABLE  
14  
15 027454 005202 NXTTAB: INC R2 ;INCREMENT LOGICAL UNIT NUMBER  
16 027456 023702 002012 CMP L$UNIT,R2 ;CHECK IF GOT ALL TABLES  
17 027462 003274 BGT INIT4 ;IF NOT, GO BACK FOR NEXT  
18  
19 027464 012701 000001 MOV #1,R1 ;ALLOCATE SPACE FOR ZERO END WORD  
20 027470 004737 014420 CALL ALO M ;AFTER CONTROLLER TABLES
```

```
1  
2  
3 027474 005005  
4 027476 005002  
5 027500  
 027500 010200  
 027502 104442  
6 027504  
 027504 103060  
7  
8  
9  
10 027506 013703 002206  
11 027512 021013  
12 027514 001403  
13 027516 062703 000046  
14 027522 000773  
  
:NOW BUILD DRIVE TABLES  
  
      CLR R5  
      CLR R2  
INIT8: GPHARD R2,R0  
  
      BNCOMPLETE INIT14  
  
:FIND CONTROLLER TABLE  
  
INIT10: MOV CTABS,R3  
        CMP (R0),(R3)  
        BEQ INIT11  
        ADD #C.SIZE,R3  
        BR INIT10  
  
;CLEAR CUSTOMER DATA FLAG  
;LOGICAL UNIT NUMBER IN R2  
;GET POINTER TO A P-TABLE  
      MOV R2,R0  
      TRAP CS$GPHRD  
;IF NOT AVAILABLE, GO GET NEXT  
      BCC INIT14  
  
;GET ADDRESS OF CONTROLLER TABLES  
;CHECK IF SAME UNIBUS ADDRESS  
;BRANCH IF TABLE FOUND  
;MOVE TO NEXT TABLE
```

1				:BUILD DRIVE TABLE	
2					
3	027524	012701	000103	INIT11: MOV #D.SIZE/2,R1	:GET SIZE OF DRIVE TABLE
4	027530	004737	014420	CALL ALOCM	:ALLOCATE SPACE FROM FREE MEMORY
5				: R0 POINTS TO P-TABLE	
6				: R1 POINTS TO DRIVE TABLE	
7				: R3 POINTS TO CONTROLLER TABLE	
8				: R2 IS UNIT NUMBER	
9	027534	010337	002262	MOV R3,TEMP	:SAVE CONTROLLER TABLE ADDRESS
10					:IN CASE AN ERROR IS DETECTED
11	027540	062703	000020	ADD #C.DRO,R3	:BUILD POINTER TO C.DR ENTRY IN CONTROLLER TABLE
12	027544	012704	000010	MOV #8,R4	:GET MAX COUNT OF DRIVES ON ONE CONTROLLER
13	027550	005713		INIT12: TST (R3)	:CHECK IF ENTRY CONTAINS POINTER TO DRIVE TABLE
14	027552	001411		BEQ INIT13	
15	027554	026033	000010	CMP HO.LDR(R0),@ (R3)+	:CHECK DRIVE NUMBER IN DRIVE TABLE
16	027560	001002		BNE 1\$	
17	027562	000137	030132	JMP MLDRER	:IF SAME, TWO P-TABLES POINT TO SAME DRIVE
18	027566	005304		1\$: DEC R4	:COUNT DRIVES
19	027570	001367		BNE INIT12	:IF EIGHT DRIVE TABLES EXIST,
20	027572	000137	030150	JMP TOOMER	: THEN REPORT ERROR
21	027576	010113		INIT13: MOV R1,(R3)	:LOAD DRIVE TABLE POINTER
22	027600	016021	000010	MOV HO.LDR(R0),(R1)+	:LOAD DRIVE NUMBER
23	027604	010221		MOV R2,(R1)+	:LOAD UNIT NUMBER
24	027606	016011	000012	MOV HO.PRM(R0),(R1)	:GET TEST AREA BIT
25	027612	051105		BIS (R1),R5	:SAVE 'OR' OF BIT FROM ALL DRIVES
26	027614	005111		COM (R1)	:COMPLIMENT IT
27	027616			AND HM.CYL,(R1)	
28	027622	042711	157777		BIC #^C<HM.CYL>,(R1)
29	027626	052721	011012	BIS #DDEF,(R1)+	:LOAD DEFAULT PARAMETER BITS
30	027632	012703	000100	MOV #<D.SIZE/2-3>,R3	:CLEAR REST OF TABLE
31	027634	005021		INIT3L: CLR (R1)+	
32	027636	005303		DEC R3	
33	027640	003375		BGT INIT3L	
33	027640	012761	177777 177754	MOV #-1,<D.ECYL+2-D.SIZE>(R1)	:MARK CYLINDERS AT TEST ALL


```

1      ;GO TO NEXT DRIVE TABLE
2
3 027646 005202
4 027650 023702 002012  INIT14: INC R2          ;INCREMENT LOGICAL UNIT NUMBER
5 027654 003311          CMP L$UNIT,R2       ;CHECK IF GOT ALL TABLES
6                                     BGT INIT8          ;IF NOT, GET NEXT TABLE
22
23 ;IF ANY DRIVE SELECTED FOR EXERCISE IN CUSTOMER DATA AREA
24 ;GIVE WARNING
25 027656 032705 020000          BIT #HM.CYL,R5       ;CHECK IF BIT EVER SET
26 027662 001460          BEQ INIT15        ;BYPASS IF NOT
27 027664          PNTF INITWA      ;PRINT WARNING HEADER
                                JSR R1,LPNTF
                                .WORD INITWA
                                .WORD PNT.CT
28 027674 013705 002206          MOV CTABS,R5        ;GET FIRST CONTROLLER TABLE
29 027700 010504          INITW1: MOV R5,R4      ;GET ADDRESS OF POINTER TO DRIVE TABLE
30 027702 062704 000020          ADD #C.DR0,R4
31 027706 012701 000010          MOV #8.,R1        ;GET COUNT OF DRIVE TABLES
32 027712 012403          INITW2: MOV (R4)+,R3  ;GET ADDRESS OF DRIVE TABLE
33 027714 001422          BEQ INITW4
34 027716 032763 020000 000004  BIT #D.DCY,D.PRM(R3) ;CHECK IF CUSTOMER DATA SELECTED
35 027724 001014          BNE INITW3
36 027726          PRINTF #INITWB,D.UNIT(R3),(R5),(R3) ;PRINT NUMBERS
                                MOV (R3),-(SP)
                                MOV (R5),-(SP)
                                MOV D.UNIT(R3),-(SP)
                                MOV #INITWB,-(SP)
                                MOV #4,-(SP)
                                MOV SP,R0
                                TRAP C$PNTF
                                ADD #12,SP
027726 011346
027730 011546
027732 016346 000002
027736 012746 004361
027742 012746 000004
027746 010600
027750 104417
027752 062706 000012
37 027756 005301          INITW3: DEC R1      ;COUNT THE DRIVE TABLES
38 027760 001354          BNE INITW2        ;LOOK AT ALL OF THEM
39 027762 062705 000046          INITW4: ADD #C.SIZE,R5 ;MOVE TO NEXT CONTROLLER TABLE
40 027766 005715          TST (R5)         ;SEE IF ANOTHER TABLE AND
41 027770 001343          BNE INITW1        ; LOOK AT IT
    
```

```

2
3      ;GET CONFIRMATION TO PROCEED
4 027772      MANUAL      ;CHECK IF MANUAL INTERVENTION ALLOWED
5 027772 104450      ;BRANCH IF ALLOWED      TRAP      CSMANI
6 027774 103013      ;ASK OPERATOR      BCC      INIT15
7 027776 104443      ;LOOK AT RESPONSE      TRAP      CS$GMAN
8 030000 000404      ;BRANCH IF YES WAS ANSWER      BR      10000$
9 030002 002262      ;ABORT PROGRAM      .WORD    TEMP
10 030004 000120      ;ERASE DOWNLINE LOAD DATA      .WORD    T$CODE
11 030006 003564      ;MAKE SURE DATA FILE IS CLOSED      .WORD    INITWC
12 030010 000001      ;SET RUNNING PRIORITY TO ZERO      .WORD    1
13 030012 032737 000001 002262      BIT #1,TEMP
14 030012 032737 000001 002262      ;SAVE CURRENT PARAMETERS TO FREE MEMORY SO EACH TEST CAN USE ALL OF IT
15 030020 001001      INIT15: MOV F$FREE,F$MEM      ;SAVE START ADDRESS
16 030022 104444      ;SAVE SIZE      MOV F$SIZE,F$MEMS
17 030024 013737 002176 002202      INITXX: SETPRI #PRI00      ; SET RUNNING PRIORITY TO ZERO
18 030032 013737 002200 002204      ;ERASE DOWNLINE LOAD DATA      MOV      #PRI00,R0
19 030040 012700 000000      ;MAKE SURE DATA FILE IS CLOSED      TRAP    C$SPRI
20 030044 104441      CLR DLL      ;ERASE DOWNLINE LOAD DATA
21 030046 005037 002414      ;MAKE SURE DATA FILE IS CLOSED      CALL CLOSEF
22 030052 004737 024650      ;MAKE SURE DATA FILE IS CLOSED      REDEF #EF.START
23 030056 012700 000040      ;ERASE DOWNLINE LOAD DATA      MOV      #EF.START,R0
24 030062 104447      ;MAKE SURE DATA FILE IS CLOSED      TRAP    C$REFG
25 030064 012700 000037      BCOMPLETE INITIM      ;ERASE DOWNLINE LOAD DATA      BCS     INITIM
26 030066 103404      ;MAKE SURE DATA FILE IS CLOSED      REDEF #EF.RESTART
27 030072 104447      ;ERASE DOWNLINE LOAD DATA      MOV      #EF.RESTART,R0
28 030074 103006      ;MAKE SURE DATA FILE IS CLOSED      TRAP    C$REFG
29 030076 012701 002370      BCOMPLETE KPRI      ;ERASE DOWNLINE LOAD DATA      BCC     KPRI
30 030102 012700 001604      INITIM: MOV #STIME,R1      ;AT 15 MINUTES FROM NOW
31 030106 004737 023604      ;MAKE SURE DATA FILE IS CLOSED      MOV #15.*60.,R0
32 030112 104432      ;MAKE SURE DATA FILE IS CLOSED      CALL SETTO
33 030114 000066      ;MAKE SURE DATA FILE IS CLOSED      EXIT INIT
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
    
```

```

1
2 030116 010305 ;DIFFERENT VECTORS, BR LEVELS OR BURST RATES FOR ONE CONTROLLER
3 030120 ;CTABER: MOV R3,R5 ;GET CONTROLLER ADDRESS
4 030120 104454 ERRSF 1,,ERR001
5 030122 000001 TRAP CSERSF
6 030124 000000 .WORD 1
7 030126 013242 .WORD 0
8 030130 DOCLN .WORD ERR001
9 030130 104444 TRAP CSDCLN
10
11 ;TWO P-TABLES FOR SAME DRIVE
12 MLDRE: MOV TEMP,R5 ;GET CONTROLLER ADDRESS
13 030132 013705 002262 ERRSF 2,,ERR002
14 030136 104454 TRAP CSERSF
15 030140 000002 .WORD 2
16 030142 000000 .WORD 0
17 030144 013260 .WORD ERR002
18 030146 DOCLN TRAP CSDCLN
19 030146 104444
20
21 ;MORE THAN EIGHT DRIVES SELECTED ON ONE CONTROLLER
22 TOOMER: MOV TEMP,R5 ;GET CONTROLLER ADDRESS
23 030150 013705 002262 ERRSF 3,,ERR003
24 030154 104454 TRAP CSERSF
25 030156 000003 .WORD 3
26 030160 000000 .WORD 0
27 030162 013276 .WORD ERR003
28 030164 DOCLN TRAP CSDCLN
29 030164 104444
30
31 ;TWO UDA'S USE THE SAME VECTOR
32 SAMVEC: MOV R3,R5 ;GET CONTROLLER ADDRESS
33 030166 010305 ERRSF 8,,ERR008
34 030170 104454 TRAP CSERSF
35 030172 000010 .WORD 8
36 030174 000000 .WORD 0
37 030176 013364 .WORD ERR008
38 030200 DOCLN TRAP CSDCLN
39 030200 104444
40
41 ENDINIT
42
43 L10044: TRAP CSINIT
    
```


1
2
3
4
5
6
7
8
9

.SBTTL AUTODROP SECTION

:++
: THIS CODE IS EXECUTED IMMEDIATELY AFTER THE INITIALIZE CODE IF
: THE "ADR" FLAG WAS SET. THE UNIT(S) UNDER TEST ARE CHECKED TO
: SEE IF THEY WILL RESPOND. THOSE THAT DON'T ARE IMMEDIATELY
: DROPPED FROM TESTING.
:--

10 030204
030204

BGNAUTO

LSAUTO::

11
12 030204
030204
030204

ENDAUTO

L10045: TRAP CSAUTO

104461

1
2
3
4
5
6
7
8
9
10
11
13
14
17
18
20
21
22
23

.SBTTL CLEANUP CODING SECTION

:++
: THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
: AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.
:--

```
8 030206          BGNCLN
   030206
9
10 030206 004737 024650          CALL CLOSEF          ;CLOSE DATA FILE
11
13 030212 022737 000004 002230  CMP    #4,TNUM          ; ARE WE DOING TEST #4?
14 030220 001402          BEQ    1$          ; IF SO, DON'T RESET BUS
17 030222 004737 025334          CALL    RESET
18 030226          1$:
20
21 030226          ENDCLN
   030226
   030226 104412          L10046: TRAP  CSCLEAN
22
23 030230          ENDMOD
```

```

1          .SBTTL TEST 1: UNIBUS ADDRESSING TEST
2
3 030230          BGNMOD
4
6 030230          BGNTST
7 030230 012701 000001          MOV #1,R1          ; INITIALIZE TEST PARAMETERS
8 030234 004737 014504          CALL TINIT
9 030240 013737 002206 002212  T1NEXT: MOV CTABS,TSTTAB ; GET ADDRESS OF FIRST CONTROLLER TABLE
10 030246 013705 002212          MOV TSTTAB,R5 ; GET CONTROLLER TABLE ADDRESS
11 030252 116537 000002 002074  MOVB C.UNIT(R5),LSLUN ; CHECK IF UNIT AVAILABLE FOR TESTING
12 030260 005765 000002          TST C.UNIT(R5)
13 030264 100010          BPL T1NOW          ; TEST IF AVAILABLE
14 030266
15 030266 062737 000046 002212  T1SKIP: ADD #C.SIZE,TSTTAB ; MOVE TO NEXT CONTROLLER
16 030274 005777 151712          TST @TSTTAB ; CHECK IF ANOTHER CONTROLLER TABLE
17 030300 001362          BNE T1NEXT
18 030302          EXIT TST
   030302 104432
   030304 000776          TRAP CSEXIT
   .WORD L10047-.
19
20 030306 004737 025334          T1NOW: CALL RESET          ; RESET ALL UNITS
    
```


1	030312		BGNSUB: 1				
	030312					T1.1:	
2	030312	104402				TRAP	C\$BSUB
3	030314	005037	002374	CLR NXMAD		:CLEAR MEMORY ERROR FLAG	
	030320			SETVEC #4,#NXMI,#PRI07			
	030320	012746	000340			MOV	#PRI07,-(SP)
	030324	012746	023564			MOV	#NXMI,-(SP)
	030330	012746	000004			MOV	#4,-(SP)
	030334	012746	000003			MOV	#3,-(SP)
	030340	104437				TRAP	C\$SVEC
	030342	062706	000010			ADD	#10,SP
4	030346	011504		MOV (R5),R4		:GET ADDRESS OF UDAIP REGISTER	
5	030350	005714		TST (R4)		:READ UDAIP	
6	030352	005764	000002	TST 2(R4)		:READ UDASA	
7	030356			CLRVEC #4		:GIVE UP VECTOR	
	030356	012700	000004			MOV	#4,R0
	030362	104436				TRAP	C\$CVEC
8	030364	005737	002374	TST NXMAD		:CHECK FLAG	
9	030370	001406		BEQ T1GOOD			
10	030372			ERRDF 38,,ERR038			
	030372	104455				TRAP	C\$ERDF
	030374	000046				.WORD	38
	030376	000000				.WORD	0
	030400	014042				.WORD	ERR038
11	030402			CKLOOP			
	030402	104406				TRAP	C\$CLP1
12	030404	000730		BR T1SKIP		:END TEST NOW	
13	030406			T1GOOD:			
14	030406			ENDSUB			
	030406					L10050:	
	030406	104403				TRAP	C\$ESUB

```
1 030410          BGNSUB; 2
  030410
  030410 104402          T1.2:          TRAP  C$BSUB
2 030412
3
4
5
6
7 030412 005014
8 030414 012737 004000 024604
9 030422 004737 024446
10 030426 103410
11 030430 012764 100000 000002
12 030436 012737 010000 024604
13 030444 004737 024446
14 030450
15 030450          1$:
    030450          ENDSUB
    030450          L10051:          TRAP  C$ESUB
    030450 104403
```

DIATST:

.....

MAKE SURE UDA PASSES INTERNAL DIAGNOSTIC
MAKE SURE UDA CAN SENSE STEP 1 AND 2

