RA60 RA80 UDA AND DISK DRV DIAG
RA81 UDA-50A CZUDCEO

COPYRIGHT (c) 1981-84
AH-S831E-MC
FICHE 01 OF 02

JUL 1984
digital
Made In USA

.REM @

IDENTIFICATION

PRODUCT CODE: AC S830E MC

PRODUCT NAME: CZUDCEO UDA & DISK DRV DIAG

PRODUCT DATE: 04-OCT 83

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: MATT TEDONE

@
.REM @

TABLE OF CONTENTS

D1

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:

CZUDEDO   UDA 50 Disk Drive Formatter.

CXUDFBO - UDA-50 Disk Drive Formatter Data File

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

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.

This diagnostic will test UDA 50's with modules M7485 and M7486.
Whenever a fault is detected in a UDA-50 and the fault can be
isolated to one of the two modules in the UDA 50. Replace that module.

E1

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.
Th's program requires the following:

        PDP 11 Unibus processor
        28K words of memory (minimum)
        Console terminal
        XXDP+ load media containing this program and the ZUDDEO.PAK
                data file
        One or more UDA50 subsystems. The subsystem controller must be
                type UDA50-A with microcode level 3 or greater.
        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 ZUDDEO.PAK be on
the XXDP+ system device. This data file is ordered under the name
C7UDDEO. The XXDP+ system device must remain on line during the
execution of this diagnostic.

F1

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
'DDDDD'.

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:DDDDD	Execute DDDDD passes (DDDDD = 1 to 64000)
/FLAGS:FLGS	Set specified flags. Flags are described in section 2.3.
/EOP:DDDDD	Report end of pass message after every DDDDD passes only. (DDDDD = 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 cf 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 un't drive, each sub unit
> number on the drive must be tested as a separate unit to
> completely test the drive. A maximum of eight log'cal drives
> may be tested on one UDA at a time (UDA configuration lim't).

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 65535 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.

K1

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 CONT 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 .ABLE 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 (0) ? 160000<CR>
         SUB-DEVICE # (0) ? 0<CR>
         Q-FACTOR (0) 0 ? 1<CR>

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

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

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

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

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

         UNIT 7
         CSR ADDRESS (0) ? 160000<CR>
         SUB DEVICE # (0) ? 6<CR>
         Q FACTOR (0) 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 (D) ?  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 accompl'shed in one pass as shown
below.

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

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

As you can see from this example, null repl'es (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 ZUDCE0"

   4. Type "START'

   5. Answer the "CHANGE HW" quest'on with 'Y'

   6. Answer all the hardware questions

   7. Answer the "CHANGE SW" question with "N'

When you follow this procedure you will be us ng only the defaults for
flags and software parameters. These defaults are descr'bed 'n
sections 2.3 and 2.5.

B2

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

CZUDC 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:XXX.(X
         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 un t 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 ZUDCEO" command to the XXDP+ monitor is only a small part of this
diagnostic. A data file called ZUDDEO.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 pogram. 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 xxxxx
     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 ident'fies 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 'n 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      general message
HOST PROGRAM  UDA AT 172150  RUNTIME 0:00:12                        basic message
UDA RESIDENT DIAGNOSTICS DETECTED FAILURE              \
   UDASA CONTAINS 104041                               )  extended message
REPLACE UDA MODULE M7485                               /
```

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

Sample informational message:

UNIT O UDA AT 172150 DRIVE O 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
d'agnostic 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. Add'tional information about the error may follow tne
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 car 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 quest;ons.

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 Serv'ces
                 prompt so that you can change the hardware questions.

G2

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 tc 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 read'ng
                    the data file is declared. Restore the data file
                    ZUDDEO.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 w'll occur
                    if the tables were corrupted after restarting the d'ag
                    nostic. 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 med'a properly. Restore the data file
                    ZUDDEO.PAK to your load med'a.

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 ⊢   .dentical vector addresses. The
                    program is aborted and returns to the Runtime Services prompt
                    so that you can change the hardware questions.

00010   CZUDC DVC FTL ERR   00010 ON UNIT 00 TST xxx SUB 000 PC: xxxxxx
          HOST PROGRAM UDA AT 172150   RUNTIME x:xx:xx
          WRONG APT DIAGNOSTIC IS BEING USED WITH THIS CONTROLLER
          USE CIUDx

                    The APT diagnostics are designed to run with one type of
                    UDA 50 board set (either M7161-2 or M7485-6). For example,
                    If the user is running CIUDA with a UDA-50 M7485-6 type,
                    this error will occur. In that case the user will be told
                    to use CIUDF. The following is a detailed description of
                    which test is used with what configuration.

                              CIUDF - UDA-50 with M7485-6 modules runs tests 1-3
                              CIUDG - UDA-50 with M7485-6 modules runs test 4
                              CIUDH - UDA-50 with M7485 6 modules runs tests 1 3
                              CIUDI   UDA-50 with M7485 6 modules runs test 4

I2

00014   CZUDC DVC FTL ERR 00014 ON UNIT 00 TST xxx SUB xxx PC: xxxxxx
         HOST PROGRAM UDA AT 172150 RUNTIME x:xx:xx
         UDA50 CONTROLLER IS AT A REVISION LEVEL NO LONGER SUPPORTED
         BY THIS DIAGNOSTIC PROGRAM. THIS PROGRAM REQUIRES A UDA50-A
         CONTROLLER (MODEL 6) WITH MICROCODE REVISON AT 3 OR GREATER.

         CONTROLLER REPORTED MODEL CODE xx AND MICROCODE VERSION xx

                  All UDA50-0's (modules M7161-2) are not supported by this diag-
                  nostic. The module set M7485-6 is the only one that can be tested
                  by this diagnostic. If the controller is a UDA50-0 (M7161 2)
                  't will not be tested. If the controller is a UDA50-A (M7485-6)
                  and it has old microcode (the microcode version is less than
                  3) this message will be printed but testing will go on. If
                  the controller consists of the M7161-2 modules, install one
                  with M7485-6 modules. Do not intermix the two, it will not
                  work!

```
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 M7486
```

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

```
104000 - Fatal sequncer 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 buffer 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 - Unexpectant 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. M7485 is the Un'bus
interface board. M7486 is the SDI interface board.

K2

```
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 M7485
```

                    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 M7485
```

                    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 M7485 module or the UNIBUS
                   maybe 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 M7485

                   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 M7485.

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 M7485

                   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 M7485.

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 M7485

                   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 M7485.

M2

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 M7485

                 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 M7485.

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 M7485
             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 M7485.

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 inited 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 TST 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 00000C
               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.

C3

```
00037    CZUDC DVC FTL ERR  00037 ON UNIT 00 TST xxx SUB 000 PC: xxxx-x
         HOST PROGRAM  UCA AT 172150  RUNTIME x:xx:xx
         UDA REPORTED FATAL ERROR IN UDASA REGISTER WHILE LOADING DM PROGRAM
          UDASA CONTAINS  100004
         REPLACE UDA MODULE M7485
```

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 M7486
         OR UNIBUS
         OR REPLACE UDA MODULE M7485
```

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
M7485 is broken or the UNIBUS is broken.

3.2.2   TEST 1 ERROR MESSAGES (01000 TO 01999)

01000   CZUDC HRD ERR 010C0 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 existant 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.

F3

```
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-existant 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-existant 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 ERRO  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 existant memory location occurred.

F 3

```
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)      OAC2(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 wh'ch
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 h  ..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 d'sk
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 dr've
gave too much data for the DM program to handle.
This message may preceed the prev'ous 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 RESTARING

              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 thr
              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
              inited. 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 orive after the drive was in'ted.
              There may be a bad transmitter on the drive s de
              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 inited.
              There may be a bad transmitter on the drive side or
              a bad receiver on the UDA s'de or the SDI cable may
              have taken a n't.

I3

```
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:xy x  UDA AT xxxxxx  DRIVE xxx  RUNTIME hhh:mm:ss
         ERROR DURING RECEIVE OF ECHO RESPONSE FROM DRIVE
           E HO 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 responce from a GET STATUS command.
                   The error bits in the responce are in bit position 3, 5
                   and 6 of word2.  For further description of the
                   GET STATUS responce, 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 exp red)
                           0040 - Available (drive offline and usuable)
                           1000   Read/Write ready
```

J3

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 cabl .

K3

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 responce 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   0501   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 n
              hard error 2009.

M3

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
                  s 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. EPROR 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.

           CZU
                                                                                                                            USE

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.

B4

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 RECLAIBRATE 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 00C3
         STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

              The GET STAUTS 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 HRL ERR   02045 ON UNI 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   CO

              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.

F 4

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 usuable)
        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 CHARACTERIS ICS
         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) befor 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.

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

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

H-4

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 0002
STATUS (R TO L): 1312  1110  0908  0706  0504  0302  0100
```

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.

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

03011
```
CZUDC HRD ERR  03011 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
DISK FUNCTION DM PC:xx.xx  UDA AT xxxxxx  DRIVE xxx  RUNTIME hhh:mm:ss
NO VALID STATE FROM DRIVE
NO DRIVE CLOCKS
CHECK THAT DRIVE IS POWERED ON.
```

If test 3 attempts to get the drive state, and finds that
there are no drive clocks on the port, the above message
is occurrs. This error usually means that the SDI cable
is not connected, the drive is not powered on or the
drive's port button that connects it to this UDA is not

depressed.

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
        NO VALID STATE FROM DRIVE
        HARD PARITY OR PULSE ERROR FOR 1/2 A SECOND

                If test 3 attempts to get the drive state, and gets pulse
                or parity errors for a full 1/2 second, the above message is
                printed. This error usually indicates a poor connection
                or grounding of the SDI cables, a bad drive transmitter, a
                bad UDA receiver or a broken 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 CHARACTERISICS 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
        STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

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

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

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 CHARACERISTICS 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
        CYLINDER aaa.   GROUP bb.   TRACK cc.

```
          REAL TIME STATE 0003
          STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

              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.

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

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

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.
          REAL TIME STATE 0003
          STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

              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.

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

                        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
          CYLINDER aaa.    GROUP bb.   TRACK cc.
          REAL TIME STATE 0003
          STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

              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.

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

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

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.
        REAL TIME STATE 0003
        STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

        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.

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

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
        CYLINDER aaa.   GROUP bb.   TRACK cc.
        REAL TIME STATE 0003
        STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

        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 transmitt'ng
properly or the UDA may not be receiving properly or
the SDI cable took a hit.

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

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

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 BLOC<:
         READ OPERATION REPORTED FAILURE -- ERROR CODE aaa OCTAL.
         CYLINDER ccc. GROUP dd. TRACK ee.
         REAL TIME STATE 0003
         STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

          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. No block
          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 compare failure
                            (desired header not found)
                        52 = SERDES overrun error
                        150 = data sync timeout on read
                        153 = suspected positioner error
                        ^13 = 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
               ccc 's the cylinder value in decimal.
               dd is the group value in decimal.
               ee is the track value in decimal.

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

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.
          REAL TIME STATE 0003
          STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

            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.

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

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
          REAL TIME STATE 0003
          STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

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

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

03030   CZUDC HRD ERR   03030 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
         DISK FUNCTION DM PC:xxxx   UDA AT xxxxxx   DRIVE xxx   RUNTIMc hhh:mm:ss
         INVALID LEVEL 2 COMMAND OPCODE aaaa WAS SUCCESSFUL
         REAL TIME STATE 0003
         STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

         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.

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

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
         REAL TIME STATE 0003
         STATUS (R TO L): 1312   1110   0908   0706   0504   0302   0100

         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

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

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 TRANSMITTION ERROR
         WHEN reason
         REAL TIME STATE 0003
         STATUS (R TO L : 1312   1110   0908   0706   0504   0302   0100

         Invalid level 1 sequences were sent to the drive.
         Severna sequences are tried and the drive should find
         fault w th 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 WIH A BAD CHECKSUM WAS SENT
                 A CONTINUE FRAME WAS SENT WITH NO START FRAME

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

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
         REAL TIME STATE 0003
         STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

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

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

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 DBN AREA WHEN DRIVE WAS WRITE PROTECTED
         REAL TIME STATE 0003
         STATUS (R TO L): 1312 1110 0908 0706 0504 0302 0100

              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.

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

03036   CZUDC DVC FTL ERR 03036 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
         DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
         DRIVE IS NOT PROPERLY FORMATTED.
         UDA WILL SPIN DOWN THIS DRIVE IF USED IN NORMAL SYSTEM OPERATION
         THIS DRIVE NEEDS TO BE FORMATTED.

              Test 3 reads a copy of the FCT in the XBN area and determined
              that the FCT was corrupted. Any normal operating system
              (which uses the UDA as a controller) will spin down the drive,
              so the drive will need to be reformatted.

03037   CZUDC DVC FTL ERR 03037 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
         DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh:mm:ss
         DRIVE IS FORMATTED IN 576 BYTE MODE.
         TO RUN WITH A UDA, THIS DRIVE NEEDS TO BE FORMATTED IN 512 BYTE MODE.
         UDA WILL SPIN DOWN THIS DRIVE IF USED IN NORMAL SYSTEM OPERATION
         THIS DRIVE NEEDS TO BE FORMATTED.

              Test 3 reads a copy of the FCT from the XBN area and determined
              that the drive was formatted in 576 byte mode. Any normal
              operating system (which uses the UDA as a controller) will spin
              down the drive, so the drive will need to be reformatted.

03038   CZUDC DVC FTL ERR 03038 ON UNIT 00 TST 003 SUB 000 PC: xxxxxx
         DISK FUNCTION DM PC:xxxx UDA AT xxxxxx DRIVE xxx RUNTIME hhh.mm:ss
         NO COPY OF THE FCT COULD BE READ.
         UDA WILL SPIN DOWN THIS DRIVE IF USED IN NORMAL SYSTEM OPERATION
         THIS DRIVE NEEDS TO BE FORMATTED.

              Test 3 attempted to read every copy of the FCT without success.
              Any normal operating system (which uses the UDA as a controller)
              will spin down the drive, so the drive will need to be reformatted

C5

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 udeadr DRIVE plug   RUNTIME hh:mm:ss
INITIAL WRITE COMPLETE

Whenever Test 4 is STArted with initial write enabled, <<OR>> whenever
it is STArted or RESt   ed 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 udeadr 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 udeadr 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 OO TST 04 SUB 000 PC: xxxxxx
         DISK EXERCISER   DM PC:xxxx   UDA AT xxxxxx   DRIVE xxx   RUNTIME hh:mm:ss
         ATTN ASSERTED DURING SEEK
         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 inserting 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 OO 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
         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 OO 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 OO 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 OO TST 04 SUB 000 PC: xxxxxx

E5

```
           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 occurr only when Test 4 is running in the
customer data area. It occurs when 1) A sector is either marked revectored
or the header can't be found in two revolutions of the disk (both cases
should be revectored) 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 revectored, and RBN s are what
LBN s are revectord 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 SENT
          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 reciever 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
                    FCC 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
            EDC COMPUTED edc
            EDC READ      edc
```

       edc: The edc computed and read in octal.