```
CLR      (R4)          ; INIT UDA
MOV      #SA.S1,UDARSD ; STEP 1 ASSERTED?
CALL     UDARSP        ; WAIT FOR RESPONSE
BCS     1$             ; IF FAIL, EXIT
MOV      #SA.STP,2(R4) ; SEND STEP 1
MOV      #SA.S2,UDARSD ; STEP 2 ASSERTED?
CALL     UDARSP
```

```

1 030452          BGNSUB; 3
  030452
  030452 104402
2 030454          PORTST:
3  :
4  :
5  :
6 030454 011504   MOV      (R5),R4          ; R4 POINTS TO UDAIP REGISTER
7 030456          ASSUME C.UADR EQ 0
8 030456 005014   CLR      (R4)           ; INITIALIZE THE UDA
9 030460 012737 004000 024604  MOV     #SA.S1,UDARSD    ; LOOK FOR STEP 1
10 030466 004737 024446      CALL    UDARSP          ; WAIT FOR RESPONSE
11 030472 103444      BCS     3$             ; IF ERROR, BRANCH
12 030474 016437 000002 031400  MOV     2(R4),WCHNGD     ; MOVE OLD PORT CONTENTS TO STORAGE
13 030502 012764 140000 000002  MOV     #<SA.STP+SA.WRP>,2(R4) ; INITIALIZE FOR PORT WRAP
14 030510 004737 031304      CALL    WCHNG          ; WAIT FOR THE PORT TO CHANGE
15 030514 001433      BEQ     3$             ; IF ERROR, BRANCH
16 030516 022764 140000 000002  CMP     #<SA.STP+SA.WRP>,2(R4) ; COMPARE WITH DATA WRITTEN
17 030524 001017      BNE     5$
18 030526 012702 000001      MOV     #1,R2          ; SET UP FOR SHIFTING '1'
19 030532 012703 000020      MOV     #16,R3         ; SET UP LOOP COUNT
20 030536 016437 000002 031400  MOV     2(R4),WCHNGD     ; SAVE OLD PORT CONTENTS
21 030544 010264 000002      MOV     R2,2(R4)       ; WRITE PATTERN TO UDASA FOR LOOP
22 030550 004737 031304      CALL    WCHNG          ; WAIT FOR UDASA TO CHANGE
23 030554 001413      BEQ     3$             ; IF ERROR, BRANCH
24 030556 020264 000002      CMP     R2,2(R4)       ; COMPARE R0 WITH WHAT WAS ECHOED
25 030562 001405      BEQ     2$             ; IF MATCH, BRANCH
26 030564          ERRDF  26,,ERR026 ; REPORT ERROR
    030564 104455          TRAP   CSERDF
    030566 000032          .WORD  26
    030570 000000          .WORD  0
    030572 013632          .WORD  ERR026
27 030574 000403      BR      3$            ; BRANCH
28 030576 006302      2$:  ASL     R2          ; MOVE THE SHIFTING ONE LEFT BY 1
29 030600 005303      DEC     R3            ; DECREMENT COUNT
30 030602 001355      BNE     1$            ; IF LOOP INCOMPLETE, BRANCH
31 030604          3$:
32 030604          ENDSUB
    030604          L10052:
    030604 104403          TRAP   C$ESUB
    
```



```

1 030606          BGNSUB; 4
  030606
  030606 104402
2 030610
3
4
5
6 030610 011504
7 030612
8 030612 016503 000004
9 030616 010302
10 030620 042703 177000
11 030624 042702 170777
12 030630 012701 000011
13 030634 006202
14 030636 005301
15 030640 001375
16 030642 010237 031402
17 030646
    030646 010346
    030650 011546
    030652 004137 022306
    030656 004143
    030660 000004
18 030662
19 030662
    030662 012746 000000
    030666 012746 025326
    030672 010346
    030674 012746 000003
    030700 104437
    030702 062706 000010
20 030706
    030706 012700 000000
    030712 104441
21 030714 006203
22 030716 006203
23 030720 052703 100200
24 030724 005037 002240
25 030730 005014
26 030732 012737 004000 024604
27 030740 004737 024446
28 030744 010364 000002
29 030750 012700 000012
30 030754 010501
31 030756 062701 000040
32 030762 004737 023604
33 030766 005737 002240
34 030772 001016
35 030774 104422
    030774

    T1.4: TRAP CSBSUB

    INTEST:
    :
    : TEST THE INTERRUPTS VECTOR AND BR LEVEL
    :
    : R4 POINTS TO UDAIP REGISTER
    : R4 POINTS TO UDAIP REGISTER
    : GET VECTOR AND BRANCH LEVEL
    : GET VECTOR AND BRANCH LEVEL
    : COPY TO R2 FOR BR LEVEL
    : COPY TO R2 FOR BR LEVEL
    : CLEAR UNUSED VECTOR BITS
    : CLEAR UNUSED VECTOR BITS
    : CLEAR UNUSED BRANCH LEVEL BITS
    : CLEAR UNUSED BRANCH LEVEL BITS
    : SET UP TO SHIFT BR LEVEL
    : SET UP TO SHIFT BR LEVEL
    : SHIFT BY ONE BIT
    : SHIFT BY ONE BIT
    : COUNT SHIFTS
    : COUNT SHIFTS
    : IF INCOMPLETE, BRANCH
    : IF INCOMPLETE, BRANCH
    : SAVE THE BRANCH LEVEL
    : SAVE THE BRANCH LEVEL
    : PRINT BEGINNING OF INTERRUPT MESSAGE
    : PRINT BEGINNING OF INTERRUPT MESSAGE
    : MOV R3,-(SP)
    : MOV R3,-(SP)
    : MOV (R5),-(SP)
    : MOV (R5),-(SP)
    : JSR R1,LPNTX
    : JSR R1,LPNTX
    : .WORD INTSTO
    : .WORD INTSTO
    : .WORD PNT.CT
    : .WORD PNT.CT

    1$:
    :
    : ASSUME C.UADR EQ 0
    : ASSUME C.UADR EQ 0
    : SETVEC R3,#INTSRV,#PRI00
    : SET UP INTERRUPT ROUTINE
    : SET UP INTERRUPT ROUTINE
    : MOV #PRI00,-(SP)
    : MOV #INTSRV,-(SP)
    : MOV R3,-(SP)
    : MOV R3,-(SP)
    : MOV #3,-(SP)
    : MOV #3,-(SP)
    : TRAP C$SVEC
    : TRAP C$SVEC
    : ADD #10,SP
    : ADD #10,SP
    : SETPRI #PRI00
    : SET PRIORITY TO 0 TO CHECK INTERRUPTS
    : SET PRIORITY TO 0 TO CHECK INTERRUPTS
    : MOV #PRI00,R0
    : MOV #PRI00,R0
    : TRAP C$SPRI
    : TRAP C$SPRI
    : ASR R3
    : ASR R3
    : ASR R3
    : ASR R3
    : BIS #<SA.STP+SA.INT>,R3
    : BIS #<SA.STP+SA.INT>,R3
    : CLR INTRCV
    : CLR INTRCV
    : CLR (R4)
    : CLR (R4)
    : INIT UDA
    : INIT UDA
    : MOV #SA.S1,UDARSD
    : MOV #SA.S1,UDARSD
    : CALL UDARSP
    : CALL UDARSP
    : WAIT FOR COMPLETION
    : WAIT FOR COMPLETION
    : MOV R3,2(R4)
    : MOV R3,2(R4)
    : MOVE STEP 1 DATA TO UDA
    : MOVE STEP 1 DATA TO UDA
    : SET UP TIMEOUT OF 10 SECONDS
    : SET UP TIMEOUT OF 10 SECONDS

    9$:
    :
    : POINT TO CONTROLLER TABLE
    : POINT TO CONTROLLER TABLE
    : TST INTRCV
    : TST INTRCV
    : BNE 11$
    : BNE 11$
    : BREAK
    : BREAK
    : TRAP CSBRK
    : TRAP CSBRK
    
```

1	030776	005737	002354	TST	KW.CSR			: SEE IF CLOCK ON SYSTEM
2	031002	001771		BEQ	9\$			
3	031004	023765	002366	000042	CMP	KW.EL+2,C.TOH(R5)		: SEE IF TIME ELAPSED
4	031012	101041		BHI	3\$			
5	031014	001364		BNE	9\$			
6	031016	023765	002364	000040	CMP	KW.EL,C.TO(R5)		
7	031024	103760		BLO	9\$			
8	031026	000433		BR	3\$: BRANCH
9	031030	005037	002240	11\$:	CLR	INTRCV		: FLAG AS NO INTERRUPTS RECEIVED
10	031034			SETPRI	#PRI07			: SET PRIORITY AS HIGHEST PRIORITY
	031034	012700	000340					MOV #PRI07,R0
	031040	104441						TRAP C\$SPRI
11	031042	005064	000002	CLR	2(R4)			: WRITE SECOND STEP TO UDA
12	031046	012702	000144	MOV	#100.,R2			: SET UP DELAY SO WE KNOW WE'RE INTERRUPTED
13	031052	005302		12\$:	DEC	R2		: DECREMENT COUNT
14	031054	001376		BNE	12\$: IF INCOMPLETE, BRANCH
15	031056	012701	000007	MOV	#7.,R1			: R1 IS PROCESS PRIORITY LEVEL
16	031062			2\$:	PUSH	R1		: SAVE PRIORITY
	031062	010146						MOV R1,-(SP)
17	031064	012702	000005	MOV	#5.,R2			: SET UP FOR SHIFTING PRIORITY
18	031070	006301		10\$:	ASL	R1		: SHIFT PRIORITY
19	031072	005302		DEC	R2			: DECREMENT SHIFT COUNT
20	031074	001375		BNE	10\$: IF INCOMPLETE, BRANCH
21	031076			SETPRI	R1			: SET RUNNING PRIORITY TO R1
	031076	010100						MOV R1,R0
	031100	104441						TRAP C\$SPRI
22	031102			POP	R1			: RESTORE R1
	031102	012601						MOV (SP)+,R1
23	031104	005737	002240	TST	INTRCV			: SEE IF INTERRUPT RECEIVED
24	031110	001007		BNE	4\$: IF SO, BRANCH
25	031112	005301		DEC	R1			: DECREMENT PRIORITY LEVEL
26	031114	100362		BPL	2\$: IF ALL LEVELS UNTESTED, BRANCH
27	031116			3\$:	ERRDF	28.,ERR028		: REPORT NO INTERRUPTS ERROR
	031116	104455						TRAP C\$ERDF
	031120	000034						.WORD 28
	031122	000000						.WORD 0
	031124	013670						.WORD ERR028
28	031126	000420		BR	6\$: BRANCH

1 031206
031206
031206 104402
2 031210 005004
3 031212 004737 023666
4 031216
031216
031216 104403

BGNSUB; 5

ENDSUB

CLR R4
CALL UDAINT

T1.5: TRAP CSBSUB
: INITIALIZE UDA WITH SMALLEST
: RING BUFFER AND INTERRUPTS DISABLED

L10054: TRAP CSESUB

```
1 031220          BGNSUB; 6
  031220
  031220 104402
2 031222 012704 126400      MOV  #<SA.STP+<5*SA.MS1>+<5*SA.CM1>>,R4      T1.6: TRAP  CSBSUB
3 031226 004737 023666      CALL UDAINT          ; INITIALIZE UDA WITH RING BUFFER
4                                     ; LARGE ENOUGH TO COVER NORMAL HOST COMM AREA
5                                     ; PACKET AND BUFFER SPACE (A 5 IN MES
6 031232          ENDSUB          ; LENGTH AND A 5 IN CMD LENGTH)
  031232
  031232 104403          L10055: TRAP  C$ESUB
```

```
1 031234          BGNSUB; 7
  031234
  031234 104402
2 031236          PUSH   FFREE          ; SAVE FREE MEMORY PARAMETERS
  031236 013746 002176
3 031242          PUSH   FSIZE
  031242 013746 002200
4 031246          MOV    #1,R1          ; RUN DM PROGRAM IN
  031246 012701 000001          ; ONE CONTROLLER ONLY
5 031252          CALL   RUNDM
  031252 004737 014632
6 031256          BEQ   1$
  031256 001402
7 031260          CALL   RESPDM
  031260 004737 014726
8 031264          1$: POP    FSIZE
  031264 012637 002200
9 031270          POP    FFREE          MOV (SP)+,FSIZE
  031270 012637 002176          MOV (SP)+,FFREE
10 031274         ENDSUB
   031274
   031274 104403
11 031276         L10056: TRAP   C$ESUB
   031276 000137 030266          JMP T1SKIP
12 031302         L10047: TRAP   C$ETST
13 031302
14 031302 104401
   031302
```



```

1          :WCHNG
2          :
3          :
4          :      WAIT UNTIL UDASA CHANGES FROM WHAT IS IN WCHNGD
5 031304 012700 000012      WCHNG:  MOV #10,,R0          ;SET TIMEOUT FOR 10 SECONDS
6 031310 010501              MOV R5,R1          ;POINT TO CONTROLLER TABLE
7 031312 062701 000040      ADD #C.TO,R1
8 031316 004737 023604      CALL SETTO
9 031322 026437 000002 031400 1$:  CMP 2(R4),WCHNGD      ;SEE IF CHANGED
10 031330 001022              BNE 2$
11 031332              BREAK
12 031332 104422              TST KW.CSR          ;SEE IF CLOCK ON SYSTEM      TRAP    C$BRK
13 031334 005737 002354      BEQ 1$
14 031340 001770              CMP KW.EL+2,C.TO(R5)      ;CHECK IF TIME OUT OCCURRED
15 031342 023765 002366 000042  BHI 3$
16 031350 101005              BNE 1$
17 031352 001363              CMP KW.EL,C.TO(R5)
18 031354 023765 002364 000040  BLO 1$
19 031362 103757              ERRDF 27,,ERR027      ; REPORT ERROR
20 031364              TRAP    C$ERDF
    031364 104455              .WORD 27
    031366 000033              .WORD 0
    031370 000000              .WORD ERR027
    031372 013652
21 031374 000264              SEZ          ; FLAG AS ERROR
22 031376 000207              RETURN      ; RETURN TO CALLING PROGRAM
23
24 031400      WCHNGD: .BLKW 1          ; OLD PORT CONTENTS
25 031402      BRLEV: .BLKW 1          ; WORD FOR BRANCH LEVEL STORAGE
    
```

```

1          .SBTTL TEST 2: DISK RESIDENT DIAGNOSTIC TEST
2
3 031404          BGNTST
4          T2::
5 031404 012701 000002          MOV #2,R1          ;INIT TEST PARAMETERS
6 031410 004737 014504          CALL TINIT
7
8 031414 013737 002206 002212          MOV CTABS,TSTTAB          ;GET POINTER TO FIRST CONTROLLER TABLE
9
10 031422 004737 025334          T2NEXT: CALL RESET          ;RESET ALL UNITS
11 031426          PUSH FFREE          ;SAVE FREE MEMORY PARAMETERS
12 031426 013746 002176          MOV FFREE,-(SP)
13 031432          PUSH FSIZE          MOV FSIZE,-(SP)
14 031432 013746 002200
15 031436 012701 000001          MOV #1,R1          ;RUN DM PROGRAM IN
16 031442 004737 014632          CALL RUNDM          ; ONE CONTROLLER ONLY
17 031446 001402          BEQ 1$
18 031450 004737 014726          CALL RESPDM
19 031454          POP FSIZE          1$:
20 031454 012637 002200          MOV (SP)+,FSIZE
21 031460          POP FFREE          MOV (SP)+,FFREE
22 031460 012637 002176
23
24 031464 062737 000046 002212          ADD #C.SIZE,TSTTAB          ;MOVE TO NEXT CONTROLLER
25 031472 005777 150514          TST @TSTTAB          ;CHECK IF ANY MORE CONTROLLER TABLES
26 031476 001351          BNE T2NEXT
27
28          ENDTST
29
30          L10057: TRAP C$ETST
  
```

```
1          .SBTTL TEST 3: DISK FUNCTION TEST
2
3 031502          BGNTST
4          T3::
5 031502 012701 000003          MOV #3,R1          ;INITIALIZE TEST PARAMETERS
6 031506 004737 014504          CALL TINIT
7
8 031512 013737 002206 002212          MOV CTABS,TSTTAB          ;GET FIRST TABLE ADDRESS
9 031520 013701 002210          MOV CTRLRS,R1          ;RUN DM PROGRAM ON ALL CONTROLLERS
10 031524 004737 014632          CALL RUNDM          ; AT ONCE
11 031530 001402          BEQ 1$
12 031532 004737 014726          CALL RESPDM
13 031536          1$:
14 031536          ENDTST
   031536          L10060: TRAP CSETST
   031536 104401
```



```

1          .SBTTL TEST 4: DISK EXERCISER
2
3 031540    BGNTST
4          T4::
8 031540 022737 000004 002230    CMP #4,TNUM          ;CHECK IF TEST 4 WAS IN PROGRESS
10 031546 001053          BNE T4$TRT          ;BRANCH IF NOT
11 031550 022737 000002 002226    CMP #ICONT,IFLAGS   ;CHECK IF HERE BY CONTINUE COMMAND
12 031556 001047          BNE T4$TRT          ;BRANCH IF NOT
13 031560 005037 002226          CLR IFLAGS          ;CLEAR FLAGS FOR NEXT TIME HERE
14 031564 013704 002406          MOV LBUFS,R4        ;GET LOG BUFFER POINTER
15 031570 001423          BEQ LOGCHK          ; IF ZERO, NONE EXISTS
16 031572          PNTF LOGM1          ;INTRODUCE ERROR LOG
    031572 004137 022266          JSR R1,LPNTF
    031576 006132          .WORD LOGM1
    031600 000000          .WORD PNT.CT
17 031602 005037 002406          CLR LBUFS          ;CLEAR START ADDRESS TO ERASE BUFFER
18 031606 012405          LOGOUT: MOV (R4)+,R5 ;GET CONTROLLER TABLE ADDRESS
19 031610 004737 022372          CALL PNTERR        ;PRINT ERROR REPORT
20 031614 062704 000104          ADD #<HC.BSZ-2>,R4 ;BUMP POINTER TO NEXT ENTRY
21 031620 020437 002410          CMP R4,LBUFN       ;CHECK IF AT END
22 031624 103770          BLO LOGOUT         ;PRINT ALL ENTRIES
23 031626 004137 022266          PNTF LOGM2
    031626 004137 022266          JSR R1,LPNTF
    031632 006164          .WORD LOGM2
    031634 000000          .WORD PNT.CT
24 031636 000410          BR T4CON
25
26 031640 032737 001000 002160 LOGCHK: BIT #SM.LOG,SFPTBL+SO.BIT ;CHECK IF LOG ENABLED
27 031646 001404          BEQ T4CON
28 031650          PNTF LOGM3          ;REPORT LOG EMPTY
    031650 004137 022266          JSR R1,LPNTF
    031654 006211          .WORD LOGM3
    031656 000000          .WORD PNT.CT
29 031660 005737 002234          T4CON: TST URNING  ;CHECK IF ANY CONTROLLERS STILL RUNNING
30 031664 001404          BEQ T4$TRT        ;RESTART IF NOT
31 031666 004737 014726          CALL RESPDM       ;CONTINUE BY RESPONDING TO REQUESTS
32 031672 000137 032244          JMP T4WAIT        ;END OF TEST WHEN DONE
    
```

```

1          ;START TEST 4
2
3 031676 032737 000014 002226 T4STRT: BIT #ISTRT+IREST,IFLAGS      ;HERE FROM OPERATOR COMMAND?
4 031704 001534                      BEQ T4RUN                    ;RUN WITH PREVIOUS PARAMETERS IF NEW PASS
5 031706 032737 000200 002160      BIT #SM.MAN,SFPTBL+SO.BIT    ;MANUAL INTERVENTION MODE?
6 031714 001476                      BEQ T4DEF                    ;IF NOT, SET UP DEFAULT PARAMETERS
7 031716 104450                      MANUAL                        ;MANUAL INTERVENTION ALLOWED?
8 031720                      BNCOMPLETE T4DEFW                    ;IF NOT, GIVE WARNING      TRAP    CSMANI
9 031720 103070                      ;IF NOT, GIVE WARNING      BCC    T4DEFW
10
11          ;INPUT PARAMETERS
12 031722 005037 002236              CLR UCNT                    ;CLEAR COUNT OF UNITS USING PATTERN 16
13 031726 013705 002206              MOV CTABS,R5                ;GET FIRST CONTROLLER TABLE
14 031732 012702 000010      T4PRM1: MOV #8.,R2          ;GET COUNT OF DRIVE TABLES
15 031736 010504              MOV R5,R4                  ;GET FIRST DRIVE TABLE POINTER
16 031740 062704 000020              ADD #C.DR0,R4
17 031744 012403      T4PRM2: MOV (R4)+,R3        ;GET DRIVE TABLE ADDRESS
18 031746 001416              BEQ T4PRM4                ;GO TO NEXT CONTROLLER IF NONE
19 031750 032763 100000 000002      BIT #DT.AVL,D.UNIT(R3)    ;SEE IF TO BE TESTED
20 031756 001010              BNE T4PRM3
21 031760 004737 032266              CALL T4QUEST                ;ASK QUESTIONS
22 031764 022763 000020 000006      CMP #16.,D.PAT(R3)
23 031772 001002              BNE T4PRM3
24 031774 005237 002236              INC UCNT
25 032000 005302      T4PRM3: DEC R2                ;COUNT DRIVE TABLES
26 032002 001360              BNE T4PRM2                ;GO LOOK AT NEXT
27 032004 062705 000046      T4PRM4: ADD #C.SIZE,R5    ;GO TO NEXT CONTROLLER
28 032010 005715              TST (R5)                   ; IF THERE IS ONE
29 032012 001347              BNE T4PRM1
    
```

```

1      ;NOW GET DATA PATTERN 16 IF SELECTED BY ANY DRIVE
2
3      032014      104443      GMANID T4DPC,PAT16C,D,-1,1,16.,YES      ;COUNT OF WORDS
      032014      000406      TRAP      CSGMAN
      032016      000406      BR      10000$
      032020      002312      .WORD   PAT16C
      032022      000052      .WORD   T$CODE
      032024      003507      .WORD   T4DPC
      032026      177777      .WORD   -1
      032030      000001      .WORD   T$LOLIM
      032032      000020      .WORD   T$HILIM
      032034      10000$:
4      032034      013701      002312      MOV PAT16C,R1      ;GET COUNT OF WORDS
5      032040      012704      002314      MOV #PAT16W,R4    ;GET ADDRESS OF STORAGE
6      032044      011437      002262      T4PRM5: MOV (R4),TEMP
7      032050      032050      104443      GMANID T4DPD,TEMP,0,-1,0,-1,YES ;DATA WORD
      032050      000406      TRAP      CSGMAN
      032052      002262      BR      10001$
      032054      000032      .WORD   TEMP
      032056      003552      .WORD   T$CODE
      032060      177777      .WORD   T4DPD
      032062      000000      .WORD   -1
      032064      177777      .WORD   T$LOLIM
      032066      10001$:
8      032070      013724      002262      MOV TEMP,(R4)+
9      032074      005301      DEC R1      ;COUNT THE WORDS
10     032076      001362      BNE T4PRM5
11     032100      000436      BR T4RUN
12
13     ;GIVE WARNING MANUAL INTERVENTION NOT ALLOWED
14
15     032102      004137      022266      T4DEFW: PNTF T4WARN
      032102      004406      JSR R1,LPNTF
      032106      000000      .WORD   T4WARN
      032110      .WORD   PNT.CT
    
```



```

1          ;SET UP DEFAULT PARAMETERS
2
3 032112 013705 002206 T4DEF:  MOV CTABS,R5          ;GET FIRST CONTROLLER TABLE
4 032116 012702 000010 T4DEFA: MOV #8.,R2          ;GET COUNT OF DRIVE TABLES
5 032122 010504          MOV R5,R4          ;GET FIRST DRIVE TABLE POINTER
6 032124 062704 000020          ADD #C.DRO,R4
7 032130 012403 T4DEFB: MOV (R4)+,R3          ;GET DRIVE TABLE ADDRESS
8 032132 001415          BEQ T4DEFB          ;GO TO NEXT CONTROLLER IF NONE
9 032134 062703 000004          ADD #D.PRM,R3
10 032140          AND D.DCY,(R3)          ;INITIALIZE ALL PARAMETER BITS
11 032140 042713 157777          BIC #^C<D.DCY>,(R3)
12 032144 052723 011012          BIS #DDEF,(R3)+
13 032150 012700 000067          MOV #55.,R0
14 032154 005023 T4DEFC: CLR (R3)+
15 032156 005300          DEC R0
16 032160 001375          BNE T4DEFC
17 032162 005302 T4DEFD: DEC R2          ;COUNT DRIVE TABLES
18 032164 001361          BNE T4DEFB          ;GO LOOK AT NEXT
19 032166 062705 000046 T4DEFE: ADD #C.SIZE,R5          ;GO TO NEXT CONTROLLER
20 032172 005715          TST (R5)          ; IF THERE IS ONE
21 032174 001350          BNE T4DEFA
22
23          ;START TEST 4
24 032176 006137 002226 T4RUN:  ROL IFLAGS          ;CLEAR FLAGS FOR NEXT TIME HERE
25 032202          AND ISTRTH,IFLAGS          ;HOLD START FOR T4UPRM REQUEST
26 032202 042737 177757 002226          BIC #^C<ISTRTH>,IFLAGS
29 032210 012701 000004          MOV #4,R1          ;INITIALIZE TEST PARAMETERS
31 032214 004737 014504          CALL TINIT
32
33 032220 013737 002206 002212          MOV CTABS,TSTTAB          ;GET FIRST TABLE ADDRESS
34 032226 013701 002210          MOV CTRLRS,R1          ;RUN DM PROGRAM ON ALL CONTROLLERS
35 032232 004737 014632          CALL RUNDM          ; AT ONCE
36 032236 001402          BEQ T4WAIT
37 032240 004737 014726          CALL RESPDM
38 032244 032737 001000 002160 T4WAIT: BIT #SM.LOG,SFPTBL+SO.BIT          ;CHECK IF LOG IS ENABLED
39 032252 001402          BEQ T4EXIT          ;EXIT IF NOT
40 032254          BREAK
41 032254 104422          TRAP CSBRK
42 032256 000772          BR T4WAIT          ;WAIT TILL STOPPED BY CONTROL C
43 032260          T4EXIT: DORPT          ;PRINT STATISTICS
44 032260 104424          TRAP CSDRPT
45 032262 104432          TRAP CSEXIT
46 032264 001632          .WORD L10061-
    
```

```

1
2
3
4
5
6
7
8
9
10
11 032266      010246      T4QUEST:PUSH <R2,R4>
    032266      010446
12 032272      011346      PNTF T4QHED,D.UNIT(R3),(R5),(R3)      ;PRINT HEADER
    032272      011546
    032274      016346      000002
    032302      004137      022266
    032306      004513
    032310      000006
13 032312      016337      000010      002262      MOV D.BB(R3),TEMP
14 032320      104443      GMANID T4BB,TEMP,D,-1,0,16.,YES ;NUMBER OF BAD BLOCKS
    032322      000406
    032324      002262
    032326      000052
    032330      002516
    032332      177777
    032334      000000
    032336      000020
    032340
15 032340      013763      002262      000010      MOV TEMP,D.BB(R3)
16 032346      001424      BEQ T4Q02
17
18 032350      010304      MOV R3,R4
19 032352      062704      000012      ADD #D.BB01,R4
20 032356      013701      002262      MOV TEMP,R1

```

:ASK TEST 4 MANUAL INTERVENTION QUESTIONS

:INPUTS:

R5 - POINTER TO CONTROLLER TABLE
 R3 - POINTER TO DRIVE TABLE
 R2 AND R4 MUST BE PRESERVED

:OUTPUTS:

DRIVE TABLE WITH NEW PARAMETERS
 R0 AND R1 CONTENTS DESTROYED

```

MOV R2,-(SP)
MOV R4,-(SP)
MOV (R3),-(SP)
MOV (R5),-(SP)
MOV D.UNIT(R3),-(SP)
JSR R1,LPNTF
.WORD T4QHED
.WORD PNT.CT

```

```

TRAP      CS$GMAN
BR        10002$
.WORD     TEMP
.WORD     T$CODE
.WORD     T4BB
.WORD     -1
.WORD     T$LOLIM
.WORD     T$HILIM

```

10002\$:

```

:GET POINTER TO STORAGE
:FOR BAD BLOCKS
:GET COUNT OF BLOCKS TO INPUT

```


1	032362	004737	021402	T4Q01:	CALL BLD28	:BUILD DEFAULT ANSWER		
2	032366				GMANID T4BBI,TEMP,A,-1,0,9.,YES	:BAD BLOCK		
	032366	104443					TRAP	CSGMAN
	032370	000406					BR	10003\$
	032372	002262					.WORD	TEMP
	032374	000152					.WORD	T\$CODE
	032376	002633					.WORD	T4BBI
	032400	177777					.WORD	-1
	032402	000000					.WORD	T\$LOLIM
	032404	000011					.WORD	T\$HILIM
	032406							
3	032406	004737	021504	T4Q02:	CALL CNV28	:CONVERT TO BINARY	10003\$:	
4	032412	103763			BCS T4Q01	:REPEAT UNTIL RIGHT		
5	032414	005301			DEC R1	:DECREMENT COUNT		
6	032416	001361			BNE T4Q01	:GET ALL NUMBERS		
7	032420							
8	032420				GMANIL T4DMN,TEMP,1,YES			
	032420	104443					TRAP	CSGMAN
	032422	000404					BR	10004\$
	032424	002262					.WORD	TEMP
	032426	000130					.WORD	T\$CODE
	032430	002543					.WORD	T4DMN
	032432	000001					.WORD	1
	032434							
9	032434	032737	000001 002262		BIT #1,TEMP		10004\$:	
10	032442	001002			BNE 1\$			
11	032444	000137	034110		JMP T4Q30			
12	032450			1\$:				
13	032450	016337	000004 002262		MOV D.PRM(R3),TEMP	:GET PARAMETER BITS		
14	032456				GMANIL T4RO,TEMP,D.RO,YES	:READ ONLY		
	032456	104443					TRAP	CSGMAN
	032460	000404					BR	10005\$
	032462	002262					.WORD	TEMP
	032464	000130					.WORD	T\$CODE
	032466	002645					.WORD	T4RO
	032470	004000					.WORD	D.RO
	032472							
15	032472	032737	004000 002262		BIT #D.RO,TEMP		10005\$:	
16	032500	001404			BEQ T4Q03	:IF NOT READ ONLY, GO TO WRITE ONLY QUESTION		
17	032502	042737	002000 002262		BIC #D.WO,TEMP	:ELSE, CLEAR WRITE ONLY BIT		
18	032510	000432			BR T4Q05	: AND BRANCH AROUND WRITE ONLY QUESTION		
19	032512			T4Q03:				
20	032512				GMANIL T4WO,TEMP,D.WO,YES	:WRITE ONLY		
	032512	104443					TRAP	CSGMAN
	032514	000404					BR	10006\$
	032516	002262					.WORD	TEMP
	032520	000130					.WORD	T\$CODE
	032522	002657					.WORD	T4WO
	032524	002000					.WORD	D.WO
	032526							
21	032526				GMANIL T4WCA,TEMP,D.WCA,YES	:CHECK ALL WRITES	10006\$:	
	032526	104443					TRAP	CSGMAN
	032530	000404					BR	10007\$
	032532	002262					.WORD	TEMP
	032534	000130					.WORD	T\$CODE
	032536	002672					.WORD	T4WCA
	032540	000004					.WORD	D.WCA

032542
22 032542 032737 000004 002262
23 032550 001007
24 032552
032552 104443
032554 000404
032556 002262
032560 000130
032562 002726
032564 000010
032566
25 032566 000403

BIT #D.WCA,TEMP
BNE T4Q04
GMANIL T4WCR,TEMP,D.WC,YES

10007\$:
:CHECK ANSWER
:BRANCH IF YES
:RANDOMLY CHECK WRITES

TRAP
BR
.WORD
.WORD
.WORD
.WORD
CSGMAN
10010\$
TEMP
T\$CODE
T4WCR
D.WC

BR T4Q05

10010\$:

1	032570	052737	000010	002262	T4Q04:	BIS #D.WC,TEMP		:BOTH BITS GET SET		
2	032576	013763	002262	000004	T4Q05:	MOV TEMP,D.PRM(R3)		:PUT PARAM BITS BACK		
3	032604	016337	000006	002262		MOV D.PAT(R3),TEMP				
4	032612					GMANID T4DP,TEMP,D,-1,0,16.,YES		:DATA PATTERN		
	032612	104443							TRAP	CSGMAN
	032614	000406							BR	10011\$
	032616	002262							.WORD	TEMP
	032620	000052							.WORD	T\$CODE
	032622	002767							.WORD	T4DP
	032624	177777							.WORD	-1
	032626	000000							.WORD	T\$LOLIM
	032630	000020							.WORD	T\$HILIM
	032632									10011\$:
5	032632	013763	002262	000006		MOV TEMP,D.PAT(R3)				
6	032640	016337	000004	002262		MOV D.PRM(R3),TEMP		:GET PARAM BITS AGAIN		
7	032646	032737	004000	002262	T4Q06:	BIT #D.R0,TEMP		:BYPASS NEXT 3 IF ONLY WRITING		
8	032654	001010				BNE T4Q07				
9	032656	032737	002000	002262		BIT #D.W0,TEMP				
10	032664	001404				BEQ T4Q07				
11	032666	032737	000010	002262		BIT #D.WC,TEMP				
12	032674	001432				BEQ T4Q09				
13	032676				T4Q07:	GMANIL T4ECC,TEMP,D.ECC,YES		:ENABLE ECC		
	032676	104443							TRAP	CSGMAN
	032700	000404							BR	10012\$
	032702	002262							.WORD	TEMP
	032704	000130							.WORD	T\$CODE
	032706	003035							.WORD	T4ECC
	032710	010000							.WORD	D.ECC
	032712									10012\$:
14	032712					GMANIL T4DCA,TEMP,D.DCA,YES		:COMPARE ALL DATA		
	032712	104443							TRAP	CSGMAN
	032714	000404							BR	10013\$
	032716	002262							.WORD	TEMP
	032720	000130							.WORD	T\$CODE
	032722	003070							.WORD	T4DCA
	032724	000001							.WORD	D.DCA
	032726									10013\$:
15	032726	032737	000001	002262		BIT #D.DCA,TEMP		:CHECK ANSWER		
16	032734	001007				BNE T4Q08		:BRANCH IF YES		
17	032736					GMANIL T4DCR,TEMP,D.DC,YES		:RANDOMLY CHECK WRITES		
	032736	104443							TRAP	CSGMAN
	032740	000404							BR	10014\$
	032742	002262							.WORD	TEMP
	032744	000130							.WORD	T\$CODE
	032746	003116							.WORD	T4DCR
	032750	000002							.WORD	D.DC
	032752									10014\$:
18	032752	000403				BR T4Q09				