       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 's 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
            's not initally written, it is likely that data comparison failures will
            occur in the fist 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 dropps offline while being tested (a normal occurance 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 operato 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 SENT
                    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 Test 4 attempts to write on a write protected drive, error 4019
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 's because
          bad blocks in the diagnostic area are not revectored, and RBN s are what
          LBN's are revectord 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 t e 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 SENT
          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 inval'd 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 reciever ready is deasserted (header sync timeout 's
          <<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

K5

DRIVE NOT FORMATTED IN 512 BYTE MODE -- UNABLE TO TEST
FCT BLOCK ZERO MODE WORD: mode

*** THIS PACK HAS AN INVALID FORMAT AND CANNOT BE USED ***

       mode: The mode word found on the drive's FCT block zero.

       Error 4021 occurs only when Test 4 finds that the mode word found in
       FCT block zero is not the 512 byte mode word (126736 octal). See DEC
       STANDARD 166 "FCT Structure". Inhibit'ng the dropping of units has
       no effect on this error.

04022   CZUDC   DEV FTL ERR 04022 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 READ FCT BLOCK ZERO

          *** THIS PACK HAS AN INVALID FORMAT AND CANNOT BE USED ***

       Error 4022 occurs when test 4 is unable to read any copy of FCT
       block zero. 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 's 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 inh'bited, Test 4
       will loop in error recovery, printing this error, unt'l 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 un'ts is inhibited, Test 4
       will loop in error recovery, printing this error, until the error
       state 's 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
               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
```

M5

```
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.

```
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 BEFORE/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 BEFORE/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 revectored or whose header could not be found (both should

be revectored). 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 indicitave 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 revectored (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 revectored 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 occurance).

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   0704   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 BEFORE/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   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 BEFORE/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 CO 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.

F 6

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**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 quest ons. Inhibiting the dropping of units has no effect
> on th s 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' cyl nder,
> the read only disgnostic area may start on that same cylinder, and Test 4
> tries to write to the end of the cylinder that the operator spec fied.
> 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 initializetion 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 cylinde- that the operator specified. Therefore,
> specify the previous cylinder if cylinders must be specified.
> Inhib'ting 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   11:0   0908   0706   0504   0302   0100

> The real t me drive state receive error is detected at the end of an
> I/O operation and nd cates that there was a pulse or parity error
> in the rece pt 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:

J6

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 LEVEL 2 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 LEVEL 2 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

L6

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 erorr occured 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 subunited drive] [further explanation]
        A drive was found at the end of the cable. It may be a subunited
        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 unspinable. 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 NOi
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/Recover, 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 stra6dgy 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.

E 7

Level 2 Command Error Response   "DE" Bit Set

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

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

Retry/Recovery   UDA   The steps l'sted below are performed.

1.   An SDI GET STATUS command is 'ssued.

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 pos tioned 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 dev'ce
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, 'ncluding those commands 'ssued 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
val d 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
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   and 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

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).

J7

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 revectored 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 revectored 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 n 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 reciever 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 drve 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 resol.ed 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 CFCHING TABLES
            .

The Replacement and Caching Tables record the locat ons of all revectored LBN
sectors and the status of each RBN on the unit. Each copy of the table 's
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' 's 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 Cach ng Table Format

Each entry in the Replacement and Cach'ng Table represents an RBN on the un't.
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 entr'es past th's 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 'n the Replacement and Caching
Tables is:


```
     !<---------------- - -  -16 b'ts              >!
     .--------- ----- ----.                        .
     !                      LBN (low)              !
     .------------------- -  .                     .
     ! CODE !              LBN (h'gh)              !
     .--------------- ---.                         .
     ! 4 bits!<----   ' -   12 b ts                >!
```

C8

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 0C, 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:

```
.--- .. .--- ....-- .....-.
!       volume          !
!    information        !  SECTOR 0
!      block            !
.- .------- .--- ------.
!   128 bad block       !
!    descriptors        !  SECTOR 1
!    512   mode         !
.--- .- .--- ------.
!   128 bad block       !
!    descriptors        !  SECTOR 2
!    512   mode         !
.- .  .- .-  .-   ----.
.             .             .
.             .             .
.             .             .
!             .             !
.- .  .- -- .-          .
!   128 bad block       !
!    descriptors        !  SECTOR m
!    576   mode         !
.- .       .-         .
!   128 bad block       !
!    descriptors        !  SECTOR m+1
!    576   mode         !
.- .  .-                .
.             .             .
.             .             .
!             .             !
.-                     .
!   128 bad block       !
!    descriptors        !  SECTOR p
!    576   mode         !
.                      .
!    subsystem          !
!     scratch           !  SECTOR p+1
!     storage           !
.           .          .
.             .             .
.             .             .
!             .             !
.                      .
!    subsystem          !
!     scratch           !  SECTOR Fct 1
!     storage           !
.                      .
```

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
olume 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 volum formatting (high)	WORD 13
number of used entries in 512 table (low)	WORD 14

F8

```
.-----------------.
! 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 Inhib't 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   RU' TIME 2:24:10

During test #4, the report will contain statistics on each dr've 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	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 oy the READ TRANSFER LIMIT software question.
MBYTES WRITTEN	The number of mega-bytes written by th's 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.

H8

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.

Four "diagnostic machine" programs are in the ZUDDEO.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 initializies 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..." isprinted 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-existant 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.

K8

When all address bits have been tested, then block transfers to and
from memory are tested with different data patterns. Th's data
's 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
d'sk drive. These diagnostic programs may be resident in the disk
drive or require downline loading from the ZUDDEO.PAK data file.
(There currently are no disk drives that require oownline loading and
no such files exist in the ZUDDEO.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 Intervent'on 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 diagnost'c
capabilities are available.

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

```
TEST #2 MANUAL INTERVENTION ON UNIT xx UDA AT xxxxx 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 (exe-
cuted 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. Go to the XBN area and read a copy of the FCT. Check to see
        if the drive has been properly formatted in 512 byte mode.

    13. 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 diag
nosis?" 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 con
verted 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 (0) 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

C9

ENABLE RETRIES (L) Y

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

RANDOM ACCESS MODE (L) Y ?

    Answer '' 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 '' and the following two ques
tions will be asked:

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

> These questions are asked if the question immediately above
> was answered "1" 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.
Becaus the "diagnostic machine" programs are too large to fit both
copies in memory at the same time (as done in Tests 1 through 3),
the program checks which type of UDA-50s are being tested. If all
are of the same type, that program is read. If both types are selected
for testing, the program for the UDA-50 with the M7485 and M7486
boards is read.
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.

E9

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)   8B8B

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

F9

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 01101101101101110)

               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 ?

19

CZUDCEO UDA & DISK DRV DIAG MACRO V05.00 Wednesday 04 Jan 84 16:12 Page 81   SFQ 0112
USER DOCUMENTATION

END BLOCK (A) 0 ? 200

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

DATA WORD (0) 0 ?
a

```
     1                                ;*LAST REVISION 04-OCT-83
     2
   358                                .TITLE   CZUDCEO UDA & DISK DRV DIAG
   367
   368                                .SBTTL   PROGRAM HEADER
   394
   396 000000                                 .ASECT
   397                                         .ENABL  AMA
   398        002000                           .      =      2000
   400
   402                                ;**
   403                                ;  THE PROGRAM HEADER IS THE INTERFACE BETWEEN
   404                                ;  THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
   405                                ;--
   406
   408
   425
   427 002000                        L$NAME::                    ;DIAGNOSTIC NAME
       002000     103                         .ASCII  /L/
       002001     132                         .ASCII  /Z/
       002002     125                         .ASCII  /U/
       002003     104                         .ASCII  /D/
       002004     103                         .ASCII  /C/
       002005     000                         .BYTE   0
       002006     000                         .BYTE   0
       002007     000                         .BYTE   0
       002010                        L$REV::                     ;REVISION LEVEL
       002010     105                         .ASCII   /E/
       002011                        L$DEPO::                    ;0
       002011     060                         .ASCII   /0/
       002012                        L$UNIT::                    ;NUMBER OF UNITS
       002012     000001                      .WORD   T$PTHV
       002014                        L$TIML::                    ;LONGEST TEST TIME
       002014     000000                      .WORD   0
       002016                        L$HPCP::                    ;POINTER TO H.W. QUES.
       002016     113266                      .WORD   L$HARD
       002020                        L$SPCP::                    ;POINTER TO S.W. QUES.
       002020     113526                      .WORD   L$SOFT
       002022                        L$HPTP::                    ;PTR. TO DEF. H.W. PTABLE
       002022     064356                      .WORD   L$HW
       002024                        L$SPTP::                    ;PTR. TO S.W. PTABLE
       002024     064374                      .WORD   L$SW
       002026                        L$LADP::                    ;DIAG. END ADDRESS
       002026     114312                      .WORD   L$LAST
       002030                        L$STA::                     ;RESERVED FOR APT STATS
       002030     000000                      .WORD   0
       002032                        L$CO::
       002032     000000                      .WORD   0
       002034                        L$DTYP::                    ;DIAGNOSTIC TYPE
       002034     000001                      .WORD   1
       002036                        L$APT::                     ;APT EXPANSION
       002036     000000                      .WORD   0
       002040                        L$DTP::                     ;PTR. TO DISPATCH TABLE
       002040     064344                      .WORD   L$DISPATCH
       002042                        L$PRIO::                    ;DIAGNOSTIC RUN PRIORITY
       002042     000340                      .WORD   PRIO7
       002044                        L$ENVI::                    ;FLAGS DESCRIBE HOW IT WAS SETUP
```

K9

```
002044  000000                        .WORD    0
002046                    L$EXP1::                       ;EXPANSION WORD
002046  000000                        .WORD    0
002050                    L$MREV::                       ;SVC REV AND EDIT #
002050  003                           .BYTE    C$REVISION
002051  003                           .BYTE    C$EDIT
002052                    L$EF::                         ;DIAG. EVENT FLAGS
002052  000000                        .WORD    0
002054  000000                        .WORD    0
002056                    L$SPC::
002056  000000                        .WORD    0
002060                    L$DEVP::                       ; POINTER TO DEVICE TYPE LIST
002060  064700                        .WORD    L$DVTYP
002062                    L$REPP::                       ;PTR. TO REPORT CODE
002062  106654                        .WORD    L$RPT
002064                    L$EXP4::
002064  000000                        .WORD    0
002066                    L$EXP5::
002066  000000                        .WORD    0
002070                    L$AUT::                        ;PTR. TO ADD UNIT CODE
002070  000000                        .WORD    0
002072                    L$DUT::                        ;PTR. TO DROP UNIT CODE
002072  000000                        .WCRD    0
002074                    L$LUN::                        ;LUN FOR EXERCISERS TO FILL
002074  000000                        .WORD    0
002076                    L$DESP::                       ;POINTER TO DIAG. DESCRIPTION
002076  064724                        .WORD    L$DESC
002100                    L$LOAD::                       ;GENERATE SPECIAL AUTOLOAD EMT
002100  104035                        EMT      E$LOAD
002102                    L$ETP::                        ;POINTER TO ERRTBL
002102  064402                        .WORD    L$ERRTBL
002104                    L$ICP::                        ;PTR. TO INIT CODE
002104  107636                        .WORD    L$INIT
002106                    L$CCP::                        ;PTR. TO CLEAN UP CODE
002106  111302                        .WORD    L$CLEAN
002110                    L$ACP::                        ;PTR. TO AUTO CODE
002110  111300                        .WORD    L$AUTO
002112                    L$PRT::                        ;PTR. TO PROTECT TABLE
002112  107630                        .WORD    L$PROT
002114                    L$TEST::                       ;TEST NUMBER
002114  000000                        .WORD    0
002116                    L$DLY::                        ;DELAY COUNT
002116  000000                        .WORD    0
002:20                    L$HIME::                       ;PTR. TO HIGH MEM
002·20  000000                        .WORD    0
```

436

L9

```
    1
    2                                  ;THIS LOCATION MUST BE AT THIS POSITION. SEPERATE CODE, STORED IN
    3                                  ;THE PAK FILE, WAS ASSEMBLED WITH THIS ADDRESS
    4
    5         062220                   STOSIZ  = 26000. - 256.              ;STORAGE SIZE
    6
    8 002122                           STORAG: .BLKB    STOSIZ
```

```
 1                                      .SBTTL   DISPATCH TABLE
 2
 3                                    ;++
 4                                    ; THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
 5                                    ; IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
 6                                    ;--
 7
 8 064342  000004                       .WORD    4
   064344                       L$DISPATCH::
   064344  111344                       .WORD    T1
   064346  112420                       .WORD    T2
   064350  112516                       .WORD    T3
   064352  112554                       .WORD    T4
11
```

```
 1                                              .SBTTL   DEFAULT HARDWARE P TABLE
 2
 3                                         ;**
 4                                         ;  THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
 c                                         ;  THE TEST-DEVICE PARAMETERS    THE STRUCTURE OF THIS TABLE
 6                                         ;  IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P TABLES,
 7                                         ;  AND IS USED AS A "TEMPLATE" FOR BUILDING THE P TABLES.
 8                                         ; -
 9
10 064354   000006                         .WORD    L10000 L$HW/2
   064356                         L$HW::
   064356                         DFPTBL::
11 064356   172150                          .WORD    172150              ; UNIBUS ADDRESS
12 064360   000154                          .WORD    154                 ; VECTOR ADDRESS
13 064362   000005                          .WORD    5.                  ; BR LEVEL
14 064364   000077                          .WORD    63.                 ; UNIBUS BURST RATE
15 064366   000000                          .WORD    0.                  ; LOGICAL DRIVE NUMBER
16 064370   000000                          .WORD    0.                  ; CUSTOMER DATA AREA
17
27 064372                         L10000:
```

```
   1                                   .SBTTL   SOFTWARE P TABLE
   2
   3                               ;..
   4                               ; THE SOFTWARE TABLE CONTAINS VARIOUS DATA USED BY THE
   5                               ; PROGRAM AS OPERATIONAL PARAMETERS.  THESE PARAMETERS ARE
   6                               ; SET UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR
   :                               ; AT RUN TIME.
   8                               ;
   9
  10 064372  000003                         .WORD    L10001 L#SW/2
     064374                     L#SW::
     064374                     SFPTBL::
  11 064374  000040                         .WORD    32.                   ;ERROR LIMIT
  12 064376  000000                         .WORD    0.                    ;DATA TRANSFER LIMIT (MEGABITS)
  13 064400  040400                         .WORD    ^B0100000100000000    ;SINGLE BIT QUESTIONS
  21
  29
  30 064402                     L10001:
```