```

1 032754 052737 000002 002262 T4Q08: BIS #D.DC,TEMP           ;BOTH BITS GET SET
2 032762 104443 T4Q09: GMANIL T4RET,TEMP,D.RET,YES ;ENABLE RETRIES
   032762 104443
   032764 000404 TRAP CS$GMAN
   032766 002262 BR 10015$
   032770 000130 .WORD TEMP
   032772 003151 .WORD T$CODE
   032774 001000 .WORD T4RET
   032776 .WORD D.RET
3 032776 005137 002262 COM TEMP 10015$:
4 033002 GMANIL T4SEK,TEMP,D.SEQ,YES ;ENABLE SEEKS
   033002 104443 TRAP CS$GMAN
   033004 000404 BR 10016$
   033006 002262 .WORD TEMP
   033010 000130 .WORD T$CODE
   033012 003170 .WORD T4SEK
   033014 000100 .WORD D.SEQ
   033016 10016$:
5 033016 005137 002262 COM TEMP ;COMPLIMENTED
6 033022 013763 002262 000004 MOV TEMP,D.PRM(R3)
7
8 033030 005037 002262 CLR TEMP ;DETERMINE DEFAULT SELECTION
9 033034 032763 000040 000004 BIT #D.BE,D.PRM(R3) ;IF D.BE SET - LOAD 1
10 033042 001403 BEQ T4Q10 ;IF D.CYL CLEAR - LOAD 0
11 033044 005237 002262 INC TEMP ;IF D.BEC CONTAINS 0 - LOAD 4
12 033050 000422 BR T4Q11 ;IF D.TR SET - LOAD 2
13 033052 032763 000400 000004 T4Q10: BIT #D.CYL,D.PRM(R3) ;LOAD 3
14 033060 001416 BEQ T4Q11
15 033062 012737 000004 002262 MOV #4,TEMP
16 033070 005763 000112 TST D.BEC(R3)
17 033074 001410 BEQ T4Q11
18 033076 005337 002262 DEC TEMP
19 033102 032763 000020 000004 BIT #D.TR,D.PRM(R3)
20 033110 001402 BEQ T4Q11
21 033112 005337 002262 DEC TEMP
    
```



```
1 033116          T4Q11: PNTF T4OPT1
  033116 004137 022266
  033122 004620
  033124 000000
2 033126          GMANID T4OPT7,TEMP,D,-1,0,4,YES ;WHICH SELECTION LIMITS
  033126 104443
  033130 000406
  033132 002262
  033134 000052
  033136 003213
  033140 177777
  033142 000000
  033144 000004
  033146
3 033146 005337 002262          DEC TEMP
4 033152 002004          BGE T4Q12
5 033154 042763 000440 000004      BIC #D.BE+D.CYL,D.PRM(R3)
6 033162 000467          BR T4Q19

                                10017$:
                                :SET UP D.PRM FROM ANSWER
                                :IF 0 - CLEAR D.BE AND D.CYL

                                JSR R1,LPNTF
                                .WORD T4OPT1
                                .WORD PNT.CT

                                TRAP      CS$GMAN
                                BR        10017$
                                .WORD    TEMP
                                .WORD    T$CODE
                                .WORD    T4OPT7
                                .WORD    -1
                                .WORD    T$LOLIM
                                .WORD    T$HILIM
```

TEST 4: DISK EXERCISER

1	033164	005337	002262		T4Q12:	DEC TEMP		:IF 1
2	033170	002013				BGE T4Q13		: IF D.BE NOT SET
3	033172	032763	000040	000004		BIT #D.BE,D.PRM(R3)		: SET D.BE
4	033200	001060				BNE T4Q19		: CLEAR D.CYL
5	033202	052763	000040	000004		BIS #D.BE,D.PRM(R3)		: LOAD 1 IN D.BEC
6	033210	042763	000400	000004		BIC #D.CYL,D.PRM(R3)		: CLEAR BLOCK STORAGE
7	033216	000436				BR T4Q16		
8	033220	042763	000040	000004	T4Q13:	BIC #D.BE,D.PRM(R3)		:IF 2, 3 OR 4
9								: CLEAR D.BE
10	033226	022737	000002	002262		CMP #2,TEMP		:IF 4
11	033234	001006				BNE T4Q14		: SET D.CYL
12	033236	052763	000400	000004		BIS #D.CYL,D.PRM(R3)		: CLEAR D.BEC
13	033244	005063	000112			CLR D.BEC(R3)		
14	033250	000434				BR T4Q19		
15	033252				T4Q14:	PUSH D.PRM(R3)		:IF 2 OR 3
	033252	016346	000004					MOV D.PRM(R3),-(SP)
16	033256	052763	000420	000004		BIS #D.CYL+D.TR,D.PRM(R3)		: SAVE D.PRM BITS
17	033264	005337	002262			DEC TEMP		: SET D.CYL AND D.TR
18	033270	100403				BMI T4Q15		: IF 3
19	033272	042763	000020	000004		BIC #D.TR,D.PRM(R3)		: CLEAR D.TR
20	033300	022663	000004		T4Q15:	CMP (SP)+,D.PRM(R3)		: IF D.CYL OR D.TR CHANGED OR D.BEC = 0
21	033304	001003				BNE T4Q16		
22	033306	005763	000112			TST D.BEC(R3)		: LOAD 1 IN D.BEC
23	033312	001013				BNE T4Q19		: CLEAR BLOCK STORAGE
24	033314	012763	000001	000112	T4Q16:	MOV #1,D.BEC(R3)		
25	033322	010304			T4Q17:	MOV R3,R4		
26	033324	062704	000114			ADD #D.BGN1,R4		
27	033330	012701	000020			MOV #16.,R1		
28	033334	005024			T4Q18:	CLR (R4)+		
29	033336	005301				DEC R1		
30	033340	001375				BNE T4Q18		

```
1 033342 032763 000040 000004 T4Q19: BIT #D.BE,D.PRM(R3) ;NOW ASK THE QUESTIONS TO ALLOW THE
2 033350 001460 BEQ T4Q22 ; NUMBERS TO CHANGE
3 033352 016337 000112 002262 MOV D.BEC(R3),TEMP
4 033360 104443 GMANID T4BE,TEMP,D,-1,1,4,YES ;NUMBER OF B/E SETS
033360 104443 TRAP CS$GMAN
033362 000406 BR 10020$
033364 002262 .WORD TEMP
033366 000052 .WORD T$CODE
033370 003216 .WORD T4BE
033372 177777 .WORD -1
033374 000001 .WORD T$LOLIM
033376 000004 .WORD T$HILIM
033400 10020$:
5 033400 013763 002262 000112 MOV TEMP,D.BEC(R3)
6 033406 013701 002262 MOV TEMP,R1 ;GET COUNT OF SETS
7 033412 010304 MOV R3,R4 ;GET POINTER TO STORAGE AREA
8 033414 062704 000114 ADD #D.BGN1,R4
9 033420 004737 021402 T4Q20: CALL BLD28
10 033424 104443 GMANID T4BEG,TEMP,A,-1,0,9.,YES ;BEGIN BLOCK
033424 104443 TRAP CS$GMAN
033426 000406 BR 10021$
033430 002262 .WORD TEMP
033432 000152 .WORD T$CODE
033434 003247 .WORD T4BEG
033436 177777 .WORD -1
033440 000000 .WORD T$LOLIM
033442 000011 .WORD T$HILIM
033444 10021$:
11 033444 004737 021504 CALL CNV28
12 033450 103763 BCS T4Q20
13 033452 004737 021402 T4Q21: CALL BLD28
14 033456 104443 GMANID T4END,TEMP,A,-1,0,9.,YES ;END BLOCK
033456 104443 TRAP CS$GMAN
033460 000406 BR 10022$
033462 002262 .WORD TEMP
033464 000152 .WORD T$CODE
033466 003263 .WORD T4END
033470 177777 .WORD -1
033472 000000 .WORD T$LOLIM
033474 000011 .WORD T$HILIM
033476 10022$:
15 033476 004737 021504 CALL CNV28
16 033502 103763 BCS T4Q21
17 033504 005301 DEC R1
18 033506 001344 BNE T4Q20
19 033510 000577 BR T4Q30
```



```

1 033512 032763 000400 000004 T4Q22: BIT #D.CYL,D.PRM(R3) ;IF D.CYL CLEAR - ALL DONE
2 033520 001573 BEQ T4Q30
3 033522 005763 000112 TST D.BEC(R3) ;IF D.BEC CLEAR - GO RIGHT TO B/E CYLS
4 033526 001526 BEQ T4Q27
5 033530 010304 MOV R3,R4
6 033532 062704 000112 ADD #D.BEC,R4
7 033536 032763 000020 000004 BIT #D.TR,D.PRM(R3) ;LOOK AT D.TR.TO DETERMINE QUESTION
8 033544 001434 BEQ T4Q24
9 033546 011437 002262 MOV (R4),TEMP
10 033552 033552 104443 GMANID T4TRC,TEMP,D,-1,1,7,YES ;NUMBER OF TRACKS
    033554 000406 TRAP CSGMAN
    033556 002262 BR 10023$
    033560 000052 .WORD TEMP
    033562 003275 .WORD T$CODE
    033564 177777 .WORD T4TRC
    033566 000001 .WORD -1
    033570 000007 .WORD T$LOLIM
    033572 .WORD T$HILIM
11 033572 013714 002262 MOV TEMP,(R4)
12 033576 012401 MOV (R4)+,R1 ;GET COUNT OF TRACKS
13 033600 011437 002262 T4Q23: MOV (R4),TEMP
14 033604 033604 104443 GMANID T4TRAK,TEMP,D,-1,0,255.,YES ;TRACK
    033606 000406 TRAP CSGMAN
    033610 002262 BR 10024$
    033612 000052 .WORD TEMP
    033614 003326 .WORD T$CODE
    033616 177777 .WORD T4TRAK
    033620 000000 .WORD -1
    033622 000377 .WORD T$LOLIM
    033624 .WORD T$HILIM
15 033624 013724 002262 MOV TEMP,(R4)+
16 033630 005301 DEC R1
17 033632 001362 BNE T4Q23
18 033634 000433 BR T4Q26
19 033636 011437 002262 T4Q24: MOV (R4),TEMP
20 033642 033642 104443 GMANID T4GRC,TEMP,D,-1,1,7,YES ;NUMBER OF GROUPS
    033644 000406 TRAP CSGMAN
    033646 002262 BR 10025$
    033650 000052 .WORD TEMP
    033652 003334 .WORD T$CODE
    033654 177777 .WORD T4GRC
    033656 000001 .WORD -1
    033660 000007 .WORD T$LOLIM
    033662 .WORD T$HILIM
21 033662 013714 002262 MOV TEMP,(R4)
22 033666 012401 MOV (R4)+,R1 ;GET COUNT OF GROUPS

```

1	033670	011437	002262	T4Q25:	MOV (R4),TEMP				
2	033674				GMANID T4GRP,TEMP,D,-1,0,255.,YES	:GROUP			
	033674	104443					TRAP	CSGMAN	
	033676	000406					BR	10026\$	
	033700	002262					.WORD	TEMP	
	033702	000052					.WORD	T\$CODE	
	033704	003365					.WORD	T4GRP	
	033706	177777					.WORD	-1	
	033710	000000					.WORD	TSLOLIM	
	033712	000377					.WORD	TSHILIM	
	033714							10026\$:	
3	033714	013724	002262		MOV TEMP,(R4)+				
4	033720	005301			DEC R1				
5	033722	001362			BNE T4Q25				
6	033724	016337	000162	002262	T4Q26:	MOV D.ECYL+2(R3),TEMP			
7	033732	005137	002262		COM TEMP				
8	033736				GMANIL T4CYL,TEMP,BIT15,YES	:WISH TO LIMIT CYLINDERS			
	033736	104443					TRAP	CSGMAN	
	033740	000404					BR	10027\$	
	033742	002262					.WORD	TEMP	
	033744	000130					.WORD	T\$CODE	
	033746	003373					.WORD	T4CYL	
	033750	100000					.WORD	BIT15	
	033752							10027\$:	
9	033752	005737	002262		TST TEMP				
10	033756	100412			BMI T4Q27				
11	033760	005063	000154		CLR D.BCYL(R3)				
12	033764	005063	000156		CLR D.BCYL+2(R3)				
13	033770	005063	000160		CLR D.ECYL(R3)				
14	033774	012763	177777	000162	MOV #-1,D.ECYL+2(R3)				
15	034002	000442			BR T4Q30				
16	034004	005763	000162		T4Q27:	TST D.ECYL+2(R3)			
17	034010	002002			BGE T4Q27A				
18	034012	005063	000162		CLR D.ECYL+2(R3)				
19	034016	010304			T4Q27A:	MOV R3,R4			
20	034020	062704	000154		ADD #D.BCYL,R4				
21	034024	004737	021402		T4Q28:	CALL BLD28			
22	034030				GMANID T4CYLB,TEMP,A,-1,0,9.,YES	:STARTING CYLINDER			
	034030	104443					TRAP	CSGMAN	
	034032	000406					BR	10030\$	
	034034	002262					.WORD	TEMP	
	034036	000152					.WORD	T\$CODE	
	034040	003445					.WORD	T4CYLB	
	034042	177777					.WORD	-1	
	034044	000000					.WORD	TSLOLIM	
	034046	000011					.WORD	TSHILIM	
	034050							10030\$:	
23	034050	004737	021504		CALL CNV28				
24	034054	103763			BCS T4Q28				

1 034056 004737 021402
2 034062
034062 104443
034064 000406
034066 002262
034070 000152
034072 003467
034074 177777
034076 000000
034100 000011
034102
3 034102 004737 021504
4 034106 103763
5 034110
034110 012604
034112 012602
6 034114 000207
7 034116
034116
034116 104401
9 034120

T4Q29: CALL BLD28
GMANID T4CYLE,TEMP,A,-1,0,9.,YES

T4Q30: CALL CNV28
BCS T4Q29
POP <R4,R2>

RETURN
ENDTST

ENDMOD

:ENDING CYLINDER
TRAP CS\$GMAN
BR 10031\$
.WORD TEMP
.WORD T\$CODE
.WORD T4CYLE
.WORD -1
.WORD T\$LOLIM
.WORD T\$HILIM

10031\$:

MOV (SP)+,R4
MOV (SP)+,R2

L10061:
TRAP C\$SETST

1
2
3 034120
4
5
6
7
8
9
10
11
12
13
14 034120
15 034120 000032
16 034122
17
18 034122
19
20 034122
21 034122
22 034122
23 034122
24 034122
25 034122
26 020000
27 034122

.SBTTL HARDWARE PARAMETER CODING SECTION

BGNMOD

;++
: THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
: THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
: MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
: INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
: MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
: WITH THE OPERATOR.
:--

BGNHRD

.WORD L10062-L\$HARD/2
L\$HARD::

;FORMAT OF HARDWARE P-TABLE IS AS FOLLOWS:

TABLE

;START A TEBLE DEFINITION

ITEM HO.UBA 2
ITEM HO.VEC 2
ITEM HO.BRL 2
ITEM HO.BST 2
ITEM HO.LDR 2
ITEM HO.PRM 2
HM.CYL == BIT13
END

: UNIBUS ADDRESS
: UDA VECTOR
: BR LEVEL
: BURST RATE
: DRIVE NUMBER
: PROGRAM PARAMETERS
: TEST CUSTOMER DATA AREA

1	034122				GPRMA	H.UBA,HO.UBA,0,160000,177774,YES		;BUS ADDRESS		
	034122	000031						.WORD	TSCODE	
	034124	034206						.WORD	H.UBA	
	034126	160000						.WORD	T\$LOLIM	
	034130	177774						.WORD	TSHILIM	
2	034132				GPRMA	H.VEC,HO.VEC,0,4,774,YES		; VECTOR		
	034132	001031						.WORD	TSCODE	
	034134	034234						.WORD	H.VEC	
	034136	000004						.WORD	T\$LOLIM	
	034140	000774						.WORD	TSHILIM	
3	034142				GPRMD	H.BRL,HO.BRL,D,-1,4.,7.,YES		; BR LEVEL		
	034142	002052						.WORD	TSCODE	
	034144	034243						.WORD	H.BRL	
	034146	177777						.WORD	-1	
	034150	000004						.WORD	T\$LOLIM	
	034152	000007						.WORD	TSHILIM	
4	034154				GPRMD	H.BST,HO.BST,D,-1,0.,63.,YES		; BURST RATE		
	034154	003052						.WORD	TSCODE	
	034156	034254						.WORD	H.BST	
	034160	177777						.WORD	-1	
	034162	000000						.WORD	T\$LOLIM	
	034164	000077						.WORD	TSHILIM	
5	034166				GPRMD	H.LDR,HO.LDR,D,-1,0.,255.,YES		; DRIVE SELECT NUMBER		
	034166	004052						.WORD	TSCODE	
	034170	034276						.WORD	H.LDR	
	034172	177777						.WORD	-1	
	034174	000000						.WORD	T\$LOLIM	
	034176	000377						.WORD	TSHILIM	
7	034200				GPRML	H.CST,HO.PRM,HM.CYL,YES ; USE CUSTOMER DATA AREA				
	034200	005130						.WORD	TSCODE	
	034202	034313						.WORD	H.CST	
	034204	020000						.WORD	HM.CYL	
9	034206				ENDHRD					
	034206							.EVEN		
								L10062:		
10										
11	034206	125	116	111	H.UBA:	.ASCIZ	\UNIBUS ADDRESS OF UDA\			
12	034234	126	105	103	H.VEC:	.ASCIZ	\VECTOR\			
13	034243	102	122	040	H.BRL:	.ASCIZ	\BR LEVEL\			
14	034254	125	116	111	H.BST:	.ASCIZ	\UNIBUS BURST RATE\			
15	034276	104	122	111	H.LDR:	.ASCIZ	\DRIVE NUMBER\			
17	034313	105	130	105	H.CST:	.ASCIZ	\EXERCISE ON CUSTOMER DATA AREA IN TEST 4\			
19						.EVEN				

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
28
29

.SBTTL SOFTWARE PARAMETER CODING SECTION

;++
: THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
: THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
: MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
: INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
: MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
: WITH THE OPERATOR.
:--

034364
034364 000030
034366

BGNSFT

.WORD L10063-L\$SOFT/2
L\$SOFT::

:FORMAT OF SOFTWARE P-TABLE IS AS FOLLOWS:

TABLE

:START A TABLE DEFINITION

ITEM SO.EL 2
ITEM SO.XL 2
ITEM SO.BIT 2
SM.MAN==BIT07
SM.SSF==BIT08
SM.LOG==BIT09
SM.IW== BIT14

:ERROR LIMIT
:DATA TRANSFER LIMIT (MEGABITS)
:SINGLE BIT ANSWERS
: MANUAL INTERVENTION MODE
: SUPPRESS SOFT ERRORS
: ERROR LOG ENABLED
: INITIAL WRITE

000200
000400
001000
040000

034366

END

1
2
3
4
5
6
7
8
9
10

035026 000050
035146
035146 035172
035150 000010
035152
035152

.SBTTL PATCH AREA
SPATCH::
.REPT 40.
.WORD 0
.ENDR
LASTAD
L\$LAST::
ENDMOD

.EVEN
.WORD T\$FREE
.WORD T\$SIZE