```
12                              .SBTTL   GLOBAL EQUATES SECTION
40
50
52                              ;..
53                              ; THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
54                              ; ARE USED IN MORE THAN ONE TEST.
55                              ;
56
57                              ;
                                ; 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
        000075                  EF.NEW==        29.             ; A NEW PASS HAS BEEN STARTED
        000034                  EF.PWR==        28.             ; A POWER FAIL/POWER UP OCCURRED
                                ;
                                ;
                                ; PRIORITY LEVEL DEFINITIONS
                                ;
        000340                  PRIO7==  340
        000300                  PRIO6==  300
        000240                  PRIO5==  240
        000200                  PRIO4==  200
        000140                  PRIO3==  140
        000100                  PRIO2==  100
```

D10

```
000040                    PRIO1·· 40
000000                    PRIO0·· 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
```

```
     1                                   .SBTTL   UDA BIT DEFINITIONS
     2
     3
     4                                ;UDASA REGISTER UNIVERSAL READ BITS
     5
     6        100000                  SA.ERR  = 100000              ;ERROR INDICATOR
     7        040000                  SA.S4   = 040000              ;STEP 4 STATUS BIT
     8        020000                  SA.S3   = 020000              ;STEP 3 STATUS BIT
     9        010000                  SA.S2   = 010000              ;STEP 2 STATUS BIT
    10        004000                  SA.S1   = 004000              ;STEP 1 STATUS BIT
    11
    12
    13                                ;UDASA REGISTER ERROR STATUS BITS
    14
    15        003777                  SA.ERC  = 003777              ;ERROR CODE
    16
    17
    18                                ;UDASA REGISTER STEP 1 SEND BITS
    19
    20        000177                  SA.VEC  = 000177              ;INTERRUPT VECTOR (DIVIDED BY 4)
    21        000200                  SA.INT  = 000200              ;INTERRUPT ENABLE DURING INITIALIZATION
    22        003400                  SA.MSG  = 003400              ;MESSAGE RING LENGTH
    23        034000                  SA.CMD  = 034000              ;COMMAND RING LENGTH
    24        040000                  SA.WRP  = 040000              ;WRAP BIT
    25        100000                  SA.STP  = 100000              ;STEP - MUST ALWAYS BE WRITTEN A ONE
    26
    27        000400                  SA.MS1  = 000400              ;LSB OF MESSAGE RING LENGTH
    28        004000                  SA.CM1  = 004000              ;LSB OF COMMAND RING LENGTH
    29
    30
    31                                ;UDASA REGISTER STEP 1 RESPONSE BITS
    32
    33        002000                  SA.NV   = 002000              ;NON SETTABLE INTERRUPT VECTOR
    34        001000                  SA.A2   = 001000              ;22 BIT ADDRESS BUS
    35        000400                  SA.DI   = 000400              ;ENHANCED DIAGNOSTICS
    36                                ;          000377              ;ALL BITS RESERVED
    37
    38
    39                                ;UDASA REGISTER STEP 2 SEND BITS
    40
    41        000001                  SA.PRG  = 000001              ;ENABLE VAX UNIBUS ADAPTER PURGE INTERRUPT
    42                                ;          177776              ;LOW ORDER MESSAGE RING BYTE ADDRESS
    43
    44
    45                                ;UDASA REGISTER STEP 2 RESPONSE BITS
    46
    47        000007                  SA.MSE  = 000007              ;MESSAGE RING LENGTH ECHO
    48        000070                  SA.CME  = 000070              ;COMMAND RING LENGTH ECHO
    49                                ;          000100              ;RESERVED
    50        000200                  SA.STE  = 000200              ;STEP ECHO
    51        003400                  SA.CTP  = 003400              ;CONTROLLER TYPE
    52
    53
    54                                ;UDASA REGISTER STEP 3 SEND BITS
    55
    56                                ;          077777              ;HIGH ORDER MESSAGE RING BYTE ADDRESS
    57        100000                  SA.TST  = 100000              ;PURGE POLE TEST ENABLE
```

F10

```
58
59
60                              ;UDASA REGISTER STEP 3 RESPONSE BITS
61
62      000177                  SA.VCE  = 000177               ;INTERRUPT VECTOR ECHO
63      000200                  SA.INE  = 000200               ;INTERRUPT ENABLE ECHO
64      000400                  SA.NVE  = 000400               ;VECTOR NOT PROGRAMMABLE
65                              :          003000               ;RESERVED
66
67
68                              ;UDASA REGISTER STEP 4 SEND BITS
69
70      000001                  SA.GO   = 000001               ;GO BIT TO START UDA FIRMWARE
71      000002                  SA.LFC  = 000002               ;LAST FAILURE CODE REQUEST
72      000374                  SA.BST  = 000374               ;BURST LEVEL
73
74
75                              ;UDASA REGISTER STEP 4 RESPONSE BITS
76
77      000017                  SA.MCV  = 000017               ;UDA MICROCODE VERSION
78      000360                  SA.CNT  = 000360               ;CONTROLLER TYPE
79                              :          003400               ;RESERVED
```

```
     1                                      .SBTTL   HOST COMMUNICATION AREA DEFINIIONS
     2
     3                                      ;COMMAND/MESSAGE RING BIT DEFINITIONS
     4
     5        100000                        RG.OWN  = 100000                  ;SET WHEN UDA OWNS RING
     6        040000                        RG.FLG  = 040000                  ;FLAG BIT
     7
     8
     9                                      ;VIRTUAL CIRCUIT IDENTIFIERS
    10
    11        000000                        MSCP    = 0                       ;MSCP CIRCUIT
    12        000001                        LOG     = 1                       ;LOG CIRCUIT
    13        177777                        DIAG    = -1                      ;DIAGNOSTIC CIRCUIT
    14        001000                        DUP     = 1000                    ;DIAGNOSTIC AND UTILITIES PROTOCOL
    15
    16
    17                                      ;OFFSETS INTO HOST COMMUNICATIONS AREA WITH ONE DESCRIPTOR TO EACH RING
    18                                      ;AND TWO PACKET
    19
    20        000004                        HC.ISZ  = 4.                      ;SIZE OF INTERRUPT INDICATOR WORDS
    21        000004                        HC.RSZ  = 4.                      ;SIZE OF RING IN BYTES
    22        000004                        HC.ESZ  = 4.                      ;SIZE OF ENVELOPE WORDS BEFORE PACKET
    23        000060                        HC.PSZ  = 48.                     ;SIZE OF COMMAND AND MESSAGE PACKETS
    24        000106                        HC.BSZ  = 70.                     ;SIZE OF BUFFER
    25
    26        000000                        HC.INT  = 0.                      ;INTERRUPT INDICATOR WORDS START
    27
    28        000004                        HC.MSG  = HC.INT+HC.ISZ           ;MESSAGE RING START
    29        000006                        HC.MCT  = HC.MSG+2.               ;MESSAGE RING CONTROL WORD
    30
    31        000010                        HC.CMD  = HC.MSG+HC.RSZ           ;COMMAND RING START
    32        000012                        HC.CCT  = HC.CMD+2.               ;COMMAND RING CONTROL WORDS
    33
    34        000014                        HC.MEV  = HC.CMD+HC.RSZ           ;MESSAGE ENVELOPE START
    35        000020                        HC.MPK  = HC.MEV+HC.ESZ           ;MESSAGE PACKET START
    36
    37        000014                        HC.CEV  = HC.MEV                  ;COMMAND ENVELOPE START
    38        000020                        HC.CPK  = HC.MPK                  ;COMMAND PACKET START
    43
    44        000100                        HC.BF1  = HC.CPK+HC.PSZ           ;FIRST BUFFER
    45        000206                        HC.BF2  = HC.BF1+HC.BSZ           ;SECOND BUFFER
    46
    47        000314                        HC.SIZ  = HC.BF2+HC.BSZ           ;TOTAL SIZE OF HOST COMMUNICATION AREA
```

```
 1
 2                              .SBTTL   HOST COMMUNICATION AREA LAYOUT
 3
 4                    HC.INT  )  +-------------------------------------------------+
 5                            )  |          INTERRUPT INDICATORS                   )  4 BYTES
 6                            )  |                                                 )
 7                               +-------------------------------------------------+
 8                    HC.MSG  )  |          MESSAGE (RESPONSE) RING                )  4 BYTES
 9                    HC.MCT  )  |                                                 )
10                               +-------------------------------------------------+
11                    HC.CMD  )  |          COMMAND RING                           )  4 BYTES
12                    HC.CCT  )  |                                                 )
13      HC.MEV & HC.CEV)          +-----------------------------------------------+
14                            )  |          MESSAGE & COMMAND ENVELOPE             )  4 BYTES
15      HC.MPK & HC.CPK)          |                                                )
16                            )  |          MESSAGE & COMMAND PACKET              )  48 BYTES
17                            )  |                                                 )
18                            )  |                                                 )
19                            )  |                                                 )
20                               +-----------------------------------------------+
21                    HC.BF1  |  |      BUFFER # 1 (RESPONSE TO DM PROGRAM)     |  70 BYTES
22                            |  |                                               |
23                            |  |                                               |
24                            |  |                                               |
25                               +-----------------------------------------------+
26                    HC.BF2  |  |      BUFFER # 2 (REQUEST FROM DM PROGRAM)    |  70 BYTES
27                            |  |                                               |
28                            |  |                                               |
29                            |  |                                               |
30                            |  |                                               |
31                               +-----------------------------------------------+
32                                NOTE: BYTES ARE GIVEN IN DECIMAL
```

```
  1                                    .SBTTL    COMMAND PACKET OPCODES DEFINITIONS
  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 ATTENION MESSAGE
 25          000103                    OP.RLC  = 103                         ;RESET COMMAND LIMIT ATTENTION MESSAGE
 26
 27          000001                    OP.GSS  = 1                           ;DUP GET DUST 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 DUST DATA
 31          000005                    OP.RSD  = 5                           ;DUP RECEIVE DUST DATA
 32
 33                                    ;NOTE: END PACKET OPCODES (ALSO CALLED ENDCODES) ARE FORMED BY ADDING THE END
 34                                    ;PACKET FLAG TO THE COMMAND OPCODE. FOE 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                              ;COMMAND MODIFIERS
   3
   4                              ;           = 020000              ;CLEAR SERIOUS EXCEPTION
   5          040000             MD.CMP  = 040000                   ;COMPARE
   6          100000             MD.EXP  = 100000                   ;EXPRESS REQUEST
   7          010000             MD.ERR  = 010000                   ;FORCE ERROR
   8          004000             MD.SCH  = 004000                   ;SUPPRESS CACHING (HIGH SPEED)
   9          002000             MD.SCL  = 002000                   ;SUPPRESS CACHING (LOW SPEED)
  10          000100             MD.SEC  = 000100                   ;SUPPRESS ERROR CORRECTION
  11          000400             MD.SER  = 000400                   ;SUPPRESS ERROR RECOVERY
  12          000200             MD.SSH  = 000200                   ;SUPPRESS SHADOWING
  13          000100             MD.WBN  = 000100                   ;WRITE BACK (NON-VOLATILE)
  14          000400             MD.WBV  = 000400                   ;WRITE BACK (VOLATILE)
  15          000020             MD.SEQ  = 000020                   ;WRITE SHADOW SET ONE UNIT AT A TIME
  16          000001             MD.SPD  = 000001                   ;SPIN-DOWN
  17          000001             MD.FEU  = 000001                   ;FLUSH ENTIRE UNIT
  18          000002             MD.VOL  = 000002                   ;VOLATILE ONLY
  19          000001             MD.NXU  = 000001                   ;NEXT UNIT
  20          000001             MD.RIP  = 000001                   ;ALLOW SELF DESTRUCTION
  21          000002             MD.IMF  = 000002                   ;IGNORE MEDIA FORMAT ERROR
  22          000004             MD.SWP  = 000004                   ;SET WRITE PROTECT
  23          000010             MD.CWB  = 000010                   ;CLEAR WRITE BACK DATA LOST
  24          000001             MD.PRI  = 000001                   ;PRIMARY REPLACEMENT BLOCK
  25
  26
  27                              ;END PACKET FLAGS
  28
  29          000200             EF.BBR  = 000200                   ;BAD BLOCK REPORTED
  30          000100             EF.BBU  = 000100                   ;BAD BLOCK UNREPORTED
  31          000040             EF.LOG  = 000040                   ;ERROR LOG GENERATED
  32          000020             EF.SEX  = 000020                   ;SERIOUS EXCEPTION
  33
  34
  35                              ;CONTROLLER FLAGS
  36
  37          000200             CF.ATN  = 000200                   ;ENABLE ATTENTION MESSAGES
  38          000100             CF.MSC  = 000100                   ;ENABLE MISCELLANEOUS ERROR LOG MESSAGES
  39          000040             CF.OTH  = 000040                   ;ENABLE OTHER HOST'S ERROR LOG MESSAGES
  40          000020             CF.THS  = 000020                   ;ENABLE THIS HOST'S ERROR LOG MESSAGES
  41          000002             CF.SHD  = 000002                   ;SHADOWING
  42          000001             CF.576  = 000001                   ;576 BYTE SECTORS
  43
  44
  45                              ;UNIT FLAGS
  46
  47          000001             UF.CMR  = 000001                   ;COMPARE READS
  48          000002             UF.CMW  = 000002                   ;COMPARE WRITES
  49          100000             UF.RPL  = 100000                   ;HOST INITIATED BAD BLOCK REPLACEMENT
  50          040000             UF.INA  = 040000                   ;INACTIVE SHADOW SET UNIT
  51          004000             UF.SCH  = 004000                   ;SUPPRESS CACHING (HIGH SPEED)
  52          002000             UF.SCL  = 002000                   ;SUPPRESS CACHING (LOW SPEED)
  53          000100             UF.WBN  = 000100                   ;WRITE-BACK (NON VOLATILE)
  54          020000             UF.WPH  = 020000                   ;WRITE PROTECT (HARDWARE)
  55          001000             UF.WPS  = 001000                   ;WRITE PROTECT (SOFTWARE OR VOLUME)
  56          000004             UF.576  = 000004                   ;576 BYTE SECTORS
```

```
     1                                    .SBTTL   COMMAND PACKET OFFSETS
     2
     3
     4                                ;GENERIC COMMAND PACKET OFFSETS
     5
     6          000000                P.CRF    = 0.                        ;COMMAND REFERENCE NUMBER
     7          000004                P.UNIT   = 4.                        ;UNIT NUMBER
     8          000010                P.OPCD   = 8.                        ;OPCODE
     9          000012                P.MOD    = 10.                       ;MODIFIERS
    10          000014                P.BCNT   = 12.                       ;BYTE COUNT
    11          000020                P.BUFF   = 16.                       ;BUFFER DESCRIPTOR
    12          000020                P.UADR   = 16.                       ;UNIBUS ADDRESS OF BUFFER DESCRIPTOR
    13          000034                P.LBN    = 28.                       ;LOGICAL BLOCK NUMBER
    14
    15
    16                                ;ABORT AND GET COMMAND STATUS COMMAND PACKET OFFSETS
    17
    18          000014                P.OTRF   = 12.                       ;OUTSTANDING REFERENCE NUMBER
    19
    20
    21                                ;ONLINE AND SET UNIT CHARACTERISTICS COMMAND PACKET OFFSETS
    22
    23          000016                P.UNFL   = 14.                       ;UNIT FLAGS
    24          000020                P.HSTI   = 16.                       ;HOST IDENTIFIER / RESERVED
    25          000034                P.ELGF   = 28.                       ;ERROR LOG FLAGS
    26          000040                P.SHUN   = 32.                       ;SHADOW UNIT
    27          000042                P.CPSP   = 34.                       ;COPY SPEED
    28
    29
    30                                ;REPLACE COMMAND PACKET OFFSETS
    31
    32          000014                P.RBN    = 12.                       ;REPLACEMENT BLOCK NUMBER
    33
    34
    35                                ;SET CONTROLLER CHARACTERISTICS COMMAND PACKET OFFSETS
    36
    37          000014                P.VRSN   = 12.                       ;MSCP VERSION
    38          000016                P.CNTF   = 14.                       ;CONTROLLER FLAGS
    39          000020                P.HTMO   = 16.                       ;HOST TIMEOUT
    40          000022                P.USEF   = 18.                       ;USE FRACTION
    41          000024                P.TIME   = 20.                       ;QUAD-WORD TIME AND DATE
    42
    43
    44                                ;MAINTENANCE READ AND MAINTENANCE WRITE COMMAND PACKET OFFSETS
    45
    46          000034                P.RGID   = 28.                       ;REGION ID
    47          000040                P.RGOF   = 32.                       ;REGION OFFSET
    48
    49
    50                                ;EXECUTE SUPPLIED PROGRAM COMMAND PACKET OFFSETS
    51
    52          000024                P.DMDT   = 20.                       ;DMDT TERMINAL ADDRESS (MAINT WRITE ONLY)
    53          000034                P.OVRL   = 28.                       ;BUFFER DESCRIPTOR FOR OPERLAYS
```

```
  1                                       .SBTTL   END PACKET OFFSETS
  2
  3
  4                                    ;GENERIC END PACKET OFFSETS
  5
  6          000000               P.CRF    = 0.                      ;COMMAND REFERENCE NUMBER
  7          000004               P.UNIT   = 4.                      ;UNIT NUMBER
  8          000010               P.OPCD   = 8.                      ;OPCODE (ALSO CALLED ENDCODE)
  9          000011               P.FLGS   = 9.                      ;END PACKET FLAGS
 10          000012               P.STS    = 10.                     ;STATUS
 11          000014               P.BCNT   = 12.                     ;BYTE COUNT
 12          000034               P.FBBK   = 28.                     ;FIRST BAD BLOCK
 13
 14
 15                                    ;GET COMMAND STATUS END PACKET OFFSETS
 16
 17          000014               P.OTRF   = 12.                     ;OUTSTANDING REFERENCE NUMBER
 18          000020               P.CMST   = 16.                     ;COMMAND STATUS
 19
 20
 21                                    ;GET UNIT STATUS END PACKET OFFSETS
 22
 23          000014               P.MLUN   = 12.                     ;MULTI UNIT CODE
 24          000016               P.UNFL   = 14.                     ;UNIT FLAGS
 25          000020               P.HSTI   = 16.                     ;HOST IDENTIFIER
 26          000024               P.UNTI   = 20.                     ;UNIT IDENTIFIER
 27          000034               P.MEDI   = 28.                     ;MEDIA TYPE IDENTIFIER
 28          000040               P.SHUN   = 32.                     ;SHADOW UNIT
 29          000042               P.SHST   = 34.                     ;SHADOW STATUS
 30          000044               P.TRKS   = 36.                     ;TRACK SIZE
 31          000046               P.GRPS   = 38.                     ;GROUP SIZE
 32          000050               P.CYLS   = 40.                     ;CYLINDER SIZE
 33          000054               P.RCTS     44.                     ;RCT TABLE SIZE
 34          000056               P.RBNS   = 46.                     ;RBNS / TRACK
 35          000057               P.RCTC   = 47.                     ;RCT COPIES
 36
 37
 38                                    ;ONLINE AND SET UNIT CHARACTERISTICS END PACKET AND AVAILABLE
 39                                    ;ATTENTION MESSAGE OFFSETS
 40
 41          000014               P.MLUN   = 12.                     ;MULTI-UNIT CODE
 42          000016               P.UNFL   = 14.                     ;UNIT FLAGS
 43          000020               P.HSTI   = 16.                     ;HOST IDENTIFIER
 44          000024               P.UNTI   = 20.                     ;UNIT IDENTIFIER
 45          000034               P.MEDI   = 28.                     ;MEDIA TYPE IDENTIFIER
 46          000040               P.SHUN   = 32.                     ;SHADOW UNIT
 47          0C0042               P.SHST   = 34.                     ;SHADOW STATUS
 48          000044               P.UNSZ   = 36.                     ;UNIT SIZE
 49          000050               P.VSER   = 40.                     ;VOLUME SERIAL NUMBER
 50
 51
 52                                    ;SET CONTROLLER CHARACTERISTICS END PACKET OFFSETS
 53
 54          000014               P.VRSN   = 12.                     ;MSCP VERSION
 55          000016               P.CNTF   = 14.                     ;CONTROLLER FLAGS
 56          000020               P.CTMO   = 16.                     ;CONTROLLER TIMEOUT
 57          000022               P.CNCL   = 18.                     ;CONTROLLER COMMAND LIMIT
```

M10

```
58        000024           P.CNTI  = 20.                      ;CONTROLLER ID
59
60
61                         ;GET DUST STATUS END PACKET OFFSETS
62
63        000014           P.DEXT  = 12.                      ;EXTENSION FOR DOWNLINE LOADABLE PROGRAM
64        000017           P.DFLG  = 15.                      ;FLAGS
65        000020           P.DPRG  = 16.                      ;PROGRESS INDICATOR FOR REMOTE PROGRAM
66        000024           P.DTMO  = 20.                      ;TIMEOUT
```

N10

```
 1                                  .SBTTL   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        000400                    ST.AOL  = 400                             ;ALREADY ON LINE
19
20                                  ;DUP MESSAGE TYPES
21
22        010000                    DU.QUE  = 10000                          ;QUESTION
23        020000                    DU.DFL  = 20000                          ;DEFAULT QUESTION
24        030000                    DU.INF  = 30000                          ;INFORMATION
25        040000                    DU.TER  = 40000                          ;TERMINATOR
26        050000                    DU.FTL  = 50000                          ;FATAL ERROR
27        060000                    DU.SPC  = 60000                          ;SPECIAL
```

```
    1                              .SBTTL   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      000077             CT.UNT  = 000077                ;LOGICAL UNIT NUMBER MASK
   11      000777             CT.VEC  = 000777                ;VECTOR ADDRESS MASK
   12      007000             CT.BRL  = 007000                ;BR LEVEL MASK
   13
   14      100000             CT.AVL  = BIT15                 ;SET WHEN NOT AVAILABLE FOR TESTING
   18      000040             CT.U50  = BIT5                  ;CONTROLLER IS UDA50 IF SET/UDA52 IF CLEARED
   19      000020             CT.REQ  = BIT4                  ;BUFFER HAS BEEN GIVEN TO UDA FOR REQUEST
   20                                                         ;SET WHENEVER READ STUD DATA COMMAND GIVEN TO UDA
   21      000010             CT.MSG  = BIT3                  ;MESSAGE RESPONSE RECEIVED
   22                                                         ;WHENEVER THIS BIT IS SET, CT.CMD IS CLEARED
   23      000002             CT.RN   = BIT1                  ;DM PROGRAM RUNNING
   24      000004             CT.CMD  = BIT2                  ;COMMAND ISSUED, WAITING FOR RESPONSE
   25
   27      000000             C.UADR  = 0                     ;UNIBUS ADDRESS OF UDAIP REGISTER
   28      000002             C.UNIT  = 2                     ;UNIT NUMBER TO TEST
   29      000004             C.VEC   = 4                     ;VECTOR ADDRESS/BR LEVEL
   30      000006             C.BST   = 6                     ;BURST LEVEL
   31      000010             C.JSR   = 10                    ;INTERRUPT SERVICE ROUTINE FOR CONTROLLER
   32      000012             C.JAD   = 12                    ;THESE TWO WORDS LOADED WITH [JSR R0     UDASRV]
   33      000014             C.FLG   = 14                    ;FLAGS
   34      000016             C.HCOM  = 16                    ;BEGINNING ADRS OF HOST COMM AREA IN MEMORY
   35      000020             C.DR0   = 20                    ;POINTER TO DRIVE TABLES
   36      000022             C.DR1   = 22                    ;IF ZERO, NO DRIVE TABLE EXISTS
   37      000024             C.DR2   = 24                    ;
   38      000026             C.DR3   = 26                    ;
   39      000030             C.DR4   = 30                    ;
   40      000032             C.DR5   = 32                    ;
   41      000034             C.DR6   = 34                    ;
   42      000036             C.DR7   = 36                    ;
   43      000040             C.TO    = 40                    ;TIMEOUT COUNTER
   44      000042             C.TOH   = 42                    ;   (TWO WORDS)
   45      000044             C.REF   = 44                    ;COMMAND REFERENCE NUMBER
   46
   47      000046             C.SIZE  = 46                    ;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                                         ;
    8                                         ;THE FIRST TABLE IS POINTED TO BY THE CONTENTS OF DTABS.
    9
   10        000077                  DT.UNT  = 000077                      ; LOGICAL UNIT NUMBER OF DRIVE
   11
   12        100000                  DT.AVL  = BIT15                       ; SET WHEN NOT AVAILABLE FOR TESTING
   13        040000                  D.IW    = BIT14                       ;INITIAL WRITE
   14        020000                  D.DCY   = BIT13                       ;DIAGNOSTIC CYLINDERS
   15        010000                  D.ECC   = BIT12                       ;ECC CORRECTION ENABLED
   16        004000                  D.RO    = BIT11                       ;READ ONLY
   17        002000                  D.WO    = BIT10                       ;WRITE ONLY
   18        001000                  D.RET   = BIT9                        ;RETRIES ENABLED
   19        000400                  D.CYL   = BIT8                        ;START/END CYLINDERS SPECIFIED
   20        000100                  D.SEQ   = BIT6                        ;SEQUENTIAL ACCESS
   21        000040                  D.BE    = BIT5                        ;BEGIN/END BLOCKS USED
   22        000020                  D.TR    = BIT4                        ;WHEN D.BE=0:  1   TRACKS, 0 - GROUPS
   23        000010                  D.WC    = BIT3                        ;WRITE CHECKS ENABLED
   24        000004                  D.WCA   = BIT2                        ;ALWAYS WRITE CHECK
   25        000002                  D.DC    = BIT1                        ;DATA COMPARES ENABLED
   26        000001                  D.DCA   = BIT0                        ;ALWAYS DATA COMPARE
   30        011012                  DDEF    = D.ECC+D.WC+D.DC+D.RET       ;DEFAULT D.PRM
   32        140200                  D.ZERO  = BIT15+BIT7+D.IW            ;BITS TO BE CLEARED
   33
   35        000000                  D.DRV   = 0                          ;DRIVE NUMBER
   36        000002                  D.UNIT  = 2                          ;
   37        000004                  D.PRM   = 4                          ;HARDWARE QUESTION FLAGS
   38        000006                  D.PAT   = 6                          ;DATA PATTERN NUMBER
   39        000010                  D.BB    = 10                         ;BAD BLOCK COUNT
   40        000012                  D.BB01  = 12                         ;BAD BLOCK 1
   41        000016                  D.BB02  = 16                         ;          2
   42        000022                  D.BB03  = 22                         ;          3
   43        000026                  D.BB04  = 26                         ;          4
   44        000032                  D.BB05  = 32                         ;          5
   45        000036                  D.BB06  = 36                         ;          6
   46        000042                  D.BB07  = 42                         ;          7
   47        000046                  D.BB08  = 46                         ;          8
   48        000052                  D.BB09  = 52                         ;          9
   49        000056                  D.BB10  = 56                         ;          10
   50        000062                  D.BB11  = 62                         ;          11
   51        000066                  D.BB12  = 66                         ;          12
   52        000072                  D.BB13  = 72                         ;          13
   53        000076                  D.BB14  = 76                         ;          14
   54        000102                  D.BB15  = 102                        ;          15
   55        000106                  D.BB16  = 106                        ;          16
```

```
    1
    2         000112              D.BEC    = 112              ;BEGIN/END SET COUNT
    3         000114              D.BGN1   = 114              ;BEGIN BLOCK 1
    4         000120              D.END1   = 120              ;END
    5         000124              D.BGN2   = 124              ;BEGIN BLOCK 2
    6         000130              D.END2   = 130              ;END
    7         000134              D.BGN3   = 134              ;BEGIN BLOCK 3
    8         000140              D.END3   = 140              ;END
    9         000144              D.BGN4   = 144              ;BEGIN BLOCK 4
   10         000150              D.END4   = 150              ;END
   11         000154              D.BCYL   = 154              ;BEGIN CYLINDER
   12         000160              D.ECYL   = 160              ;END CYLINDER
   13         000164              D.XFRW   = 164              ;MEGABITS WRITTEN COUNT
   14         000166              D.XFRR   = 166              ;MEGABITS READ COUNT
   15         000170              D.HERR   = 170              ;HARD ERROR COUNTER
   16         000172              D.SERR   = 172              ;SOFT ERROR COUNTER
   17         000174              D.SEEK   = 174              ;NUMBER OF SEEKS X1000
   18         000176              D.ECCC   = 176              ;ECC COUNTER
   19         000200              D.SERN   = 200              ;DRIVE SERIAL NUMBER
   24
   25         000206              D.SIZE   = 206              ;SIZE OF DRIVE TABLE IN BYTES
   26
   27                                      ;DM PROGRAM HEADER DEFINITIONS
   28
   29         000000              DMTRLN   = 0                ;OFFSET TO SIZE OF PROGRAM NEEDING DOWNLINE LOAD
   30         000004              DMOVRL   = 4                ;OFFSET TO SIZE OF OVERLAY
   31         000040              DMMAIN   = 40               ;OFFSET TO FIRST WORD OF MAIN PROGRAM
   32         001000              DMFRST   = 1000             ;ADDRESS IN DM FILE CONTAINING FIRST BYTE OF HEADER
```