```
1 035152          BGNSETUP          1
2
3 035152          BGNPTAB
  035152 000000
  035154 000006
  035156
4
5 035156 172150   .WORD 172150
6 035160 000154   .WORD 154
7 035162 000005   .WORD 5.
8 035164 000077   .WORD 63.
9 035166 000000   .WORD 0.
10 035170 000000  .WORD 0
11
12 035172          ENDPTAB
  035172
13
14 035172          ENDSETUP
15
16
17
18
19
20
21
22          000001          .END
```

L10064: .WORD 0
.WORD L10066-./2-1

: UNIBUS ADDRESS
: VECTOR ADDRESS
: BR LEVEL
: UNIBUS BURST RATE
: LOGICAL DRIVE NUMBER
: CUSTOMER DATA AREA

L10066:

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 29952 WORDS (117 PAGES)
DYNAMIC MEMORY AVAILABLE FOR 70 PAGES
ZUDCBO.BIC,A:ZUDCBO/C=[20,0]SVC34R.MLB/P:1,ZUDCBO.DOC,ZUDCBO.MAC

LSDESC	112-42	135-17#				
LSDESP	112-42#					
LSDEVP	112-42#					
LSDISP	112-42	113-9#				
LSDLY	112-42#					
LSDTP	112-42#					
LSDTYP	112-42#					
LSDUT	112-42#					
LSDVTY	112-42	135-12#				
LSEF	112-42#					
LSEVI	112-42#					
LSERRT	112-42	132-8#				
LSETP	112-42#					
LSEXP1	112-42#					
LSEXP4	112-42#					
LSEXP5	112-42#					
LSHARD	112-42	287-14	287-14#			
LSHIME	112-42#	157-5				
LSHPCP	112-42#					
LSHPTP	112-42#					
LSHW	112-42	114-10	114-10#			
LSICP	112-42#					
LSINIT	112-42	246-8#				
LSLADP	112-42#					
LSLAST	112-42	291-8#	292-14			
LSLOAD	112-42#					
LSLUN	112-42#	148-28*	149-11*	195-29*	207-16*	260-11*
LSMREV	112-42#					
LSNAME	112-42#					
LSPRIO	112-42#					
LSPROT	112-42	245-8#				
LSPRT	112-42#					
LSREPP	112-42#					
LSREV	112-42#					
LSRPT	112-42	241-10#				
LSSOFT	112-42	289-12	289-12#			
LSSPC	112-42#					
LSSPCP	112-42#					
LSSPTP	112-42#					
LSSTA	112-42#					
LSSW	112-42	115-10	115-10#			
LSTEST	112-42#					
LSTIML	112-42#					
LSUNIT	112-42#	248-37	252-16	255-4		
L10000	114-10	114-18#				
L10001	115-10	115-19#				
L10002	140-16#					
L10003	140-20#					
L10004	140-24#					
L10005	140-28#					
L10006	140-32#					
L10007	140-36#					
L10010	140-40#					
L10011	140-44#					
L10012	140-48#					
L10013	140-57#					

L10014	140-63#						
L10015	140-77#						
L10016	140-81#						
L10017	140-85#						
L10020	140-89#						
L10021	140-93#						
L10022	140-97#						
L10023	140-101#						
L10024	140-105#						
L10025	140-109#						
L10026	140-114#						
L10027	140-118#						
L10030	140-122#						
L10031	140-127#						
L10032	140-131#						
L10033	140-135#						
L10034	140-139#						
L10035	141-22#						
L10036	219-14#						
L10037	220-21#						
L10040	238-9#						
L10041	238-13#						
L10042	244-12	244-43#					
L10044	256-31	257-23#					
L10045	258-12#						
L10046	259-21#						
L10047	260-18	269-14#					
L10050	261-14#						
L10051	262-15#						
L10052	263-32#						
L10053	266-11#						
L10054	267-4#						
L10055	268-6#						
L10056	269-10#						
L10057	271-24#						
L10060	272-14#						
L10061	276-44	286-7#					
L10062	287-14	288-9#					
L10063	289-12	290-15#					
L10064	292-3#						
L10066	292-3	292-12#					
LBUFE	134-22#	176-8*	176-11				
LBUFN	134-21#	176-5*	176-9	176-10*	176-11	176-19*	273-21
LBUFS	134-20#	146-15*	176-1	176-4*	273-14	273-17*	
LDDM	148-22#	148-37					
LDNEXT	148-30	148-33	148-35#				
LOAD	210-23	211-15#					
LOADB	209-6#	210-31					
LOADDM	148-32	208-14#					
LOADE1	209-18	211-26	212-3#				
LOADER	208-37	210-24	211-24	212-4#			
LOADT1	209-3	210-4#					
LOE	116-10#						
LOG	120-32#						
LOGCHK	273-15	273-26#					
LOGM1	137-38#	273-16					

PNTERR	175-70	207-13#	273-19						
PNTNUM	182-24	182-30	183-4	197-14#					
PNTNUS	187-11	197-16#							
PNTPKL	140-147#	140-151							
PNTPKT	140-117	140-121	140-141#						
PORTST	263-2#								
PRI	116-10#								
PRI00	116-10#	256-17	264-19	264-20	266-1				
PRI01	116-10#								
PRI02	116-10#								
PRI03	116-10#								
PRI04	116-10#								
PRI05	116-10#								
PRI06	116-10#								
PRI07	112-42	116-10#	208-32	225-24	261-3	265-10			
PS	205-18#	206-11							
PTYPE	132-10#	141-20*	189-8*	190-8*	205-11	206-5*	206-7*	206-9*	206-11*
PX	141-20	189-8	190-8	205-16#	206-9				
RDDAT	233-34	234-2	235-1#						
RDDL	159-7	231-11#							
RDERR	234-3	237-1#							
RDREC	147-15	231-12	233-18#						
RDST	233-24#	233-28	234-14	234-16	234-18	235-33	236-6		
RDSTS	233-20	233-23#							
READD	146-24	147-14#							
README	147-16	147-21#							
RESET	146-14	239-10#	247-10	259-17	260-20	271-10			
RESETX	239-13	239-15#							
RESPCT	149-8#	150-16							
RESPDM	149-6#	150-17	159-19	269-7	271-16	272-12	273-31	276-37	
RG.FLG	120-4#	214-24							
RG.OWN	120-3#	214-24	214-25						
RNTIM	137-5#	240-24							
RNTIM1	137-6#	240-28							
RNTIM2	137-7#	240-33							
RNTIME	141-10	173-55	179-26	189-6	190-6	240-13#	241-14		
RNTIMX	240-14	240-35#							
RPTCT	242-10#	244-9							
RPTCTN	242-13	242-24	244-3#						
RPTDT	242-23#	244-2							
RPTDTN	242-27	244-1#							
RPTMD2	243-24	244-35#							
RPTMSD	243-22	244-34#							
RPTMSG	241-13	244-31#							
RPTMSH	242-8	244-32#							
RPTXX	242-7	244-11#							
RSPDRP	149-23	149-41	150-21#	151-22	152-12	152-24			
RSPDSP	152-23	153-3#	153-19						
RSPERR	151-10	151-15	151-21#						
RSPIN	149-13	151-5#							
RSPMWR	151-7	151-9#							
RSPNRP	150-4	150-7	150-9	150-14#					
RSPNTO	149-34	149-37	149-39	149-42#					
RSPNXT	149-10	150-3#	150-23	152-47					
RSPQU	149-15	151-27#							
RSPQU2	152-31	152-40#							

	284-20	284-20	284-20	284-20	284-20	284-20	284-20	284-20	284-20	284-20	284-20	284-20	285-2	285-2
	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2
	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2	285-2
	285-8	285-8	285-8	285-8	285-8	285-8	285-8	285-8	285-8	285-8	285-8	285-8	285-8	285-8
	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22
	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22
	286-2	286-2	286-2	286-2	286-2	286-2	286-2	286-2	286-2	286-2	286-2	286-2	286-2	286-2
	286-2	286-2	286-2	286-2	286-7	286-7	286-7	286-7	287-14	287-14	287-14	288-1	288-1	288-1
	288-1	288-1	288-1	288-1	288-1	288-1	288-1	288-1	288-2	288-2	288-2	288-2	288-2	288-2
	288-2	288-2	288-2	288-2	288-2	288-2	288-3	288-3	288-3	288-3	288-3	288-3	288-3	288-3
	288-3	288-3	288-3	288-3	288-3	288-3	288-3	288-4	288-4	288-4	288-4	288-4	288-4	288-4
	288-4	288-4	288-4	288-4	288-4	288-4	288-4	288-4	288-5	288-5	288-5	288-5	288-5	288-5
	288-5	288-5	288-5	288-5	288-5	288-5	288-5	288-5	288-7	288-7	288-7	288-7	288-7	288-7
	288-7	288-7	288-7	288-7	288-9	288-9	288-9	289-12	289-12	289-12	290-1	290-1	290-1	290-1
	290-1	290-1	290-1	290-1	290-1	290-4	290-4	290-4	290-4	290-4	290-4	290-6	290-6	290-6
	290-6	290-6	290-6	290-6	290-6	290-6	290-6	290-6	290-6	290-6	290-6	290-6	290-7	290-7
	290-7	290-7	290-7	290-7	290-7	290-7	290-7	290-7	290-7	290-7	290-7	290-7	290-7	290-8
	290-8	290-8	290-8	290-8	290-8	290-8	290-8	290-8	290-9	290-9	290-9	290-9	290-9	290-9
	290-9	290-9	290-9	290-10	290-10	290-10	290-10	290-10	290-10	290-10	290-10	290-10	290-15	290-15
	290-15	291-8	291-8	291-8	291-8	291-8	291-8	291-8	291-8	291-8	292-3	292-3	292-3	292-3
	292-3	292-3												
SVCSUB	112-12#	112-16#	140-10#	142-3#	261-1	262-1	263-1	264-1	267-1	268-1	269-1			
SVCTAG	112-12#	112-18#	114-18	115-19	140-12#	140-16	140-16	140-16	140-20	140-20	140-20	140-24	140-24	140-24
	140-28	140-28	140-28	140-32	140-32	140-32	140-36	140-36	140-36	140-40	140-40	140-40	140-44	140-44
	140-44	140-48	140-48	140-48	140-57	140-57	140-57	140-63	140-63	140-63	140-77	140-77	140-77	140-81
	140-81	140-81	140-85	140-85	140-85	140-89	140-89	140-89	140-93	140-93	140-93	140-97	140-97	140-97
	140-101	140-101	140-101	140-105	140-105	140-105	140-109	140-109	140-109	140-114	140-114	140-114	140-118	140-118
	140-118	140-122	140-122	140-122	140-127	140-127	140-127	140-131	140-131	140-131	140-135	140-135	140-135	140-139
	140-139	140-139	141-22	141-22	141-22	142-5#	163-3	219-14	220-21	238-9	238-13	244-43	256-6	257-23
	258-12	259-21	261-14	262-15	263-32	266-11	267-4	268-6	269-10	269-14	271-24	272-14	275-3	275-7
	277-14	278-2	278-8	278-14	278-20	278-21	278-24	279-4	279-13	279-14	279-17	280-2	280-4	281-2
	283-4	283-10	283-14	284-10	284-14	284-20	285-2	285-8	285-22	286-2	286-7	288-9	290-15	292-3
	292-12													
SVCTST	112-12#	112-15#	140-9#	142-2#	260-6	271-3	272-3	273-3						
TSSAUT	258-10#	258-12												
TSSCLE	259-8#	259-21												
TSSDAT	292-3	292-3#	292-12											
TSSHAR	287-14	287-14#	288-9											
TSSHW	114-10	114-10#	114-18											
TSSINI	246-8#	256-31	257-23											
TSSMSG	140-14#	140-16	140-18#	140-20	140-22#	140-24	140-26#	140-28	140-30#	140-32	140-34#	140-36	140-38#	140-40
	140-42#	140-44	140-46#	140-48	140-50#	140-57	140-59#	140-63	140-65#	140-77	140-79#	140-81	140-83#	140-85
	140-87#	140-89	140-91#	140-93	140-95#	140-97	140-99#	140-101	140-103#	140-105	140-107#	140-109	140-111#	140-114
	140-116#	140-118	140-120#	140-122	140-124#	140-127	140-129#	140-131	140-133#	140-135	140-137#	140-139	141-1#	141-22
TSSPC	292-1#	292-14												
TSSPRO	245-8#													
TSSPTA	292-1#	292-3	292-3#											
TSSRPT	241-10#	244-12	244-43											
TSSSOF	289-12	289-12#	290-15											
TSSSRV	219-10#	219-14	220-18#	220-21	238-5#	238-9	238-11#	238-13						
TSSSUB	261-1#	261-14	262-1#	262-15	263-1#	263-32	264-1#	266-11	267-1#	267-4	268-1#	268-6	269-1#	269-10
TSSSW	115-10	115-10#	115-19											
TSSTES	260-6#	260-18	269-14	271-3#	271-24	272-3#	272-14	273-3#	276-44	286-7				
TSARGC	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42#	112-42#	112-42#
	112-42#	112-42#	112-42#	140-62	140-62	140-62#	205-12	205-12	205-12	205-12#	205-12#	205-14	205-14	205-14
	205-14#	205-14#	205-16	205-16	205-16	205-16#	205-16#	205-18	205-18	205-18	205-18#	205-18#	243-22	243-22

	243-22	243-22	243-22	243-22	243-22	243-22	243-22#	243-22#	243-22#	243-22#	243-22#	243-22#	243-22#	243-24	
	243-24	243-24	243-24	243-24	243-24#	243-24#	243-24#	243-24#	255-36	255-36	255-36	255-36	255-36	255-36#	
TSCODE	163-3	163-3	163-3	163-3#	163-3#	163-3#	256-6	256-6	256-6	256-6#	256-6#	256-6#	256-6#	275-3	275-3
	275-3	275-3#	275-3#	275-3#	275-7	275-7	275-7	275-7#	275-7#	275-7#	275-7#	277-14	277-14	277-14	277-14#
	277-14#	277-14#	278-2	278-2	278-2	278-2#	278-2#	278-2#	278-8	278-8	278-8	278-8	278-8#	278-8#	278-8#
	278-14	278-14	278-14	278-14#	278-14#	278-14#	278-20	278-20	278-20	278-20#	278-20#	278-20#	278-20#	278-21	278-21
	278-21	278-21#	278-21#	278-21#	278-24	278-24	278-24	278-24#	278-24#	278-24#	278-24#	279-4	279-4	279-4	279-4#
	279-4#	279-4#	279-13	279-13	279-13	279-13#	279-13#	279-13#	279-14	279-14	279-14	279-14#	279-14#	279-14#	279-14#
	279-17	279-17	279-17	279-17#	279-17#	279-17#	280-2	280-2	280-2	280-2#	280-2#	280-2#	280-2#	280-4	280-4
	280-4	280-4#	280-4#	280-4#	281-2	281-2	281-2	281-2#	281-2#	281-2#	281-2#	283-4	283-4	283-4	283-4#
	283-4#	283-4#	283-10	283-10	283-10	283-10#	283-10#	283-10#	283-14	283-14	283-14	283-14#	283-14#	283-14#	283-14#
	284-10	284-10	284-10	284-10#	284-10#	284-10#	284-14	284-14	284-14	284-14#	284-14#	284-14#	284-14#	284-20	284-20
	284-20	284-20#	284-20#	284-20#	285-2	285-2	285-2	285-2#	285-2#	285-2#	285-2#	285-8	285-8	285-8	285-8#
	285-8#	285-8#	285-22	285-22	285-22	285-22#	285-22#	285-22#	286-2	286-2	286-2	286-2#	286-2#	286-2#	286-2#
	288-1	288-1	288-1	288-1#	288-1#	288-1#	288-2	288-2	288-2	288-2#	288-2#	288-2#	288-2#	288-3	288-3
	288-3	288-3#	288-3#	288-3#	288-4	288-4	288-4	288-4#	288-4#	288-4#	288-4#	288-5	288-5	288-5	288-5#
	288-5#	288-5#	288-7	288-7	288-7	288-7#	288-7#	288-7#	290-1	290-1	290-1	290-1#	290-1#	290-1#	290-1#