```
 1                                    .SBTTL   GLOBAL DATA SECTION
 2
 3                                    ;..
 4                                    ; THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
 5                                    ; IN MORE THAN ONE TEST.
 6                                    ;
 7
 8 064402                            L$ERRTBL::
   064402  000000                    ERRTYP::          .WORD   0
   064404  000000                    ERRNBR::          .WORD   0
   064406  000000                    ERRMSG::          .WORD   0
   064410  000000                    ERRBLK::          .WORD   0
 9
10 064412                            FFREE:: .BLKW 1                          ;FIRST FREE WORD IN MEMORY
11 064414                            FSIZE:: .BLKW 1                          ;SIZE OF FREE MEMORY IN WORDS
12 064416                            FMEM:   .BLKW 1                          ;COPY OF FFREE AT END OF INIT SECTION
13 064420                            FMEMS:  .BLKW 1                          ;COPY OF FSIZE AT END OF INIT SECTION
14 064422                            DTABS:: .BLKW 1                          ;START OF DRIVE TABLE STORAGE
15 064424                            CTABS:: .BLKW 1                          ;START OF CONTROLLER TABLE STORAGE
16 064426                            CTRLRS: .BLKW 1                          ;COUNT OF UDA CONTROLLERS IN PTABLES
17 064430                            TSTTAB: .BLKW 1                          ;POINTER TO 1ST CONTROLLER TABLE UNDER TEST
18 064432                            DMPROG: .BLKW 1                          ;START ADDRESS OF UDA52 DM PROGRAM
19
20 064434                            KTBASA: .BLKW 1                          ;HIGH TWO BYTES OF BASE ADDRESS FOR KT ACCESS
21 064436                            KTBASC: .BLKW 1                          ;LOW BYTE OF ADDRESS FOR KT ACCESS
22
23 064440                            IFLAGS::.BLKW 1                          ;FLAGS FROM INIT CODE FOR TEST 4
24
25         000002                    ICONT   == BIT1                         ; CONTINUE EVENT FLAG
26         000004                    IREST   == BIT2                         ; RESTART FLAG
27         000010                    ISTRT   == BIT3                         ; START FLAG
28         000020                    ISTRTH  == BIT4                         ; START FLAG HOLD FOR T4UPRM ROUTINE
29
30 064442  000000                    FNUM:   .WORD 0                         ;FILE # IN PAK FILE THAT IS CURRENTLY LOADED
31 064444  000000                    TNUM:   .WORD 0                         ;NUMBER OF TEST EXECUTING
32 064446                            URUN:   .BLKW 1                         ;NUMBER OF UNITS TO RUN AT ONE TIME
33 064450                            URNING: .BLKW 1                         ;NUMBER OF UNITS STILL RUNNING
34 064452                            UCNT:   .BLKW 1                         ;COUNTER OF UNITS UNDER TEST
35 064454                            INTRCV: .BLKW 1                         ;INTERRUPT RECEIVED FLAG FOR INT TESTING
```

F11

```
 1
 5
 7 064456    132    125    104  FNAME:   .ASCIZ  \ZUDDEO.PAK\              ;NAME OF DATA FILE
 8                                        .EVEN
 9
10 064472  000000              FDATA:   .WORD 0
11 064474  000000              FILOPN:  .WORD 0                           ;FILE OPEN WHEN NON ZERO
12 064476                      TEMP:    .BLKW 12.                         ;TEMPORY STORAGE FOR GMANI RESPONSES
13
14 064526    125    065    262  U52EXT:  .ASCII  "U52"
15                                        .EVEN
19
20 064532  000000              TYPCNT:  .WORD   0                        ; TYPE OF CONTROLLER WORD
21
22         000002              TY.U50  = BIT1
23         000001              TY.U52  = BIT0
24
25 064534  000000              IPADRS:  .WORD   0                        ; EIGHT ENTRIES
26 064536  000000                       .WORD   0
27 064540  000000                       .WORD   0
28 064542  000000                       .WORD   0
29 064544  000000                       .WORD   0
30 064546  000000                       .WORD   0
31 064550  000000                       .WORD   0
32 064552  000000                       .WORD   0
33
34 064554  000001              PAT16C:  .WORD 1                          ;COUNT OF WORDS IN DATA PATTERN 16
35 064556  000000              PAT16W:  .WORD 0                          ;WORD SEQUENCE FOR DATA PATTERN 16
36 064560  000000                       .WORD 0
37 064562  000000                       .WORD 0
38 064564  000000                       .WORD 0
39 064566  000000                       .WORD 0
40 064570  000000                       .WORD 0
41 064572  000000                       .WORD 0
42 064574  000000                       .WORD 0
43 064576  000000                       .WORD 0
44 064600  000000                       .WORD 0
45 064602  000000                       .WORD 0
46 064604  000000                       .WORD 0
47 064606  000000                       .WORD 0
48 064610  000000                       .WORD 0
49 064612  000000                       .WORD 0
50 064614  000000                       .WORD 0
```

```
  1
  2                                    ;KW11 CLOCK CONTROL
  3
  4 064616    000000         KW.CSR: .WORD 0               ;CSR OF CLOCK
  5 064620                   KW.BRL: .BLKW 1               ;BR LEVEL
  6 064622                   KW.VEC: .BLKW 1               ;VECTOR
  7 064624                   KW.HZ:  .BLKW 1               ;HERTZ (50. OR 60.)
  8 064626                   KW.EL:  .BLKW 2               ;ELAPSED TIME
  9 064632                   STIME:  .BLKW 2               ;STATISTICAL REPORT TIMER
 10
 11 064636                   NXMAD:  .BLKW 1               ;SET TO ALL ONES BY NON EXISTANT ADDRESS
 12 064640    177777         KTMEM:  .WORD  1               ;SET TO ALL ONES IF NO KT EXISTS
 13
 14 064642                   T2WRR:  .BLKW 1               ;WRITE/READ REGION
 15 064644                   T2WRO:  .BLKW 1               ;WRITE/READ OFFSET
 16 064646                   T2DR:   .BLKW 1               ;DIAGNOSE REGION
 17
 18
 19                                    ;ERROR LOG CONTROL WORDS
 20
 21 064650                   LBUFS:  .BLKW 1               ;START ADDRESS OF LOG/ZERO IF NONE
 22 064652                   LBUFN:  .BLKW 1               ;ADDRESS FOR MORE DATA FOR LOG
 23 064654                   LBUFE:  .BLKW 1               ;LAST ADDRESS AVAILABLE FOR LOG DATA
 24
 25                                    ;DISK DIAGNOSTIC DLL CONTROL WORDS
 26
 27 064656                   DLL:    .BLKW 1               ;DOWNLINE LOAD RESPONSE CODE = 0   NO DATA,
 28                                                        ;1   PROGRAM PROVIDED, 2   PROGRAM NOT FOUND
 29 064660                   DLLDR:  .BLKW 1               ;DRIVE NUMBER REQUESTING PROGRAM
 30 064662                   DLLV:   .BLKW 1               ;A VALUE FROM DM PROGRAM TO BE RETURNED
 31 064664                   DLLR:   .BLKW 1               ;REGION
 32 064666                   DLLADR: .BLKW 2               ;ADDRESS WHERE PROGRAM STORED
 33 064672                   DLLSIZ: .BLKW 1               ;SIZE OF PROGRAM IN BYTES
 34 064674                   DLLNAM: .BLKW 2               ;NAME OF PROGRAM IN RAD50
```

```
 1                                      .SBTTL   GLOBAL TEXT SECTION
 2
 3                                     ;**
 4                                     ; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
 5                                     ; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
 6                                     ; MORE ThAN ONE TEST.
 7                                     ;
 8
12                                     ;NAMES OF DEVICES SUPPORTED BY PROGRAM
13                                     ;
14 064700                             L$DVTYP::
   064700       114      117      107          .ASCIZ   /LOGICAL DISK DRIVE/
                                                .EVEN
15
21
22                                     ; TEST DESCRIPTION
23                                     ;
25 064724                             L$DESC::
   064724       103      132      125          .ASCIZ   /CZUDCEO UDA & DISK DRV DIAG/
                                                .EVEN
34
41
43
44                                     ; UNFORMATTED MESSAGES
45                                     ;
46
47 064760       040      040      000 T4OPT7: .ASCIZ  \ \
48 064763       101      122      105 INITWC: .ASCIZ  \ARE YOU SURE CUSTOMER DATA CAN BE DESTROYED\
50
51                                     ; FORMAT STATEMENTS USED IN PRINT CALLS
52                                     ;
53
54 065037       045      124      000 FRMTT:  .ASCIZ  \%T\
55 065042       045      116      000 CRLF:   .ASCIZ  \%N\
56 065045       042      040      040 RNTIM:  .ASCIZ  \"  RUNTIME "D16':"\
57 065070       104      071      042 RNTIM1: .ASCIZ  \D9 : \
58 065076       104      071      000 RNTIM2: .ASCIZ  \D9\
59 065101       042      040      040 ERRME1: .ASCIZ  \" * * * ERROR PROCESSING MESSAGE STRING * * *'N\
60 065170       116      042      122 MXFERP: .ASCIZ  \N"REACHED TRANSFER LIMIT   TESTING STOPPED'N\
61 065245       116      042      125 ERRLIM: .ASCIZ  \N"UNIT "D6" REACHED ERROR LIMIT   WILL NO LONGER BE TESTED 'N\
62 065342       116      042      124 INTSTO: .ASCIZ  \N"TESTING INTERRUPT ABILITY OF UDA AT ADR "O16' VEC "O9 ... \
63 065437       042      103      117 INTST1: .ASCIZ  \"COMPLETED"N\
64 065454       116      042      103 INITWA: .ASCIZ  \N"CUSTOMER DATA WILL BE DESTROYED ON:"NS5"UNIT"S5'UDA AT"S3"DRIVE N\
65 065560       045      123      066 INITWB: .ASCIZ  \%S6%D2%S6%06%S4%D3%N\
66 065605       116      042      115 T4WARN: .ASCIZ  \N"MANUAL INTERVENTION NOT ALLOWED.  TEST 4 USING DEFAULT PARAMETERS 'N\
67 065713       116      042      125 MESSG:  .ASCIZ  \N"UNIT "D6" UDA AT "O16" DRIVE   D9S\
68 065757       116      042      115 T2WARN: .ASCIZ  \N"MANUAL INTERVENTION NOT ALLOWED. TEST 2 RUNNING UNATTENDED N\
69 066056       116      042      124 T2CMS1: .ASCII  \N"TEST #2 MANUAL INTERVENTION ON UNIT 'D8  UDA AT 'O16  DRIVE  D9N\
70 066160       042      124      117         .ASCII  \"TO WRITE AND READ MEMORY:"N\
71 066214       042      040      040         .ASCII  \"  W DATA REGION OFFSET 'N\
72 066245       042      040      040         .ASCII  \"  R REGION OFFSET'N\
73 066271       042      124      117         .ASCII  \"TO RUN A DIAGNOSTIC:"N\
74 066320       042      040      040         .ASCII  \"  D REGION"N\
75 066335       042      124      117         .ASCII  \"TO EXIT QUESTIONING:'N\
76 066364       042      040      040         .ASCII  \"  E"N\
77 066372       042      104      101         .ASCIZ  \"DATA, REGION AND OFFSET ARE HEX VALUES. N\
78 066445       042      077      040 T2CMS5: .ASCIZ  \"? INPUT ERRCR N\
```

```
79 066466    .042    116    117   NOCLOCK:.ASCIZ   \"NO LINE CLOCK AVAILABLE FOR TIMING EVENTS"N\
80 066543    .116    042    103   LOGM1:  .ASCIZ   \N"CONTENTS OF ERROR LOG:'\
81 066575    116    042    105   LOGM2:  .ASCIZ   \N"END OF ERROR LOG"N\
82 066622    116    042    105   LOGM3:  .ASCIZ   \N"ERROR LOG IS EMPTY"N\
83
84 066651    042    110    117   BASN0:  .ASCIZ   \"HOST PROGRAM"\
86 066670    042    125    116   BASN1:  .ASCIZ   \"UNIBUS ADDRESSING"\
87 066714    042    104    111   BASN2:  .ASCIZ   \"DISK RESIDENT"\
88 066734    042    104    111   BASN3:  .ASCIZ   \"DISK FUNCTION"\
91 066754    042    104    111   BASN4:  .ASCIZ   \"DISK EXERCISER'\
93 066775    042    040    040   BASL1:  .ASCIZ   \"   DM PC:"012\
94 067013    042    040    040   BASL2:  .ASCIZ   \"   UDA AT "016\
95 067032    042    040    040   BASL3:  .ASCIZ   \"   DRIVE "D9\
96 067047    000                 BAS:    .BYTE 0                    ;NULL TO PRINT NOTHING
97
98 067050    122    066    122   BASLN:  .ASCIZ   \R6R6R6R6\        ;USED TO PRINT BASIC LINE OF ERROR MESSAGE
```