	290-4	290-4#	290-6	290-6	290-6	290-6#	290-6#	290-6#	290-7	290-7	290-7	290-7#	290-7#	290-7#	290-7#
	290-8	290-8	290-8	290-8#	290-8#	290-8#	290-9	290-9	290-9	290-9#	290-9#	290-9#	290-9#	290-10	290-10
	290-10	290-10#	290-10#	290-10#											
TSERRN	112-12#	143-7	143-7#	147-21	147-21#	149-22	149-22#	149-40	149-40#	151-21	151-21#	152-11	152-11#	195-24	
	195-24#	212-3	212-3#	216-34	216-34#	217-5	217-5#	223-18	223-18#	223-40	223-40#	225-30	225-30#	227-44	
	227-44#	228-36	228-36#	229-5	229-5#	237-6	237-6#	248-29	248-29#	257-3	257-3#	257-8	257-8#	257-14	
	257-14#	257-20	257-20#	261-10	261-10#	263-26	263-26#	265-27	265-27#	266-5	266-5#	270-19	270-19#		
TSEXCP	163-3	163-3#	275-3	275-3#	275-7	275-7#	277-14	277-14#	278-2	278-2#	279-4	279-4#	281-2	281-2#	
	283-4	283-4#	283-10	283-10#	283-14	283-14#	284-10	284-10#	284-14	284-14#	284-20	284-20#	285-2	285-2#	
	285-22	285-22#	286-2	286-2#	288-1	288-1#	288-2	288-2#	288-3	288-3#	288-4	288-4#	288-5	288-5#	
	290-6	290-6#	290-7	290-7#											
TSFLAG	244-12	244-12#	244-12#	256-31	256-31	256-31#	256-31#	260-18	260-18	260-18#	260-18#	276-44	276-44	276-44#	
	276-44#														
TSFREE	291-8	292-14#													
TSGMAN	112-12#	163-3	163-3#	163-3#	275-3#	275-3#	275-7#	275-7#	277-14#	277-14#	278-2	278-2#	278-2#	279-4#	
	279-4#	281-2#	281-2#	283-4#	283-4#	283-10	283-10#	283-10#	283-14	283-14#	283-14#	284-10#	284-10#	284-14#	
	284-14#	284-20#	284-20#	285-2#	285-2#	285-22	285-22#	285-22#	286-2	286-2#	286-2#				
TSHILI	163-3	163-3#	275-3	275-3#	275-7	275-7#	277-14	277-14#	278-2	278-2#	279-4	279-4#	281-2	281-2#	
	283-4	283-4#	283-10	283-10#	283-14	283-14#	284-10	284-10#	284-14	284-14#	284-20	284-20#	285-2	285-2#	
	285-22	285-22#	286-2	286-2#	288-1	288-1#	288-2	288-2#	288-3	288-3#	288-4	288-4#	288-5	288-5#	
	290-6	290-6#	290-7	290-7#											
TSLAST	112-12#	291-8#	292-1												
TSLOLI	163-3	163-3#	275-3	275-3#	275-7	275-7#	277-14	277-14#	278-2	278-2#	279-4	279-4#	281-2	281-2#	
	283-4	283-4#	283-10	283-10#	283-14	283-14#	284-10	284-10#	284-14	284-14#	284-20	284-20#	285-2	285-2#	
	285-22	285-22#	286-2	286-2#	288-1	288-1#	288-2	288-2#	288-3	288-3#	288-4	288-4#	288-5	288-5#	
	290-6	290-6#	290-7	290-7#											
TSLSYM	112-12	112-12#	114-18	115-19	140-16	140-20	140-24	140-28	140-32	140-36	140-40	140-44	140-48	140-57	
	140-63	140-77	140-81	140-85	140-89	140-93	140-97	140-101	140-105	140-109	140-114	140-118	140-122	140-127	
	140-131	140-135	140-139	141-22	219-14	220-21	238-9	238-13	244-43	257-23	258-12	259-21	261-14	262-15	
	263-32	266-11	267-4	268-6	269-10	269-14	271-24	272-14	286-7	288-9	290-15				
TSLTNO	291-8#														
TSNEST	112-12#	112-33	112-33	112-33#	114-10	114-10	114-10#	114-18	114-18	114-18	114-18#	115-10	115-10	115-10#	
	115-19	115-19	115-19	115-19#	115-21	115-21	115-21	115-21#	116-3	116-3	116-3#	140-14	140-14	140-14#	
	140-16	140-16	140-16	140-16#	140-18	140-18	140-18#	140-20	140-20	140-20	140-20#	140-22	140-22	140-22#	
	140-24	140-24	140-24	140-24#	140-26	140-26	140-26#	140-28	140-28	140-28	140-28#	140-30	140-30	140-30#	
	140-32	140-32	140-32	140-32#	140-34	140-34	140-34#	140-36	140-36	140-36	140-36#	140-38	140-38	140-38#	
	140-40	140-40	140-40	140-40#	140-42	140-42	140-42#	140-44	140-44	140-44	140-44#	140-46	140-46	140-46#	
	140-48	140-48	140-48	140-48#	140-50	140-50	140-50#	140-57	140-57	140-57	140-57#	140-59	140-59	140-59#	

T2CMD0	160-33	160-36#							
T2CMD2	160-38	162-1#							
T2CMD3	163-9	164-3#							
T2CMD9	160-34#								
T2CMDE	163-16	164-7	164-10	164-12	164-27	164-32#	204-47		
T2CMDM	160-30	160-32#							
T2CMDN	164-17	164-19#							
T2CMDQ	162-2	163-3#	164-33						
T2CMDR	164-5	164-14#							
T2CMDV	163-7	163-15#							
T2CMDW	164-20	164-24	164-26#						
T2CMDX	160-31	160-35	163-17	164-30#					
T2CMS1	137-27#	161-1							
T2CMS5	137-36#	164-32							
T2DLL	153-4	158-37#							
T2DR	134-15#	161-4*	163-13*	163-14					
T2GND1	204-16	204-19#							
T2GND2	204-20#	204-41							
T2GND3	204-25	204-31#							
T2GNE	204-23	204-27	204-29	204-45#					
T2GNUM	163-11	163-15	164-9	164-19	164-23	164-26	204-12#	204-18	
T2GNX	204-14	204-39	204-42#						
T2NEXT	271-10#	271-22							
T2PNT	203-6	203-8	203-15#						
T2PNTB	162-10	203-11#							
T2PNTD	203-25	203-27#							
T2PNTQ	203-20	203-22#							
T2PNTW	162-5	162-7	203-3#						
T2WARN	137-26#	160-34							
T2WRO	134-14#	161-3*	162-6	164-15	164-29*				
T2WRR	134-13#	161-2*	162-4	164-14	164-28*				
T3	113-9	272-3#							
T4	113-9	273-3#							
T4BB	136-3#	277-14							
T4BB1	153-8	169-22#							
T4BB1E	169-24	169-30#							
T4BB2	153-9	170-15#							
T4BB2E	170-17	170-23#							
T4BBI	136-5#	278-2							
T4BE	136-17#	283-4							
T4BEG	136-18#	283-10							
T4CON	273-24	273-27	273-29#						
T4CYL	136-24#	285-8							
T4CYLB	136-25#	285-22							
T4CYLE	136-26#	286-2							
T4DCA	136-12#	279-14							
T4DCR	136-13#	279-17							
T4DEF	274-6	276-3#							
T4DEFA	276-4#	276-20							
T4DEFB	276-7#	276-17							
T4DEFC	276-13#	276-15							
T4DEFD	276-16#								
T4DEFE	276-8	276-18#							
T4DEFW	274-8	275-15#							
T4DMN	136-4#	278-8							
T4DP	136-10#	279-4							

T4DPC	136-27#	275-3			
T4DPD	136-28#	275-7			
T4ECC	136-11#	279-13			
T4END	136-19#	283-14			
T4EXIT	276-39	276-43#			
T4GRC	136-22#	284-20			
T4GRP	136-23#	285-2			
T4MPRM	153-6	165-18#			
T4MXFR	153-12	173-18#			
T4OPT1	137-17#	281-1			
T4OPT7	136-16#	163-3	281-2		
T4PRM1	274-14#	274-29			
T4PRM2	274-17#	274-26			
T4PRM3	274-20	274-23	274-25#		
T4PRM4	274-18	274-27#			
T4PRM5	275-6#	275-10			
T4Q01	278-1#	278-4	278-6		
T4Q02	277-16	278-7#			
T4Q03	278-16	278-19#			
T4Q04	278-23	279-1#			
T4Q05	278-18	278-25	279-2#		
T4Q06	279-7#				
T4Q07	279-8	279-10	279-13#		
T4Q08	279-16	280-1#			
T4Q09	279-12	279-18	280-2#		
T4Q10	280-10	280-13#			
T4Q11	280-12	280-14	280-17	280-20	281-1#
T4Q12	281-4	282-1#			
T4Q13	282-2	282-8#			
T4Q14	282-11	282-15#			
T4Q15	282-18	282-20#			
T4Q16	282-7	282-21	282-24#		
T4Q17	282-25#				
T4Q18	282-28#	282-30			
T4Q19	281-6	282-4	282-14	282-23	283-1#
T4Q20	283-9#	283-12	283-18		
T4Q21	283-13#	283-16			
T4Q22	283-2	284-1#			
T4Q23	284-13#	284-17			
T4Q24	284-8	284-19#			
T4Q25	285-1#	285-5			
T4Q26	284-18	285-6#			
T4Q27	284-4	285-10	285-16#		
T4Q27A	285-17	285-19#			
T4Q28	285-21#	285-24			
T4Q29	286-1#	286-4			
T4Q30	278-11	283-19	284-2	285-15	286-5#
T4QHED	137-16#	277-12			
T4QUES	274-21	277-11#			
T4RET	136-14#	280-2			
T4RO	136-6#	278-14			
T4RUN	274-4	275-11	276-24#		
T4SEEK	153-11	172-1#			
T4SEK	136-15#	280-4			
T4SOFT	153-10	171-16#			
T4STRT	273-10	273-12	273-30	274-3#	

X30	138-42#	140-104		
X31	138-43#	140-108		
X32	138-45#	140-112		
X35	138-46#	140-125		
X36	138-47#	140-130		
X37	138-49#	140-134		
X38	138-15#	140-138		
X3A	138-3#	140-23		
X4	138-8#	140-27		
X5	138-10#	140-31		
X6	138-11#	140-35		
X7	138-12#	140-39		
X8	138-13#	140-43		
X8A	138-4#	140-43		
XFRU	139-8#	140-62	140-76	192-5
XMSG1	139-1#	140-154		
XMSG2	139-2#	140-158		
XPKT1	139-3#	140-141		
XPKT2	139-6#	140-147		
XSA	139-7#	193-5		

ERRDF	149-22	149-40	151-21	152-11	195-24	212-3	216-34	217-5	223-18	223-40	225-30	227-44	228-36	229-5
	261-10	263-26	265-27	266-5	270-19									
ERROR	207-41													
ERRSF	143-7	147-21	237-6	248-29	257-3	257-8	257-14	257-20						
ERRTBL	132-8													
EXIT	244-12	256-31	260-18	276-44										
GETBYT	233-26	233-31	234-7	235-23	235-30	236-1	236-11	236-14						
GMANID	163-3	275-3	275-7	277-14	278-2	279-4	281-2	283-4	283-10	283-14	284-10	284-14	284-20	285-2
	285-22	286-2												
GMANIL	256-6	278-8	278-14	278-20	278-21	278-24	279-13	279-14	279-17	280-2	280-4	285-8		
GPHARD	248-21	250-10	253-5											
GPRMA	288-1	288-2												
GPRMD	163-3	163-3#	275-3	275-3#	275-7	275-7#	277-14	277-14#	278-2	278-2#	279-4	279-4#	281-2	281-2#
	283-4	283-4#	283-10	283-10#	283-14	283-14#	284-10	284-10#	284-14	284-14#	284-20	284-20#	285-2	285-2#
	285-22	285-22#	286-2	286-2#	288-3	288-4	288-5	290-6	290-7					
GPRML	256-6	256-6#	278-8	278-8#	278-14	278-14#	278-20	278-20#	278-21	278-21#	278-24	278-24#	279-13	279-13#
	279-14	279-14#	279-17	279-17#	280-2	280-2#	280-4	280-4#	285-8	285-8#	288-7	290-1	290-8	290-9
	290-10													
HEADER	112-42													
ITEM	117-24#	127-12	127-13	127-16	127-19	127-20	127-21	127-22	127-33	127-34	127-35	127-36	127-37	127-38
	127-39	127-40	127-41	127-42	127-43	127-44	128-9	128-10	128-13	128-34	128-35	128-36	128-37	128-38
	128-39	128-40	128-41	128-42	128-43	128-44	128-45	128-46	128-47	128-48	128-49	128-50	128-51	129-1
	129-2	129-3	129-4	129-5	129-6	129-7	129-8	129-9	129-10	129-11	129-12	129-13	129-14	129-15
	129-16	129-17	129-18	287-20	287-21	287-22	287-23	287-24	287-25	289-18	289-19	289-20		
LASTAD	291-8													
MSBYTE	112-42	112-42	112-42	112-42#										
MSCHEC	244-12	244-12#	256-31	256-31#	260-18	260-18#	276-44	276-44#						
MSCNTO	163-3	163-3#	256-6	256-6#	275-3	275-3#	275-7	275-7#	277-14	277-14#	278-2	278-2#	278-8	278-8#
	278-14	278-14#	278-20	278-20#	278-21	278-21#	278-24	278-24#	279-4	279-4#	279-13	279-13#	279-14	279-14#
	279-17	279-17#	280-2	280-2#	280-4	280-4#	281-2	281-2#	283-4	283-4#	283-10	283-10#	283-14	283-14#
	284-10	284-10#	284-14	284-14#	284-20	284-20#	285-2	285-2#	285-8	285-8#	285-22	285-22#	286-2	286-2#
	288-1	288-1#	288-2	288-2#	288-3	288-3#	288-4	288-4#	288-5	288-5#	288-7	288-7#	290-1	290-1#
	290-6	290-6#	290-7	290-7#	290-8	290-8#	290-9	290-9#	290-10	290-10#				
MSCOUN	140-62	140-62#	205-12	205-12#	205-14	205-14#	205-16	205-16#	205-18	205-18#	243-22	243-22	243-22	243-22
	243-22	243-22	243-22#	243-24	243-24	243-24#	255-36	255-36	255-36	255-36#	255-36#	255-36#	255-36#	255-36#
MSDATA	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42
	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42
	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42	112-42
	135-17#													
MSDECR	114-18	114-18#	115-19	115-19#	115-21	115-21#	140-16	140-16#	140-20	140-20#	140-24	140-24#	140-28	140-28#
	140-32	140-32#	140-36	140-36#	140-40	140-40#	140-44	140-44#	140-48	140-48#	140-57	140-57#	140-63	140-63#
	140-77	140-77#	140-81	140-81#	140-85	140-85#	140-89	140-89#	140-93	140-93#	140-97	140-97#	140-101	140-101#
	140-105	140-105#	140-109	140-109#	140-114	140-114#	140-118	140-118#	140-122	140-122#	140-127	140-127#	140-131	140-131#
	140-135	140-135#	140-139	140-139#	141-22	141-22#	219-14	219-14#	220-21	220-21#	238-9	238-9#	238-13	238-13#
	240-38	240-38#	244-43	244-43#	245-14	245-14#	257-23	257-23#	258-12	258-12#	259-21	259-21#	259-23	259-23#
	261-14	261-14#	262-15	262-15#	263-32	263-32#	266-11	266-11#	267-4	267-4#	268-6	268-6#	269-10	269-10#
	269-14	269-14#	271-24	271-24#	272-14	272-14#	286-7	286-7#	286-9	286-9#	288-9	288-9#	290-15	290-15#
	291-10	291-10#	292-3	292-3#										
MSDEFA	163-3	163-3#	256-6	256-6#	275-3	275-3#	275-7	275-7#	277-14	277-14#	278-2	278-2#	278-8	278-8#
	278-14	278-14#	278-20	278-20#	278-21	278-21#	278-24	278-24#	279-4	279-4#	279-13	279-13#	279-14	279-14#
	279-17	279-17#	280-2	280-2#	280-4	280-4#	281-2	281-2#	283-4	283-4#	283-10	283-10#	283-14	283-14#
	284-10	284-10#	284-14	284-14#	284-20	284-20#	285-2	285-2#	285-8	285-8#	285-22	285-22#	286-2	286-2#
	288-1	288-1#	288-2	288-2#	288-3	288-3#	288-4	288-4#	288-5	288-5#	288-7	288-7#	290-1	290-1#
	290-6	290-6#	290-7	290-7#	290-8	290-8#	290-9	290-9#	290-10	290-10#				
MSENDE	114-18#	115-19#	115-21#	140-16#	140-20#	140-24#	140-28#	140-32#	140-36#	140-40#	140-44#	140-48#	140-57#	140-63#
	140-77#	140-81#	140-85#	140-89#	140-93#	140-97#	140-101#	140-105#	140-109#	140-114#	140-118#	140-122#	140-127#	140-131#

	255-36#	255-36#	255-36#	255-36#	255-36#	255-36#	255-36#	255-36#	256-4	256-4#	256-5	256-5#	256-6	256-6	256-6
	256-6	256-6	256-6	256-6#	256-6#	256-6#	256-6#	256-6#	256-9	256-9#	256-17	256-17#	256-17#	256-17#	256-20
	256-20	256-20#	256-20#	256-21	256-21#	256-22	256-22	256-22#	256-22#	256-22#	256-23	256-23#	256-31	256-31	256-31#
	256-31#	257-3	257-3	257-3	257-3	257-3#	257-3#	257-3#	257-3#	257-3#	257-3#	257-4	257-4#	257-8	257-8
	257-8	257-8	257-8#	257-8#	257-8#	257-8#	257-8#	257-8#	257-9	257-9#	257-14	257-14	257-14	257-14	257-14#
	257-14#	257-14#	257-14#	257-14#	257-15	257-15#	257-20	257-20	257-20	257-20	257-20	257-20#	257-20#	257-20#	257-20#
	257-20#	257-21	257-21#	257-23	257-23#	258-12	258-12#	259-21	259-21#	259-21#	260-18	260-18	260-18#	260-18#	261-1
	261-1#	261-3	261-3	261-3	261-3	261-3	261-3	261-3#	261-3#	261-3#	261-3#	261-3#	261-3#	261-3#	261-7
	261-7	261-7#	261-7#	261-10	261-10	261-10	261-10	261-10#	261-10#	261-10#	261-10#	261-10#	261-10#	261-11	261-11#
	261-14	261-14#	262-1	262-1#	262-15	262-15#	263-1	263-1#	263-1#	263-26	263-26	263-26	263-26	263-26#	263-26#
	263-26#	263-26#	263-26#	263-32	263-32#	264-1	264-1#	264-19	264-19	264-19	264-19	264-19	264-19	264-19	264-19#
	264-19#	264-19#	264-19#	264-19#	264-19#	264-20	264-20	264-20#	264-20#	264-20#	264-35	264-35#	265-10	265-10	265-10#
	265-10#	265-21	265-21	265-21#	265-21#	265-27	265-27	265-27	265-27	265-27#	265-27#	265-27#	265-27#	265-27#	265-27#
	266-1	266-1	266-1#	266-1#	266-5	266-5	266-5	266-5	266-5#	266-5#	266-5#	266-5#	266-5#	266-5#	266-10
	266-10	266-10#	266-10#	266-11	266-11#	267-1	267-1#	267-4	267-4#	268-1	268-1#	268-6	268-6#	268-6#	269-1
	269-1#	269-10	269-10#	269-14	269-14#	270-11	270-11#	270-19	270-19	270-19	270-19	270-19#	270-19#	270-19#	270-19#
	270-19#	270-19#	271-24	271-24#	272-14	272-14#	274-7	274-7#	274-8	274-8#	275-3	275-3	275-3	275-3	275-3
	275-3	275-3	275-3	275-3	275-3#	275-3#	275-3#	275-3#	275-7	275-7	275-7	275-7	275-7	275-7	275-7
	275-7	275-7	275-7#	275-7#	275-7#	275-7#	276-40	276-40#	276-43	276-43#	276-44	276-44	276-44#	276-44#	276-44#
	277-14	277-14	277-14	277-14	277-14	277-14	277-14	277-14	277-14#	277-14#	277-14#	277-14#	277-14#	278-2	278-2
	278-2	278-2	278-2	278-2	278-2	278-2	278-2#	278-2#	278-2#	278-2#	278-8	278-8	278-8	278-8	278-8
	278-8	278-8	278-8#	278-8#	278-8#	278-8#	278-14	278-14	278-14	278-14	278-14	278-14	278-14	278-14#	278-14#
	278-14#	278-14#	278-20	278-20	278-20	278-20	278-20	278-20	278-20#	278-20#	278-20#	278-20#	278-20#	278-21	278-21
	278-21	278-21	278-21	278-21	278-21#	278-21#	278-21#	278-21#	278-24	278-24	278-24	278-24	278-24	278-24	278-24
	278-24#	278-24#	278-24#	278-24#	279-4	279-4	279-4	279-4	279-4	279-4	279-4	279-4	279-4	279-4#	279-4#
	279-4#	279-4#	279-13	279-13	279-13	279-13	279-13	279-13	279-13#	279-13#	279-13#	279-13#	279-13#	279-14	279-14
	279-14	279-14	279-14	279-14	279-14#	279-14#	279-14#	279-14#	279-17	279-17	279-17	279-17	279-17	279-17	279-17
	279-17#	279-17#	279-17#	279-17#	280-2	280-2	280-2	280-2	280-2	280-2	280-2#	280-2#	280-2#	280-2#	280-2#
	280-4	280-4	280-4	280-4	280-4	280-4	280-4#	280-4#	280-4#	280-4#	281-2	281-2	281-2	281-2	281-2
	281-2	281-2	281-2	281-2	281-2#	281-2#	281-2#	281-2#	283-4	283-4	283-4	283-4	283-4	283-4	283-4
	283-4	283-4	283-4#	283-4#	283-4#	283-4#	283-10	283-10	283-10	283-10	283-10	283-10	283-10	283-10	283-10
	283-10#	283-10#	283-10#	283-10#	283-14	283-14	283-14	283-14	283-14	283-14	283-14	283-14	283-14	283-14#	283-14#
	283-14#	283-14#	284-10	284-10	284-10	284-10	284-10	284-10	284-10	284-10	284-10#	284-10#	284-10#	284-10#	284-10#
	284-14	284-14	284-14	284-14	284-14	284-14	284-14	284-14	284-14#	284-14#	284-14#	284-14#	284-14#	284-20	284-20
	284-20	284-20	284-20	284-20	284-20	284-20	284-20#	284-20#	284-20#	284-20#	285-2	285-2	285-2	285-2	285-2
	285-2	285-2	285-2	285-2	285-2#	285-2#	285-2#	285-2#	285-8	285-8	285-8	285-8	285-8	285-8	285-8
	285-8#	285-8#	285-8#	285-8#	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22	285-22#	285-22#
	285-22#	285-22#	286-2	286-2	286-2	286-2	286-2	286-2	286-2	286-2	286-2#	286-2#	286-2#	286-2#	286-2#
	286-7	286-7#	287-14	287-14#	288-1	288-1	288-1	288-1	288-1#	288-2	288-2	288-2	288-2	288-2	288-2#
	288-3	288-3	288-3	288-3	288-3#	288-4	288-4	288-4	288-4	288-4	288-4	288-4#	288-5	288-5	288-5
	288-5	288-5	288-5	288-5#	288-7	288-7	288-7	288-7#	288-9	288-9#	289-12	289-12#	290-1	290-1	290-1
	290-1	290-1#	290-4	290-4	290-4#	290-4#	290-6	290-6	290-6	290-6	290-6	290-6#	290-7	290-7	290-7
	290-7	290-7	290-7	290-7#	290-8	290-8	290-8	290-8#	290-9	290-9	290-9	290-9#	290-10	290-10	290-10
	290-10	290-10#	290-15	290-15#	291-8	291-8	291-8	291-8#	292-3	292-3	292-3#	292-3#	292-3#	292-3#	292-3#
MSGNLS	163-3	163-3#	256-6	256-6#	275-3	275-3#	275-7	275-7#	277-14	277-14#	278-2	278-2#	278-8	278-8	278-8#
	278-14	278-14#	278-20	278-20#	278-21	278-21#	278-24	278-24#	279-4	279-4#	279-13	279-13#	279-14	279-14#	279-14#
	279-17	279-17#	280-2	280-2#	280-4	280-4#	281-2	281-2#	283-4	283-4#	283-10	283-10#	283-14	283-14#	283-14#
	284-10	284-10#	284-14	284-14#	284-20	284-20#	285-2	285-2#	285-8	285-8#	285-22	285-22#	286-2	286-2#	286-2#
MSGNSU	261-1	261-1#	262-1	262-1#	263-1	263-1#	264-1	264-1#	267-1	267-1#	268-1	268-1#	269-1	269-1#	269-1#
MSGNTA	114-18	114-18#	115-19	115-19#	140-16	140-16#	140-20	140-20#	140-24	140-24#	140-28	140-28#	140-32	140-32#	140-32#
	140-36	140-36#	140-40	140-40#	140-44	140-44#	140-48	140-48#	140-57	140-57#	140-63	140-63#	140-77	140-77#	140-77#
	140-81	140-81#	140-85	140-85#	140-89	140-89#	140-93	140-93#	140-97	140-97#	140-101	140-101#	140-105	140-105#	140-105#
	140-109	140-109#	140-114	140-114#	140-118	140-118#	140-122	140-122#	140-127	140-127#	140-131	140-131#	140-135	140-135#	140-135#
	140-139	140-139#	141-22	141-22#	219-14	219-14#	220-21	220-21#	238-9	238-9#	238-13	238-13#	244-43	244-43#	244-43#
	257-23	257-23#	258-12	258-12#	259-21	259-21#	261-14	261-14#	262-15	262-15#	263-32	263-32#	266-11	266-11#	266-11#
	267-4	267-4#	268-6	268-6#	269-10	269-10#	269-14	269-14#	271-24	271-24#	272-14	272-14#	286-7	286-7#	286-7#

MSGNTE	288-9	288-9#	290-15	290-15#	292-3	292-3#	292-12	292-12#						
MSHAPT	260-6	260-6#	271-3	271-3#	272-3	272-3#	273-3	273-3#						
MSHNAP	112-42	112-42#												
MSINCR	112-42	112-42#												
	112-33	112-33#	114-10	114-10	114-10#	114-10#	115-10	115-10	115-10#	115-10#	116-3	116-3#	140-14	140-14
	140-14#	140-14#	140-16#	140-18	140-18	140-18#	140-20#	140-22	140-22	140-22#	140-22#	140-22#	140-24#	140-26
	140-26	140-26#	140-26#	140-28#	140-30	140-30#	140-30#	140-32#	140-34	140-34#	140-34#	140-34#	140-34#	140-36#
	140-38	140-38#	140-38#	140-38#	140-40#	140-42	140-42#	140-42#	140-44#	140-46	140-46#	140-46#	140-46#	140-46#
	140-48#	140-50	140-50	140-50#	140-50#	140-57#	140-59	140-59#	140-59#	140-59#	140-62#	140-63#	140-65	140-65
	140-65#	140-65#	140-77#	140-79	140-79	140-79#	140-79#	140-81#	140-83	140-83#	140-83#	140-83#	140-85#	140-87
	140-87	140-87#	140-87#	140-89#	140-91	140-91	140-91#	140-91#	140-93#	140-95	140-95#	140-95#	140-95#	140-97#
	140-99	140-99	140-99#	140-99#	140-101#	140-103	140-103	140-103#	140-103#	140-105#	140-107	140-107	140-107#	140-107#
	140-109#	140-111	140-111	140-111#	140-111#	140-114#	140-116	140-116#	140-116#	140-118#	140-120	140-120	140-120	140-120#
	140-120#	140-122#	140-124	140-124	140-124#	140-124#	140-127#	140-129	140-129#	140-129#	140-131#	140-133	140-133	140-133
	140-133#	140-133#	140-135#	140-137	140-137	140-137#	140-137#	140-139#	141-1	141-1	141-1#	141-1#	141-22#	143-7#
	143-8#	147-21#	147-22#	149-22#	149-40#	149-43#	150-10#	151-21#	152-11#	160-32#	163-3	163-3#	163-3#	173-49#
	175-38#	177-44#	195-24#	205-12#	205-14#	205-16#	205-18#	207-41#	208-32#	212-3#	216-26#	216-34#	217-5#	219-10
	219-10	219-10#	219-10#	220-18	220-18	220-18#	220-18#	223-18#	223-40#	225-13#	225-24#	225-27#	225-30#	227-44#
	228-27#	228-36#	229-5#	232-12#	233-21#	233-24#	233-26#	233-31#	234-7#	235-23#	235-30#	236-1#	236-11#	236-14#
	237-6#	237-7#	238-5	238-5	238-5#	238-5#	238-11	238-11	238-11#	238-11#	239-10#	239-11#	241-3	241-3#
	241-10	241-10	241-10#	241-1、#	243-22#	243-24#	244-43#	245-8	245-8	245-8#	245-8#	246-8	246-8	246-8#
	246-8#	247-1#	247-5#	247-8#	247-11#	247-12#	248-21#	248-29#	248-30#	249-1#	249-2#	249-12#	249-14#	249-23#
	250-10#	253-5#	255-36#	256-4#	256-6	256-6#	256-6#	256-9#	256-17#	256-20#	256-22#	256-31#	257-3#	257-4#
	257-8#	257-9#	257-14#	257-15#	257-20#	257-21#	257-23#	258-10	258-10	258-10#	258-10#	258-12#	259-8	259-8
	259-8#	259-8#	259-21#	260-3	260-3#	260-6	260-6	260-6	260-6#	260-6#	260-6#	260-18#	261-1	261-1
	261-1	261-1#	261-1#	261-1#	261-3#	261-7#	261-10#	261-11#	261-14#	262-1	262-1	262-1	262-1#	262-1#
	262-1#	262-15#	263-1	263-1	263-1	263-1#	263-1#	263-1#	263-26#	263-32#	264-1	264-1	264-1	264-1#
	264-1#	264-1#	264-19#	264-20#	264-35#	265-10#	265-21#	265-27#	266-1#	266-5#	266-10#	266-11#	267-1	267-1
	267-1	267-1#	267-1#	267-1#	267-4#	268-1	268-1	268-1	268-1#	268-1#	268-1#	268-6#	269-1	269-1
	269-1	269-1#	269-1#	269-1#	269-10#	269-14#	270-11#	270-19#	271-3	271-3	271-3	271-3#	271-3#	271-3#
	271-24#	272-3	272-3	272-3	272-3#	272-3#	272-3#	272-14#	273-3	273-3	273-3	273-3#	273-3#	273-3#
	274-7#	275-3	275-3#	275-3#	275-7	275-7#	275-7#	276-40#	276-43#	276-44#	277-14	277-14#	277-14#	278-2
	278-2#	278-2#	278-8	278-8#	278-8#	278-14	278-14#	278-14#	278-20	278-20#	278-20#	278-21	278-21#	278-21#
	278-24	278-24#	278-24#	279-4	279-4#	279-4#	279-13	279-13#	279-13#	279-14	279-14#	279-14#	279-17	279-17#
	279-17#	280-2	280-2#	280-2#	280-4	280-4#	280-4#	281-2	281-2#	281-2#	283-4	283-4#	283-4#	283-10
	283-10#	283-10#	283-14	283-14#	283-14#	284-10	284-10#	284-10#	284-14	284-14#	284-14#	284-20	284-20#	284-20#
	285-2	285-2#	285-2#	285-8	285-8#	285-8#	285-22	285-22#	285-22#	286-2	286-2#	286-2#	286-7#	287-3
	287-3#	287-14	287-14	287-14#	287-14#	289-12	289-12	289-12#	289-12#	292-1	292-1#	292-3	292-3	292-3
	292-3#													
M\$LDRO	225-27	225-27#	233-21	233-21#	247-1	247-1#	247-5	247-5#	247-8	247-8#	247-12	247-12#	248-21	248-21#
	249-12	249-12#	249-14	249-14#	250-10	250-10#	253-5	253-5#	256-17	256-17#	256-20	256-20#	256-22	256-22#
	261-7	261-7#	264-20	264-20#	265-10	265-10#	265-21	265-21#	266-1	266-1#	266-10	266-10#		
M\$MCHI	112-12	112-12#												
M\$MCLO	112-12	112-12#												
M\$POPOP	114-18	114-18#	115-19	115-19#	115-21	115-21#	140-16	140-16#	140-20	140-20#	140-24	140-24#	140-28	140-28#
	140-32	140-32#	140-36	140-36#	140-40	140-40#	140-44	140-44#	140-48	140-48#	140-57	140-57#	140-63	140-63#
	140-77	140-77#	140-81	140-81#	140-85	140-85#	140-89	140-89#	140-93	140-93#	140-97	140-97#	140-101	140-101#
	140-105	140-105#	140-109	140-109#	140-114	140-114#	140-118	140-118#	140-122	140-122#	140-127	140-127#	140-131	140-131#
	140-135	140-135#	140-139	140-139#	141-22	141-22#	219-14	219-14#	220-21	220-21#	238-9	238-9#	238-13	238-13#
	240-38	240-38#	244-43	244-43#	245-14	245-14#	257-23	257-23#	258-12	258-12#	259-21	259-21#	259-23	259-23#
	261-14	261-14#	262-15	262-15#	263-32	263-32#	266-11	266-11#	267-4	267-4#	268-6	268-6#	269-10	269-10#
	269-14	269-14#	271-24	271-24#	272-14	272-14#	286-7	286-7#	286-9	286-9#	288-9	288-9#	290-15	290-15#
	291-10	291-10#												
M\$PRIN	140-62	140-62#	205-12	205-12#	205-14	205-14#	205-16	205-16#	205-18	205-18#	243-22	243-22#	243-24	243-24#
	255-36	255-36#												