```
   1 067061                              X1A:
   2 067061                              X2A:
   3 067061                              X3A:
   4 067061    042  111  040   X8A:      .ASCIZ  \"I DON'T LIKE THE ANSWERS YOU GAVE TO THE HARDWARE QUESTIONS"N\
   5 067160    122  065  122   X1:       .ASCIZ  \R5R6"UDA HAS MORE THAN ONE VECTOR, BR LEVEL OR BURST RATE"N\
   6 067254    122  065  122   X2:       .ASCIZ  \R5R6"TWO UNITS SELECT THE SAME DRIVE"N\
   7 067323    122  065  122   X3:       .ASCIZ  \R5R6"MORE THAN EIGHT DRIVES SELECTED ON THIS UDA"N\
   8 067406    122  064  042   X4:       .ASCII  \R4"NOT ENOUGH ROOM IN MEMORY TO TEST THE UNITS SELECTED"N\
   9 067477    042  120  114             .ASCIZ  \"PLEASE START PROGRAM OVER AND TEST FEWER UNITS AT A TIME"N\
  10 067573    122  064  042   X6:       .ASCIZ  \R4"TABLE CONSISTANCY ERROR. PLEASE RE-LOAD PROGRAM"N\
  11 067660    122  065  122   X8:       .ASCIZ  \R5R6"TWO UDA'S USE THE SAME VECTOR"N\
  13 067725    122  064  042   X5:       .ASCIZ  \R4"CHECKSUM ERROR IN DM PROGRAM FILE "N\
  14 067775    122  064  042   X7:       .ASCIZ  \R4"ERROR IN DM PROGRAM FILE. DM PROGRAM NOT FOUND"N\
  16 070061    122  064  042   X14:      .ASCII  \R4"UDA50 CONTROLLER IS AT A REVISION LEVEL NO LONGER SUPPORTED"N\
  17 070161    042  102  131             .ASCII  \"BY THIS DIAGNOSTIC PROGRAM.  THIS PROGRAM REQUIRES A UDA50-A"N\
  18 070260    042  103  117             .ASCII  \"CONTROLLER (MODEL 6) WITH MICROCODE REVISION AT 3 OR GREATER."N\
  19 070360    116  042  103             .ASCII  \N"CONTROLLER REPORTED MODE CODE "D4" AND MICROCODE VERSION "D4N\
  35 070460    122  065  042   X38:      .ASCII  \R5"MEMORY ERROR TRYING TO READ UDA REGISTERS"N\
  36 070536    042  103  110     ♦       .ASCII  \"CHECK UNIBUS SELECTION SWITCHES ON UDA MODULE M7485'N\
  37 070624    042  117  122             .ASCII  \"OR UNIBUS"N\
  38 070640    042  117  122             .ASCIZ  \"OR "R7\
  39 070650    122  065  042   X21:      .ASCII  \R5"UDA RESIDENT DIAGNOSTICS DETECTED FAILURE'NR8\
  40 070730    042  122  105             .ASCIZ  \"REPLACE UDA MODULE M748"03N\
  41 070765    122  065  042   X22:      .ASCII  \R5"STEP BIT DID NOT SET IN UDASA REGISTER DURING INITIALIZATION"N\
  42 071066    042  123  124             .ASCIZ  \"STEP BIT EXPECTED "016NR8R7\
  43 071123    122  065  042   X23A:     .ASCII  \R5"UDA DID NOT CLEAR RING STRUCTURE IN HOST MEMORY DURING INITIALIZATION'N\
  44 071235    104  071  042             .ASCII  \D9" WORDS WERE TO BE CLEARED STARTING AT ADDRESS "016N\
  45 071323    042  106  111             .ASCII  \"FIRST SEVERAL WORDS NOT CLEARED (UP TO 6):"N\
  46 071400    123  066  042             .ASCIZ  \S6"ADDRESS"S4"CONTENTS"N\
  47 071431    123  067  117   X23B:     .ASCIZ  \S7016S5016N\
  48 071445    122  065  042   X24:      .ASCII  \R5"UDASA REGISTER DID NOT GO TO ZERO AFTER STEP 3 WRITE OF INITIALIZATION"N
\
  49 071560    042  120  125             .ASCIZ  \"PURGE/POLE DIAGNOSTICS WERE REQUESTED"NR8R7\
  50 071635    122  065  042   X25:      .ASCII  \R5"UDA DID NOT RETURN CORRECT DATA IN UDASA REGISTER DURING INITIALIZATION"
N\
  51 071751    042  040  040             .ASCIZ  \"  UDASA EXPECTED  "016NR8R7\
  53 072006    122  065  042   X26:      .ASCII  \R5"DATA COMPARISON ERROR DURING DIAGNOSTIC PORT LOOP TEST"N\
  54 072101    042  040  040             .ASCII  \"  DATA SENT TO UDASA  "016N\
  55 072135    042  040  040             .ASCIZ  \"  RECEIVED FROM UDASA "016NR7\
  56 072174    122  065  042   X27:      .ASCII  \R5"UDASA REGISTER DID NOT CHANGE AFTER WRITING TO IT"N\
  57 072262    042  111  116             .ASCIZ  \"IN PORT LOOP DIAGNOSTIC"NR8R7\
  58 072321    122  065  042   X28:      .ASCIZ  \R5"UDA DID NOT INTERRUPT THE PDP 11"NR7\
  59 072371    122  065  042   X29:      .ASCII  \R5"UDA INTERRUPTED AT DIFFERENT BR LEVEL THAN SPECIFIED IN HARDWARE"N\
  60 072476    042  121  125             .ASCII  \"QUESTIONS. INTERRUPT WAS AT BR LEVEL "03N\
  61 072550    042  103  110             .ASCII  \"CHECK PRIORITY PLUG ON UDA MODULE M7485"N\
  62 072622    042  117  122             .ASCIZ  \"OR CHANGE HARDWARE QUESTIONS"N\
  64 072662    122  065  042   X30:      .ASCIZ  \R5"UDA REPORTED FATAL ERROR IN UDASA REGISTER WHILE RUNNING DM PROGRAM NR8\
  65 072775    122  065  042   X31:      .ASCII  \R5"NO INTERRUPT RECEIVED FROM DM PROGRAM FOR 3 MINUTES'N\
  66 073065    042  101  123             .ASCIZ  \"ASSUME PROGRAM IS HUNG"N\
  67 073117    122  065  042   X32:      .ASCIZ  \R5"MESSAGE BUFFER RECEIVED FROM DM PROGRAM WITH UNKNOWN REQUEST NUMBER N\
  68 073230    122  065  042   X35:      .ASCIZ  \R5"DM PROGRAM ASKED FOR DATA ON UNKNOWN DRIVE'N\
  69 073310    122  065  042   X36:      .ASCII  \R5"NO INTERRUPT RECEIVED FROM UDA FOR 30 SECONDS"N\
  70 073372    042  127  110             .ASCIZ  \"WHILE LOADING DM PROGRAM'N\
  71 073426    122  065  042   X37:      .ASCIZ  \R5"UDA REPORTED FATAL ERROR IN UDASA REGISTER WHILE LOADING DM PROGRAM NR8R
7\
```

```
   1 073543      042      115      105  XMSG1:  .ASCIZ  \"MESSAGE BUFFER CONTAINS:"N\
   2 073577      123      063      117  XMSG2:  .ASCIZ  \S3016S1016S1016S1016S1016S1016S1016N\
   3 073644      122      065      042  XPKT1:  .ASCII  \R5"RESPONSE PACKET FROM UDA DOES NOT CONTAIN  XPECTED DATA"N\
   4 073740      042      105      111          .ASCII  \"EITHER UDA RETURNED ERROR STATUS OR PACKET WAS NOT RECEIVED CORRECTLY"N\
   5 074050      123      063      042          .ASCIZ  \S3"COMMAND PACKET SENT"S6"RESPONSE PACKET RECEIVED"N\
   6 074135      123      066      117  XPKT2:  .ASCIZ  \S6016S1016S14016S1016N\
   7 074164      042      040      040  XSA:    .ASCIZ  \"  UDASA CONTAINS  "016N\
   8 074215      042      122      105  XFRU:   .ASCIZ  \"REPLACE UDA MODULE M7485"N\
  12                                            .EVEN
```

```
   1                              .SBTTL  GLOBAL ERROR REPORT SECTION
   2
   3                          ;++
   4                          ; THE GLOBAL ERROR REPORT SECTION CONTAINS MESSAGE PRINTING AREAS
   5                          ; USED BY MORE THAN TEST TO OUTPUT ADDITIONAL ERROR INFORMATION.  PRINTB
   6                          ; (BASIC) AND PRINTX (EXTENDED) CALLS ARE USED TO CALL PRINT SERVICES.
   7                          ;--
  25
  26 074252                  ERR001::
  27 074252 012746 067061            MOV     #X1A,-(SP)      ;PUSH #X1A ON STACK
     074256 004137 075674            JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
     074262 067160                   .WORD   X1              ;ADDRESS OF ASCIZ STRING
     074264 000002                   .WORD   ARG.CT          ;ARGUMENT COUNT * 2
  28 074266                  L10002:
     074266 104423                   TRAP    C$MSG
  29
  30 074270                  ERR002::
  31 074270 012746 067061            MOV     #X2A,-(SP)      ;PUSH #X2A ON STACK
     074274 004137 075674            JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
     074300 067254                   .WORD   X2              ;ADDRESS OF ASCIZ STRING
     074302 000002                   .WORD   ARG.CT          ;ARGUMENT COUNT * 2
  32 074304                  L10003:
     074304 104423                   TRAP    C$MSG
  33
  34 074306                  ERR003::
  35 074306 012746 067061            MOV     #X3A,-(SP)      ;PUSH #X3A ON STACK
     074312 004137 075674            JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
     074316 067323                   .WORD   X3              ;ADDRESS OF ASCIZ STRING
     074320 000002                   .WORD   ARG.CT          ;ARGUMENT COUNT * 2
  36 074322                  L10004:
     074322 104423                   TRAP    C$MSG
  37
  38 074324                  ERR004::
  39 074324 004137 075674            JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
     074330 067406                   .WORD   X4              ;ADDRESS OF ASCIZ STRING
     074332 000000                   .WORD   ARG.CT          ;ARGUMENT COUNT * 2
  40 074334                  L10005:
     074334 104423                   TRAP    C$MSG
  41
  43 074336                  ERR005::
  44 074336 004137 075674            JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
     074342 067725                   .WORD   X5              ;ADDRESS OF ASCIZ STRING
     074344 000000                   .WORD   ARG.CT          ;ARGUMENT COUN * 2
  45 074346                  L10006:
     074346 104423                   TRAP    C$MSG
  46
  47 074350                  ERR007::
  48 074350 004137 075674            JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
     074354 067775                   .WORD   X7              ;ADDRESS OF ASCIZ STRING
     074356 000000                   .WORD   ARG.CT          ;ARGUMENT COUNT * 2
  49 074360                  L10007:
     074360 104423                   TRAP    C$MSG
  50
  52 074362                  ERR014::
  53 074362 010146                   MOV     R1, (SP)        ;PUSH R1 ON STACK
     074364 010346                   MOV     R3,-(SP)        ;PUSH R3 ON STACK
     074366 004137 075674            JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
```

M11

CZUDCEO UDA & DISK DRV DIAG MACRO V05.00 Wednesday 04 Jan 84 16:12 Page 106 1
GLOBAL ERROR REPORT SECTION   SFQ 0142   CZ
                                                                                                                          GL

```
      074372  070061                              .WORD   X14             ;ADDRESS OF ASCIZ STRING
      074374  000004                              .WORD   ARG.CT          ;ARGUMENT COUNT * 2
54 074376                               L10010:
      074376  104423                              TRAP    C$MSG
55
56 074400                               ERR006::
57 074400  004137  075674                         JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
      074404  067573                              .WORD   X6              ;ADDRESS OF ASCIZ STRING
      074406  000000                              .WORD   ARG.CT          ;ARGUMENT COUNT * 2
58 074410                               L10011:
      074410  104423                              TRAP    C$MSG
59
60 074412                               ERR008::
61 074412  012746  067061                         MOV     #X8A,-(SP)      ;PUSH #X8A ON STACK
      074416  004137  075674                       JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
      074422  067660                              .WORD   X8              ;ADDRESS OF ASCIZ STRING
      074424  000002                              .WORD   ARG.CT          ;ARGUMENT COUNT * 2
62 074426                               L10012:
      074426  104423                              TRAP    C$MSG
63
74
75 074430                               ERR021::
76 074430  010201                               MOV     R2,R1
77 074432  000301                               SWAB R1
78 074434  042701  177775                        BIC     #^C<2>,R1
79 074440  006201                               ASR     R1
80 074442  005201                               INC     R1
81 074444  010103                               MOV     R1,R3
82 074446  062703  000004                        ADD     #4,R3
83 074452  010346                               MOV     R3,-(SP)        ;PUSH R3 ON STACK
      074454  010246                              MOV     R2,-(SP)        ;PUSH R2 ON STACK
      074456  004137  075674                      JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
      074462  070650                              .WORD   X21             ;ADDRESS OF ASCIZ STRING
      074464  000004                              .WORD   ARG.CT          ;ARGUMENT COUNT * 2
84 074466                               L10013:
      074466  104423                              TRAP    C$MSG
85
86 074470                               ERR022::
87 074470  042737  100000  105546         BIC     #SA.ERR,UDARSD
88 074476  010246                               MOV     R2, (SP)        ;PUSH R2 ON STACK
      074500  013746  105546                      MOV     UDARSD,-(SP)    ;PUSH UDARSD ON STACK
      074504  004137  075674                      JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
      074510  070765                              .WORD   X22             ;ADDRESS OF ASCIZ STRING
      074512  000004                              .WORD   ARG.CT          ;ARGUMENT COUNT * 2
89 074514                               L10014:
      074514  104423                              TRAP    C$MSG
90
91 074516                               ERR023::
92 074516  013746  064412                         MOV     FFREE, (SP)     ;PUSH FFREE ON STACK
      074522  010146                              MOV     R1, (SP)        ;PUSH R1 ON STACK
      074524  004137  075674                      JSR     R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
      074530  071123                              .WORD   X23A            ;ADDRESS OF ASCIZ STRING
      074532  000004                              .WORD   ARG.CT          ;ARGUMENT COUNT * 2
93 074534  005742                               TST     -(R2)
94 074536  005712                       ERR23A:  TST     (R2)
95 074540  001410                               BEQ     ERR23B
96 074542  011246                               MOV     (R2), (SP)      ;PUSH (R2) ON STACK
```

N11

CZUDCEO UDA & DISK DRV DIAG MACRO V05.00 Wednesday 04 Jan-84 16:12 Page 106 2   SFQ 0143   CZU
GLOBAL ERROR REPORT SECTION   GLC

```
        074544  010246                  MOV     R2,-(SP)          ;PUSH R2 ON STACK
        074546  004137  075674          JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
        074552  071431                  .WORD   X23B              ;ADDRESS OF ASCIZ STRING
        074554  000004                  .WORD   ARG.CT            ;ARGUMENT COUNT * 2
 97     074556  005304                  DEC     R4
 98     074560  001403                  BEQ     ERR23C
 99     074562  005722          ERR23B: TST     (R2)+
100     074564  005303                  DEC     R3
101     074566  001363                  BNE     ERR23A
102     074570                  ERR23C:
        074570  004137  075674          JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
        074574  074215                  .WORD   XFRU              ;ADDRESS OF ASCIZ STRING
        074576  000000                  .WORD   ARG.CT            ;ARGUMENT COUNT * 2
103     074600                  L10015:
        074600  104423                  TRAP    C$MSG
104
105     074602                  ERR024::
106     074602  010246                  MOV     R2,-(SP)          ;PUSH R2 ON STACK
        074604  004137  075674          JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
        074610  071445                  .WORD   X24               ;ADDRESS OF ASCIZ STRING
        074612  000002                  .WORD   ARG.CT            ;ARGUMENT COUNT * 2
107     074614                  L10016:
        074614  104423                  TRAP    C$MSG
108
109     074616                  ERR025::
110     074616  010246                  MOV     R2, (SP)          ;PUSH R2 ON STACK
        074620  010146                  MOV     R1, (SP)          ;PUSH R1 ON STACK
        074622  004137  075674          JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
        074626  071635                  .WORD   X25               ;ADDRESS OF ASCIZ STRING
        074630  000004                  .WORD   ARG.CT            ;ARGUMENT COUNT * 2
111     074632                  L10017:
        074632  104423                  TRAP    C$MSG
112
114     074634                  ERR026::
115     074634  016446  000002          MOV     2(R4), (SP)       ;PUSH 2(R4) ON STACK
        074640  010246                  MOV     R2, (SP)          ;PUSH R2 ON STACK
        074642  004137  075674          JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
        074646  072006                  .WORD   X26               ;ADDRESS OF ASCIZ STRING
        074650  000004                  .WORD   ARG.CT            ;ARGUMENT COUNT * 2
116     074652                  L10020:
        074652  104423                  TRAP    C$MSG
117
118     074654                  ERR027::
119     074654  016446  000002          MOV     2(R4), (SP)       ;PUSH 2(R4) ON STACK
        074660  004137  075674          JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
        074664  072174                  .WORD   X27               ;ADDRESS OF ASCIZ STRING
        074666  000002                  .WORD   ARG.CT            ;ARGUMENT COUNT * 2
120     074670                  L10021:
        074670  104423                  TRAP    C$MSG
121
122     074672                  ERR028::
123     074672  004137  075674          JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
        074676  072321                  .WORD   X28               ;ADDRESS OF ASCIZ STRING
        074700  000000                  .WORD   ARG.CT            ;ARGUMENT COUNT * 2
124     074702                  L10022:
        074702  104423                  TRAP    C$MSG
125
```

```
126 074704                      ERR029::
127 074704  010146                      MOV     R1,-(SP)          ;PUSH R1 ON STACK
    074706  004137   075674             JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
    074712  072371                      .WORD   X29               ;ADDRESS OF ASCIZ STRING
    074714  000002                      .WORD   ARG.CT            ;ARGUMENT COUNT • 2
128 074716                      L10023:
    074716  104423                      TRAP    C$MSG
130
131 074720                      ERR030::
132 074720  010146                      MOV     R1,-(SP)          ;PUSH R1 ON STACK
    074722  004137   075674             JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
    074726  072662                      .WORD   X30               ;ADDRESS OF ASCIZ STRING
    074730  000002                      .WORD   ARG.CT            ;ARGUMENT COUNT • 2
133 074732                      L10024:
    074732  104423                      TRAP    C$MSG
134
135 074734                      ERR031::
136 074734  004137   075674             JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
    074740  072775                      .WORD   X31               ;ADDRESS OF ASCIZ STRING
    074742  000000                      .WORD   ARG.CT            ;ARGUMENT COUNT • 2
137 074744                      L10025:
    074744  104423                      TRAP    C$MSG
138
139 074746                      ERR032::
140 074746  004137   075674             JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
    074752  073117                      .WORD   X32               ;ADDRESS OF ASCIZ STRING
    074754  000000                      .WORD   ARG.CT            ;ARGUMENT COUNT • 2
141 074756  004737   075150             CALL    MSGPKT
142 074762                      L10026:
    074762  104423                      TRAP    C$MSG
143
144 074764                      ERR033::
145 074764  004737   075056             CALL    PNTPKT
146 074770                      L10027:
    074770  104423                      TRAP    C$MSG
147
148 074772                      ERR034::
149 074772  004737   075056             CALL    PNTPKT
150 074776                      L10030:
    074776  104423                      TRAP    C$MSG
151
152 075000                      ERR035::
153 075000  004137   075674             JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
    075004  073230                      .WORD   X35               ;ADDRESS OF ASCIZ STRING
    075006  000000                      .WORD   ARG.CT            ;ARGUMENT COUNT • 2
154 075010  004737   075150             CALL    MSGPKT
155 075014                      L10031:
    075014  104423                      TRAP    C$MSG
156
157 075016                      ERR036::
158 075016  004137   075674             JSR     R1,LPNTB          ;CALL    LPNTB PRINT ROUTINE
    075022  073310                      .WORD   X36               ;ADDRESS OF ASCIZ STRING
    075024  000000                      .WORD   ARG.CT            ;ARGUMENT COUNT • 2
159 075026                      L10032:
    075026  104423                      TRAP    C$MSG
160
161 075030                      ERR037::
```

C12

```
162 075030  010146              MOV    R1, (SP)        ;PUSH R1 ON STACK
    075032  004137  075674      JSR    R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
    075036  073426              .WORD  X37             ;ADDRESS OF ASCIZ STRING
    075040  000002              .WORD  ARG.CT          ;ARGUMENT COUNT * 2
163 075042              L10033:
    075042  104423              TRAP   C$MSG
164
165 075044              ERRO38::
166 075044  004137  075674      JSR    R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
    075050  070460              .WORD  X38             ;ADDRESS OF ASCIZ STRING
    075052  000000              .WORD  ARG.CT          ;ARGUMENT COUNT * 2
167 075054              L10034:
    075054  104423              TRAP   C$MSG
168
169 075056              PNTPKT:
    075056  004137  075674      JSR    R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
    075062  073644              .WORD  XPKT1           ;ADDRESS OF ASCIZ STRING
    075064  000000              .WORD  ARG.CT          ;ARGUMENT COUNT * 2
170 075066  010401              MOV    R4,R1
171 075070  062701  000020      ADD    #MC.CPK,R1
172 075074  010402              MOV    R4,R2
173 075076  062702  000020      ADD    #MC.MPK,R2
174 075102  012703  000014      MOV    #12.,R3
175
176 075106              PNTPKL:
    075106  011246              MOV    (R2),(SP)       ;PUSH (R2) ON STACK
    075110  016246  000002      MOV    2(R2),(SP)      ;PUSH 2(R2) ON STACK
    075114  011146              MOV    (R1),(SP)       ;PUSH (R1) ON STACK
    075116  016146  000002      MOV    2(R1),(SP)      ;PUSH 2(R1) ON STACK
    075122  004137  075674      JSR    R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
    075126  074135              .WORD  XPKT2           ;ADDRESS OF ASCIZ STRING
    075130  000010              .WORD  ARG.CT          ;ARGUMENT COUNT * 2
177 075132  062701  000004      ADD    #4,R1
178 075136  062702  000004      ADD    #4,R2
179 075142  005303              DEC    R3
180 075144  001360              BNE    PNTPKL
181 075146  000207              RETURN
182
183 075150              MSGPKT:
    075150  004137  075674      JSR    R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
    075154  073543              .WORD  XMSG1           ;ADDRESS OF ASCIZ STRING
    075156  000000              .WORD  ARG.CT          ;ARGUMENT COUNT * 2
184 075160  016504  000016      MOV    C.MCOM(R5),R4
185 075164  062704  000206      ADD    #MC.BF2,R4
186 075170  012703  000005      MOV    #5,R3
187 075174              MSGPKL:
    075174  016446  000014      MOV    12.(R4),(SP)    ;PUSH 12.(R4) ON STACK
    075200  016446  000012      MOV    10.(R4),(SP)    ;PUSH 10.(R4) ON STACK
    075204  016446  000010      MOV    8.(R4),(SP)     ;PUSH 8.(R4) ON STACK
    075210  016446  000006      MOV    6(R4),(SP)      ;PUSH 6(R4) ON STACK
    075214  016446  000004      MOV    4(R4),(SP)      ;PUSH 4(R4) ON STACK
    075220  016446  000002      MOV    2(R4),(SP)      ;PUSH 2(R4) ON STACK
    075224  011446              MOV    (R4),(SP)       ;PUSH (R4) ON STACK
    075226  004137  075674      JSR    R1,LPNTB        ;CALL    LPNTB PRINT ROUTINE
    075232  073577              .WORD  XMSG2           ;ADDRESS OF ASCIZ STRING
    075234  000016              .WORD  ARG.CT          ;ARGUMENT COUNT * 2
188 075236  062704  000016      ADD    #14.,R4
```

```
189 075242 005303                        DEC    R3
190 075244 001353                        BNE    MSGPKL
191 075246 000207                        RETURN
192
193 075250                        ERR.IN::
194 075250 013702 064444                MOV    TNUM,R2              ;GET TEST NUMBER
195 075254 006302                        ASL    R2                      ;DOUBLE
196 075256 012703 067032                MOV    @BASL3,R3               ;GET ADDRESS OF DRIVE PRINT LINE
197 075262 005764 000004                TST    4(R4)                ;CHECK IF DRIVE NUMBER GIVEN
198 075266 100002                        BPL    1$                      ;BRANCH IF SO
199 075270 012703 067047                MOV    @BAS,R3
200 075274                        1$:
    075274 016446 000004                MOV    4(R4),(SP)           ;PUSH 4(R4) ON STACK
    075300 010346                        MOV    R3,-(SP)             ;PUSH R3 ON STACK
    075302 011546                        MOV    (R5),(SP)            ;PUSH (R5) ON STACK
    075304 012746 067013                MOV    @BASL2,-(SP)         ;PUSH @BASL2 ON STACK
    075310 011446                        MOV    (R4),-(SP)           ;PUSH (R4) ON STACK
    075312 012746 066775                MOV    @BASL1,-(SP)         ;PUSH @BASL1 ON STACK
    075316 016246 076272                MOV    TNAMES-2(R2),-(SP)      ;PUSH TNAMES-2(R2) ON STACK
    075322 004137 075674                JSR    R1,LPNTB             ;CALL   LPNTB PRINT ROUTINE
    075326 067050                        .WORD  BASLN                ;ADDRESS OF ASCIZ STRING
    075330 000016                        .WORD  ARG.CT               ;ARGUMENT COUNT * 2
202 075332 004737 106362                CALL   RNTIME               ;GET RUNTIME PARAMETERS
203 075336 112700 000015                MOVB   #CR,R0               ;STORE #CR IN R0 AND
    075342 004737 075506                JSR    PC,PRINTC            ;PRINT THE CHARACTER.
204 075346 062704 000006                ADD    #6,R4                ;INCREASE R4 TO POINT TO MESSAGE POINTER
205 075352 012402                        MOV    (R4)+,R2                ;GET MESSAGE POINTER
206 075354 006302                        ASL    R2                      ;DOUBLE TO MAKE BYTE OFFSET
207 075356 063702 064432                ADD    DMPROG,R2               ;ADD TO START OF MESSAGE STRINGS
208 075362 067702 167044                ADD    @DMPROG,R2              ;ADD SIZE OF MAIN PROGRAM
209 075366 105712                        TSTB   (R2)                 ;CHECK FIRST BYTE
210 075370 001001                        BNE    NCON                 ;IF ZERO
211 075372 005202                        INC    R2                      ; INCREMENT TO NEXT BYTE
212 075374 012737 075612 075770 NCON:    MOV    #PX,PTYPE               ;CHANGE TO EXTENDED OUTPUT
213 075402 004737 075772                CALL   OSTRNG               ;OUTPUT ACCORDING TO STRING
214 075406                        L10035:
    075406 104423                        TRAP   C$MSG
215
```

E12

```
     1                              .SBTTL  GLOBAL SUBROUTINES SECTION
     2
     3                          ;MEMORY ALLOCATION ERROR
     4                          ;
     5                          ;THIS ROUTINE PRINTS A SYSTEM FATAL ERROR AND EXITS THE TEST
     6
     7  075410                  FMERR:
        075410  104454                  TRAP    C$ERSF
        075412  000004                  .WORD   4
        075414  000000                  .WORD   0
        075416  074324                  .WORD   ERR004
     8                                                          ;DO CLEAN UP TRAP
     9  075420  104444                  TRAP    C$DCLN
```

F12

```
 1                                      ;ALOCM
 2                                      ;
 3                                      ;
 4                                      ;ALLOCATE A BLOCK OF FREE MEMORY.  REPORT ERROR IF MEMORY EXHAUSTED.
 5                                      ;
 6                                      ;INPUTS:
 7                                      ;       R1 - NUMBER OF WORDS TO ALLOCATE
 8                                      ;       FFREE   FIRST FREE WORD IN MEMORY
 9                                      ;       FSIZE   SIZE OF FREE MEMORY AVAILABLE IN WORDS
10                                      ;OUTPUTS:
11                                      ;       R1    ADDRESS OF FIRST WORD OF ALLOCATED MEMORY
12                                      ;       FFREE   NEW FIRST FREE WORD IN MEMORY
13                                      ;       FSIZE   SIZE OF FREE MEMORY LEFT AFTER ALLOCATION
14                                      ;SYSTEM FATAL ERROR WILL BE REPORTED IF NOT ENOUGH MEMORY AVAILABLE
15                                      ;AND ENTIRE PROGRAM WILL BE STOPPED.
16
17 075422                              ALOCM:
   075422  013746  064412                  MOV     FFREE,-(SP)        ;;PUSH FFREE ON STACK
18 075426  160137  064414                  SUB     R1,FSIZE           ;REDUCE SIZE OF FREE MEMORY
19 075432  002766                          BLT     FMERR              ;REPORT ERROR IF NOT ENOUGH MEMORY
20 075434  060101                          ADD     R1,R1              ;CHANGE WORDS TO BYTES
21 075436  060137  064412                  ADD     R1,FFREE           ;CALCULATE NEW START OF FREE MEMORY
22 075442  012601                          MOV     (SP)+,R1           ;;POP STACK INTO R1
23 075444  000207                          RTS     PC
```

```
 1
 2                                        ;HCOMM
 3                                        ;
 4                                        ;ALLOCATES MEMORY FOR HOST COMMUNICATION AREA AND PACKET BUFFERS WITH ONE
 5                                        ;DESCRIPTOR IN EACH RING. TO BE CALLED AFTER INITIALIZING
 6                                        ;A CONTROLLER WITH SA.MSG=0 AND SA.CMD=0.
 7                                        ;
 8                                        ;INPUTS:
 9                                        ;       R5      ADDRESS OF CONTROLLER TABLE
10                                        ;OUTPUTS:
11                                        ;       CONTROLLER TABLE POINTING TO HOST COMMUNICATION AREA,
12                                        ;       RING POINTERS TO PACKETS,
13                                        ;       R4 - ADDRESS OF HOST COMMUNICATION AREA
14
15 075446 012701 000146         HCOMM:   MOV    @<HC.SIZ>/2,R1            ;GET SIZE OF AREA TO ALLOCATE
16 075452 004737 075422                  JSR    PC,ALOCM                  ;ALLOCATE THE MEMORY
17 075456 010104                         MOV    R1,R4                     ;GET ADDRESS OF HOST COMM AREA
18 075460 010465 000016                  MOV    R4,C.HCOM(R5)             ;PLACE BEGINNING ADRS OF HOST COMM AREA IN THE
19                                                                        ;CONTROLLER TABLE
20 075464 062701 000020                  ADD    @HC.MPK,R1                ;COMPUTE START OF MESSAGE PACKET
21 075470 010164 000004                  MOV    R1,HC.MSG(R4)             ;PLACE IN RING
22 075474 062701 000000                  ADD    @<HC.CPK HC.MPK>,R1       ;COMPUTE START OF COMMAND PACKET
23 075500 010164 000010                  MOV    R1,HC.CMD(R4)             ;PLACE IN RING
24 075504 000207                         RTS    PC
```