M\$PUSH	112-33	112-33#	114-10	114-10#	115-10	115-10#	116-3	116-3#	140-14	140-14#	140-18	140-18#	140-22	140-22#

	140-26	140-26#	140-30	140-30#	140-34	140-34#	140-38	140-38#	140-42	140-42#	140-46	140-46#	140-50	140-50#
	140-59	140-59#	140-65	140-65#	140-79	140-79#	140-83	140-83#	140-87	140-87#	140-91	140-91#	140-95	140-95#
	140-99	140-99#	140-103	140-103#	140-107	140-107#	140-111	140-111#	140-116	140-116#	140-120	140-120#	140-124	140-124#
	140-129	140-129#	140-133	140-133#	140-137	140-137#	141-1	141-1#	219-10	219-10#	220-18	220-18#	238-5	238-5#
	238-11	238-11#	241-3	241-3#	241-10	241-10#	245-8	245-8#	246-8	246-8#	258-10	258-10#	259-8	259-8#
	260-3	260-3#	260-6	260-6#	261-1	261-1#	262-1	262-1#	263-1	263-1#	264-1	264-1#	267-1	267-1#
	268-1	268-1#	269-1	269-1#	271-3	271-3#	272-3	272-3#	273-3	273-3#	287-3	287-3#	287-14	287-14#
	289-12	289-12#												
MSPUT	140-62	140-62	140-62#	205-12	205-12	205-12	205-12#	205-14	205-14	205-14	205-14#	205-16	205-16	205-16
	205-16#	205-18	205-18	205-18	205-18#	208-32	208-32	208-32	208-32	208-32	208-32#	225-24	225-24	225-24
	225-24#	243-22	243-22	243-22	243-22	243-22	243-22	243-22	243-22	243-22	243-22#	243-24	243-24	243-24
	243-24	243-24#	249-23	249-23	249-23	249-23	249-23#	255-36	255-36	255-36	255-36	255-36	255-36#	255-36#
	261-3	261-3	261-3	261-3#	264-19	264-19	264-19	264-19	264-19#	264-19#				261-3
MSPUT1	140-62	140-62	140-62#	140-62#	205-12	205-12	205-12	205-12#	205-12#	205-12#	205-14	205-14	205-14	205-14#
	205-14#	205-14#	205-16	205-16	205-16	205-16#	205-16#	205-16#	205-18	205-18	205-18	205-18#	205-18#	205-18#
	208-32	208-32	208-32	208-32	208-32#	208-32#	208-32#	208-32#	225-24	225-24	225-24	225-24	225-24#	225-24#
	225-24#	225-24#	243-22	243-22	243-22	243-22	243-22	243-22	243-22	243-22	243-22#	243-22#	243-22#	243-22#
	243-22#	243-22#	243-22#	243-22#	243-24	243-24	243-24	243-24	243-24	243-24	243-24#	243-24#	243-24#	243-24#
	249-23	249-23	249-23	249-23	249-23#	249-23#	249-23#	249-23#	255-36	255-36	255-36	255-36	255-36	255-36#
	255-36#	255-36#	255-36#	255-36#	261-3	261-3	261-3	261-3	261-3#	261-3#	261-3#	261-3#	261-3#	264-19
	264-19	264-19	264-19#	264-19#	264-19#	264-19#	264-19#	264-19#						
MSRADI	163-3	163-3#	256-6	256-6#	275-3	275-3#	275-7	275-7#	277-14	277-14#	278-2	278-2#	278-8	278-8#
	278-14	278-14#	278-20	278-20#	278-21	278-21#	278-24	278-24#	279-4	279-4#	279-13	279-13#	279-14	279-14#
	279-17	279-17#	280-2	280-2#	280-4	280-4#	281-2	281-2#	283-4	283-4#	283-10	283-10#	283-14	283-14#
	284-10	284-10#	284-14	284-14#	284-20	284-20#	285-2	285-2#	285-8	285-8#	285-22	285-22#	286-2	286-2#
	288-1	288-1#	288-2	288-2#	288-3	288-3#	288-4	288-4#	288-5	288-5#	288-7	288-7#	290-1	290-1#
	290-6	290-6#	290-7	290-7#	290-8	290-8#	290-9	290-9#	290-10	290-10#	290-10#			
MSRBRO	233-26	233-26#	233-31	233-31#	234-7	234-7#	235-23	235-23#	235-30	235-30#	236-1	236-1#	236-11	236-11#
	236-14	236-14#												
MSRNRO	173-49	173-49#	175-38	175-38#	177-44	177-44#	248-21	248-21#	249-2	249-2#	249-12	249-12#	249-14	249-14#
	250-10	250-10#	253-5	253-5#										
MSSETS	112-33	112-33#	114-10	114-10#	115-10	115-10#	116-3	116-3#	140-14	140-14#	140-18	140-18#	140-22	140-22#
	140-26	140-26#	140-30	140-30#	140-34	140-34#	140-38	140-38#	140-42	140-42#	140-46	140-46#	140-50	140-50#
	140-59	140-59#	140-65	140-65#	140-79	140-79#	140-83	140-83#	140-87	140-87#	140-91	140-91#	140-95	140-95#
	140-99	140-99#	140-103	140-103#	140-107	140-107#	140-111	140-111#	140-116	140-116#	140-120	140-120#	140-124	140-124#
	140-129	140-129#	140-133	140-133#	140-137	140-137#	141-1	141-1#	219-10	219-10#	220-18	220-18#	238-5	238-5#
	238-11	238-11#	241-3	241-3#	241-10	241-10#	245-8	245-8#	246-8	246-8#	258-10	258-10#	259-8	259-8#
	260-3	260-3#	260-6	260-6#	261-1	261-1#	262-1	262-1#	263-1	263-1#	264-1	264-1#	267-1	267-1#
	268-1	268-1#	269-1	269-1#	271-3	271-3#	272-3	272-3#	273-3	273-3#	287-3	287-3#	287-14	287-14#
	289-12	289-12#												
MSSVC	140-16	140-16#	140-20	140-20#	140-24	140-24#	140-28	140-28#	140-32	140-32#	140-36	140-36#	140-40	140-40#
	140-44	140-44#	140-48	140-48#	140-57	140-57#	140-62	140-62#	140-63	140-63#	140-77	140-77#	140-81	140-81#
	140-85	140-85#	140-89	140-89#	140-93	140-93#	140-97	140-97#	140-101	140-101#	140-105	140-105#	140-109	140-109#
	140-114	140-114#	140-118	140-118#	140-122	140-122#	140-127	140-127#	140-131	140-131#	140-135	140-135#	140-139	140-139#
	141-22	141-22#	143-7	143-8	143-8#	147-21	147-22	147-22#	149-22	149-40	149-43	149-43#	150-10	150-10#
	151-21	152-11	160-32	160-32#	163-3	163-3#	173-49	173-49#	175-38	175-38#	177-44	177-44#	195-24	205-12
	205-12#	205-14	205-14#	205-16	205-16#	205-18	205-18#	207-41	207-41#	208-32	208-32#	212-3	216-26	216-26#
	216-34	217-5	223-18	223-40	225-13	225-13#	225-24	225-24#	225-27	225-27#	225-30	227-44	228-27	228-27#
	228-36	229-5	232-12	232-12#	233-21	233-21#	233-24	233-24#	233-26	233-26#	233-31	233-31#	234-7	234-7#
	235-23	235-23#	235-30	235-30#	236-1	236-1#	236-11	236-11#	236-14	236-14#	237-6	237-7	237-7#	239-10
	239-10#	239-11	239-11#	243-22	243-22#	243-24	243-24#	244-12#	244-43	244-43#	247-1	247-1#	247-5	247-5#
	247-8	247-8#	247-11	247-11#	247-12	247-12#	248-21	248-21#	248-29	248-30	248-30#	249-1	249-1#	249-2
	249-2#	249-12	249-12#	249-14	249-14#	249-23	249-23#	250-10	250-10#	253-5	253-5#	255-36	255-36#	256-4
	256-4#	256-6	256-6#	256-9	256-9#	256-17	256-17#	256-20	256-20#	256-22	256-22#	256-31	256-31#	257-3
	257-4	257-4#	257-8	257-9	257-9#	257-14	257-14#	257-15	257-15#	257-20	257-21	257-21#	257-23	257-23#
	258-12#	259-21	259-21#	260-18	260-18#	261-1	261-1#	261-3	261-3#	261-7	261-7#	261-10	261-11	261-11#

	261-14	261-14#	262-1	262-1#	262-15	262-15#	263-1	263-1#	263-26	263-32	263-32#	264-1	264-1#	264-19
	264-19#	264-20	264-20#	264-35	264-35#	265-10	265-10#	265-21	265-21#	265-27	266-1	266-1#	266-5	266-10
	266-10#	266-11	266-11#	267-1	267-1#	267-4	267-4#	268-1	268-1#	268-6	268-6#	269-1	269-1#	269-10
	269-10#	269-14	269-14#	270-11	270-11#	270-19	271-24	271-24#	272-14	272-14#	274-7	274-7#	275-3	275-3#
	275-7	275-7#	276-40	276-40#	276-43	276-43#	276-44	276-44#	277-14	277-14#	278-2	278-2#	278-8	278-8#
	278-14	278-14#	278-20	278-20#	278-21	278-21#	278-24	278-24#	279-4	279-4#	279-13	279-13#	279-14	279-14#
	279-17	279-17#	280-2	280-2#	280-4	280-4#	281-2	281-2#	283-4	283-4#	283-10	283-10#	283-14	283-14#
	284-10	284-10#	284-14	284-14#	284-20	284-20#	285-2	285-2#	285-8	285-8#	285-22	285-22#	286-2	286-2#
	286-7	286-7#												
MSTLAB	140-16#	140-20#	140-24#	140-28#	140-32#	140-36#	140-40#	140-44#	140-48#	140-57#	140-62#	140-63#	140-77#	140-81#
	140-85#	140-89#	140-93#	140-97#	140-101#	140-105#	140-109#	140-114#	140-118#	140-122#	140-127#	140-131#	140-135#	140-139#
	141-22#	143-7#	143-8#	147-21#	147-22#	149-22#	149-40#	149-3#	150-10#	151-21#	152-11#	160-32#	163-3#	173-49#
	175-38#	177-44#	195-24#	205-12#	205-14#	205-16#	205-18#	207-41#	208-32#	212-3#	216-26#	216-34#	217-5#	223-18#
	223-40#	225-13#	225-24#	225-27#	225-30#	227-44#	228-27#	228-36#	229-5#	232-12#	233-21#	233-24#	233-26#	233-31#
	234-7#	235-23#	235-30#	236-1#	236-11#	236-14#	237-6#	237-7#	239-10#	239-11#	243-22#	243-22#	243-24#	247-1#
	247-5#	247-8#	247-11#	247-12#	248-21#	248-29#	248-30#	249-1#	249-2#	249-12#	249-14#	249-23#	250-10#	253-5#
	255-36#	256-4#	256-6#	256-9#	256-17#	256-20#	256-22#	256-31#	257-3#	257-4#	257-8#	257-9#	257-14#	257-15#
	257-20#	257-21#	257-23#	258-12#	259-21#	260-18#	261-1#	261-3#	261-7#	261-10#	261-11#	261-14#	262-1#	262-15#
	263-1#	263-26#	263-32#	264-1#	264-19#	264-20#	264-35#	265-10#	265-21#	265-27#	266-1#	266-5#	266-10#	266-11#
	267-1#	267-4#	268-1#	268-6#	269-1#	269-10#	269-14#	270-11#	270-19#	271-24#	272-14#	274-7#	275-3#	275-7#
	276-40#	276-43#	276-44#	277-14#	278-2#	278-8#	278-14#	278-20#	278-21#	278-24#	279-4#	279-13#	279-14#	279-17#
	280-2#	280-4#	281-2#	283-4#	283-10#	283-14#	284-10#	284-14#	284-20#	285-2#	285-8#	285-22#	286-2#	286-7#
MSTSTL	140-16	140-16#	140-20	140-20#	140-24	140-24#	140-28	140-28#	140-32	140-32#	140-36	140-36#	140-40	140-40#
	140-44	140-44#	140-48	140-48#	140-57	140-57#	140-62	140-62#	140-63	140-63#	140-77	140-77#	140-81	140-81#
	140-85	140-85#	140-89	140-89#	140-93	140-93#	140-97	140-97#	140-101	140-101#	140-105	140-105#	140-109	140-109#
	140-114	140-114#	140-118	140-118#	140-122	140-122#	140-127	140-127#	140-131	140-131#	140-135	140-135#	140-139	140-139#
	141-22	141-22#	143-7	143-7#	143-7#	143-8	143-8#	147-21	147-21#	147-21#	147-22	147-22#	149-22	149-22#
	149-22#	149-40	149-40#	149-40#	149-43	149-43#	150-10	150-10#	151-21	151-21#	151-21#	152-11	152-11#	152-11#
	160-32	160-32#	163-3	163-3#	173-49	173-49#	175-38	175-38#	177-44	177-44#	195-24	195-24#	195-24#	205-12
	205-12#	205-14	205-14#	205-16	205-16#	205-18	205-18#	207-41	207-41#	208-32	208-32#	212-3	212-3#	212-3#
	216-26	216-26#	216-34	216-34#	216-34#	217-5	217-5#	217-5#	223-18	223-18#	223-18#	223-40	223-40#	223-40#
	225-13	225-13#	225-24	225-24#	225-27	225-27#	225-30	225-30#	225-30#	227-44	227-44#	227-44#	228-27	228-27#
	228-36	228-36#	228-36#	229-5	229-5#	229-5#	232-12	232-12#	233-21	233-21#	233-24	233-24#	233-26	233-26#
	233-31	233-31#	234-7	234-7#	235-23	235-23#	235-30	235-30#	236-1	236-1#	236-11	236-11#	236-14	236-14#
	237-6	237-6#	237-6#	237-7	237-7#	239-10	239-10#	239-11	239-11#	243-22	243-22#	243-24	243-24#	244-43
	244-43#	247-1	247-1#	247-5	247-5#	247-8	247-8#	247-11	247-11#	247-12	247-12#	248-21	248-21#	248-29
	248-29#	248-29#	248-30	248-30#	249-1	249-1#	249-2	249-2#	249-12	249-12#	249-14	249-14#	249-23	249-23#
	250-10	250-10#	253-5	253-5#	255-36	255-36#	256-4	256-4#	256-6	256-6#	256-9	256-9#	256-17	256-17#
	256-20	256-20#	256-22	256-22#	256-31	256-31#	257-3	257-3#	257-3#	257-4	257-4#	257-8	257-8#	257-8#
	257-9	257-9#	257-14	257-14#	257-14#	257-15	257-15#	257-20	257-20#	257-20#	257-21	257-21#	257-23	257-23#
	258-12	258-12#	259-21	259-21#	260-18	260-18#	261-1	261-1#	261-3	261-3#	261-7	261-7#	261-10	261-10#
	261-10#	261-11	261-11#	261-14	261-14#	262-1	262-1#	262-15	262-15#	263-1	263-1#	263-26	263-26#	263-26#
	263-32	263-32#	264-1	264-1#	264-19	264-19#	264-20	264-20#	264-35	264-35#	265-10	265-10#	265-21	265-21#
	265-27	265-27#	265-27#	266-1	266-1#	266-5	266-5#	266-5#	266-10	266-10#	266-11	266-11#	267-1	267-1#
	267-4	267-4#	268-1	268-1#	268-6	268-6#	269-1	269-1#	269-10	269-10#	269-14	269-14#	270-11	270-11#
	270-19	270-19#	270-19#	271-24	271-24#	272-14	272-14#	274-7	274-7#	275-3	275-3#	275-7	275-7#	276-40
	276-40#	276-43	276-43#	276-44	276-44#	277-14	277-14#	278-2	278-2#	278-8	278-8#	278-14	278-14#	278-20
	278-20#	278-21	278-21#	278-24	278-24#	279-4	279-4#	279-13	279-13#	279-14	279-14#	279-17	279-17#	280-2
	280-2#	280-4	280-4#	281-2	281-2#	283-4	283-4#	283-10	283-10#	283-14	283-14#	284-10	284-10#	284-14
	284-14#	284-20	284-20#	285-2	285-2#	285-8	285-8#	285-22	285-22#	286-2	286-2#	286-7	286-7#	
MSWORD	112-42	112-42#	113-9	113-9	113-9	113-9	113-9	113-9	143-7	143-7	143-7	143-7#	147-21	147-21
	147-21	147-21#	149-22	149-22	149-22	149-22#	149-40	149-40	149-40	149-40#	151-21	151-21	151-21	151-21#
	152-11	152-11	152-11	152-11#	163-3	163-3	163-3#	163-3#	195-24	195-24	195-24	195-24#	212-3	212-3
	212-3	212-3#	216-34	216-34	216-34	216-34#	217-5	217-5	217-5	217-5#	223-18	223-18	223-18	223-18#
	223-40	223-40	223-40	223-40#	225-30	225-30	225-30	225-30#	227-44	227-44	227-44	227-44#	228-36	228-36
	228-36	228-36#	229-5	229-5	229-5	229-5#	237-6	237-6	237-6	237-6#	244-12	244-12#	248-29	248-29