```
  1
  2                                        ;PRINTC
  3                                        ;
  4                                        ;PRINT A CHARACTER
  5                                        ;
  6                                        ;CALL WITH MACRO PRINT
  7
  8 075506  110037  075662    PRINTC  MOVB    R0,TTYOUT        ;SAVE CHARACTER FOR TTY OUTPUT
  9 075512  010146                    MOV     R1,-(SP)         ;;PUSH R1 ON STACK
 10 075514  012701  065037            MOV     #FRMTT,R1        ;PICKUP FORMATTED ASCIZ STRING STATEMENT
 11 075520  120027  000015            CMPB    R0,#CR           ;IF NOT A CARRIAGE RETURN, THEN
 12 075524  001002                    BNE     1$               ;PRINT SOME OTHER CHARACTER, ELSE
 13 075526  012701  065042            MOV     #CRLF,R1         ;PICKUP FORMATTED ASCIZ STRING STATEMENT
 14                                                            ;GO PRINT CR-LF.
 15 075532  004777  000232    1$:     JSR     PC,@PTYPE        ;PRINT THE ASCIZ STRING.
 16 075536  012601                    MOV     (SP)+,R1         ;;POP STACK INTO R1
 17 075540  000207                    RTS     PC
 18 075542                    PF:
    075542  012746  075662            MOV     #TTYOUT,-(SP)
    075546  010146                    MOV     R1,-(SP)
    075550  012746  000002            MOV     #2,(SP)
    075554  010600                    MOV     SP,R0
    075556  104417                    TRAP    C$PNTF
    075560  062706  000006            ADD     #6,SP
 19 075564  000207                    RTS     PC
 20 075566                    PB:
    075566  012746  075662            MOV     #TTYOUT,(SP)
    075572  010146                    MOV     R1,-(SP)
    075574  012746  000002            MOV     #2,-(SP)
    075600  010600                    MOV     SP,R0
    075602  104414                    TRAP    C$PNTB
    075604  062706  000006            ADD     #6,SP
 21 075610  000207                    RTS     PC
 22 075612                    PX:
    075612  012746  075662            MOV     #TTYOUT,-(SP)
    075616  010146                    MOV     R1,(SP)
    075620  012746  000002            MOV     #2,-(SP)
    075624  010600                    MOV     SP,R0
    075626  104415                    TRAP    C$PNTX
    075630  062706  000006            ADD     #6,SP
 23 075634  000207                    RTS     PC
 24 075636                    PS:
    075636  012746  075662            MOV     #TTYOUT,-(SP)
    075642  010146                    MOV     R1,-(SP)
    075644  012746  000002            MOV     #2,-(SP)
    075650  010600                    MOV     SP,R0
    075652  104416                    TRAP    C$PNTS
    075654  062706  000006            ADD     #6,SP
 25 075660  000207                    RTS     PC
 26
 27 075662  000          TTYOUT: .BYTE   0                     ;TTY OUTPUT BUFFER
 28 075663  000                  .BYTE   0                     ;TERMINATOR FOR ASCIZ STRING
 29                              .EVEN
```

I12

```
  1
  2                                      ;PRINT FORMATTED MESSAGE
  3                                      ;
  4                                      ;CALL WITH MACRO PNT, PNTF, PNTB, PNTX, OR PNTS
  5
  6 075664  012737  075542  075770  LPNTF:  MOV     @PF,PTYPE
  7 075672  000413                          BR      LPNT
  8
  9 075674  012737  075566  075770  LPNTB:  MOV     @PB,PTYPE
 10 075702  000407                          BR      LPNT
 11
 12 075704  012737  075612  075770  LPNTX:  MOV     @PX,PTYPE
 13 075712  000403                          BR      LPNT
 14
 15 075714  012737  075636  075770  LPNTS:  MOV     @PS,PTYPE
 16
 17 075722                          LPNT:
    075722  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
    075724  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
    075726  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
    075730  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 18 075732  012102                          MOV     (R1)+,R2        ;GET ADDRESS OF ASCIZ STRING
 19 075734  010604                          MOV     SP,R4           ;COMPUTE ADDRESS OF 1ST ARGUMENT AND
 20 075736  062704  000012                  ADD     @12,R4          ;SAVE IT IN R4.
 21 075742  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 22 075744  004737  075772                  JSR     PC,OSTRNG       ;PRINT THE FORMATTED MESSAGE
 23 075750  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
    075752  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
    075754  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
    075756  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
    075760  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
    075762  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 24 075764  062006                          ADD     (R0)+,SP        ;ADJUST STACK POINTER OVER ARGUMENTS
 25 075766  000110                          JMP     @R0             ;RETURN
 26
 27 075770  075542                  PTYPE:  .WORD   PF              ;PRINT TYPE
```

```
   4
   5                                        ;OSTRNG
   6                                        ;
   7                                        ;OUTPUT A MESSAGE ACCORDING TO A FORMAT STRING
   8                                        ;FORMAT OF THE ASCIZ STRING IS AS FOLLOWS:
   9                                        ;
  10                                        ;CHARACTERS ENCLOSED IN QUOTES ARE TO BE PRINTED AS THEY ARE.
  11                                        ;
  12                                        ;OTHERWISE CODE IS A SINGLE LETTER FOLLOWED BY AN OPTIONAL DECIMAL
  13                                        ;NUMBER:
  14                                        ;   On - PRINT OCTAL NUMBER. n REPRESENTS SIZE OF BINARY NUMBER PASSED
  15                                        ;           IN PARAMETER IN BITS. MAY BE IN RANGE 1 TO 32. IF n>16, TWO PARAMETER
  16                                        ;           WORDS ARE USED, OTHERWISE ONLY ONE WORD. LEADING ZEROS ARE PRINTED.
  17                                        ;           n IS ALWAYS SPECIFIED.
  18                                        ;   Dn - PRINT UNSIGNED DECIMAL NUMBER FROM n BIT PARAMETER. LEADING ZEROS
  19                                        ;           ARE NOT PRINTED. A 16 BIT NUMBER EQUAL TO ZERO WILL PRINT "0".
  20                                        ;   Hn - PRINT HEX NUMBER FROM PARAMETER OF n BITS. IF n>16 TWO PARAMETERS
  21                                        ;           ARE USED, OTHERWISE ONLY ONE PARAMETER. LEADING ZEROS ARE PRINTED.
  22                                        ;   Sn - PRINT n SPACES. n ASSUMED TO BE 1.
  23                                        ;   Nn - START NEW LINE (CR-LF SEQUENCE). n ASSUMED TO BE 1.
  24                                        ;   An - PRINT N ASCII CHARACTERS FROM PARAMETERS. n ASSUMED TO BE 1.
  25                                        ;           n/2 PARAMETER WORDS USED.
  26                                        ;   Rn - EXECUTE ROUTINE @n. n MUST BE GIVEN AND DEFINED IN HOST PROGRAM.
  27                                        ;
  28                                        ;A NULL CHARACTER MEANS END OF MESSAGE. A NULL AS FIRST CHARACTER IN STRING
  29                                        ;MUST BE IGNORED.
  30                                        ;
  31                                        ;INPUTS.
  32                                        ;        R2    ADDRESS OF START OF FORMAT STRING
  33                                        ;        R4    ADDRESS OF PARAMETERS
  34                                        ;OUTPUTS:
  35                                        ;        R2 AND R4 UPDATED TO END OF STRING AND PARAMETERS
  39
  40 075772  112201           OSTRNG: MOVB    (R2)+,R1         ;SEE IF TERMINATOR IN ASCIZ STRING.
  41 075774  001421                   BEQ     OSTRE            ;EXIT
  42 075776  012700  076304           MOV     @ERRC,R0         ;GET POINTER TO CHARACTER TABLE
  43 076002  120110           NCONS:  CMPB    R1,(R0)          ;COMPARE CHARACTER WITH TABLE ENTRY
  44 076004  001407                   BEQ     NCONF            ;BRANCH IF MATCH FOUND
  45 076006  105720                   TSTB    (R0)+            ;INCREMENT POINTER
  46 076010  001374                   BNE     NCONS            ;CONTINUE SEARCH IF NOT END OF TABLE
  47 076012  004137  075664           JSR     R1,LPNTF         ;CALL    LPNTF PRINT ROUTINE
     076016  065101                   .WORD   ERRME1           ;ADDRESS OF ASCIZ STRING
     076020  000000                   .WORD   ARG.CT           ;ARGUMENT COUNT * 2
  48 076022  000406                   BR      OSTRE
  49 076024  162700  076304   NCONF:  SUB     @ERRC,R0         ;GET INCREMENT INTO TABLE
  50 076030  006300                   ASL     R0               ;DOUBLE TO WORD COUNT
  51 076032  004770  076316           JSR     PC,@ERRD(R0)     ;DISPATCH TO PRINT ROUTINE
  52 076036  000755                   BR      OSTRNG           ;GET NEXT
  53 076040  000207           OSTRE:  RTS     PC
```

```
  1
  2                                                    ;CONTROL CHARACTER WAS A QUOTE, SO  PRINT ALL CHARACTERS TO
  3                                                    ;THE NEXT QUOTE.
  4
  5 076042  112200              CON.QU: MOVB    (R2)+,RO             ;GET CHARACTER
  6 076044  120027   000042             CMPB    RO,#'"               ;CHECK IF ENDING QUOTE
  7 076050  001403                       BEQ     CON.QX               ;IF SO, GO GET NEXT CONTROL CHARACTER
  8 076052  004737   075506             JSR     PC,PRINTC            ;PRINT THE CHARACTER.
  9 076056  000771                       BR      CON.QU               ;CONTINUE PRINTING
 10 076060  000207              CON.QX: RTS     PC
 11
 12                                                    ;CONTROL CHARACTER WAS AN 'A', SO PRINT ASCII CHARACTERS FROM
 13                                                    ;PARAMETERS.
 14
 15 076062  004737   102364     CON.A:  JSR     PC,GETCNT            ;GET COUNT OF CHARACTERS
 16 076066                       CON.A1:
    076066  112400                       MOVB    (R4)+,RO                      ;STORE (R4)+ IN RO AND
    076070  004737   075506             JSR     PC,PRINTC            ;PRINT THE CHARACTER.
 17 076074  005301                       DEC     R1                   ;COUNT THE CHARACTERS
 18 076076  001373                       BNE     CON.A1               ;PRINT UNTIL COUNT REACHES ZERO
 19 076100  032704   000001             BIT     #1,R4                ;CHECK IF R4 NOW ODD
 20 076104  001401                       BEQ     CON.A2
 21 076106  005204                       INC     R4                   ;IF SO, INCREMENT TO NEXT EVEN ADDRESS
 22 076110  000207              CON.A2: RTS     PC                   ;NOW GET NEXT CONTROL CHARACTER
 23
 24                                                    ;CONTROL CHARACTER WAS A 'D', SO PRINT A DECIMAL NUMBER.
 25
 26 076112  012701   000012     CON.D:  MOV     #10.,R1              ;LOAD RADIX
 27 076116  004737   102442             JSR     PC,PNTNUM            ;PRINT NUMBER
 28 076122  000207                       RTS     PC                   ;NOW GET NEXT CONTROL CHARACTER
 29
 30                                                    ;CONTROL CHARACTER WAS AN 'H', SO PRINT A HEX NUMBER.
 31
 32 076124  012701   000020     CON.H:  MOV     #16.,R1              ;LOAD RADIX
 33 076130  004737   102442             JSR     PC,PNTNUM            ;PRINT NUMBER
 34 076134  000207                       RTS     PC                   ;NOW GET NEXT CONTROL CHARACTER
```

```
 1
 2                                               ;CONTROL CHARACTER WAS AN 'O', SO PRINT AN OCTAL NUMBER.
 3
 4 076136  012701  000010        CON.O:  MOV    #8.,R1                ;LOAD RADIX
 5 076142  004737  102442                JSR    PC,PNTNUM             ;PRINT NUMBER
 6 076146  000207                        RTS    PC                    ;NOW GET NEXT CONTROL CHARACTER
 7
 8                                               ;CONTROL CHARACTER WAS AN 'N', SO PRINT A CARRIAGE RETURN LINE FEED.
 9
10 076150  004737  102364        CON.N:  JSR    PC,GETCNT             ;GET COUNT
11 076154                        CON.N1:
   076154  112700  000015                MOVB   #CR,RO                ;STORE #CR IN RO AND
   076160  004737  075506                JSR    PC,PRINTC             ;PRINT THE CHARACTER.
12 076164  005301                        DEC    R1                    ;COUNT THE SEQUENCES
13 076166  001372                        BNE    CON.N1
14 076170  000207                        RTS    PC                    ;NOW GET NEXT CONTROL CHARACTER
15
16                                               ;CONTROL CHARACTER WAS AN 'R', SO CALL ONE OF THE PRE PROGRAMMED
17                                               ;ROUTINE.
18
19 076172  004737  102364        CON.R:  JSR    PC,GETCNT             ;GET ROUTINE NUMBER
20 076176  020127  000011                CMP    R1,#ERR.SZ            ;CHECK IF DEFINED ROUTINE NUMBER
21 076202  101004                        BHI    CON.R1
22 076204  060101                        ADD    R1,R1                 ;DOUBLE COUNT TO GET WORD INDEX
23 076206  004771  076250                JSR    PC,@ERR.TB 2(R1)      ;CALL ROUTINE
24 076212  000207                        RTS    PC                    ;NOW GET NEXT CONTROL CHARACTER
25
26 076214                        CON.R1:
   076214  004137  075664                JSR    R1,LPNTF              ;CALL   LPNTF PRINT ROUTINE
   076220  065101                        .WORD  ERRME1                ;ADDRESS OF ASCIZ STRING
   076222  000000                        .WORD  ARG.CT                ;ARGUMENT COUNT * 2
27 076224  012601                        MOV    (SP)+,R1              ;;POP STACK INTO R1
28 076226  000207                        RTS    PC
29
30                                               ;CONTROL CHARACTER WAS AN 'S', SO PRINT SOME NUMBER OF SPACES.
31
32 076230  004737  102364        CON.S:  JSR    PC,GETCNT             ;GET COUNT
33 076234                        CON.S1:
   076234  112700  000040                MOVB   #' ,RO                ;STORE #'  IN RO AND
   076240  004737  075506                JSR    PC,PRINTC             ;PRINT THE CHARACTER.
34 076244  005301                        DEC    R1                    ;COUNT THE SPACES
35 076246  001372                        BNE    CON.S1
36 076250  000207                        RTS    PC                    ;NOW GET NEXT CONTROL CHARACTER
```

;

```
  1
  2                                 ;PRE PROGRAMMED ERROR ROUTINE DISPATCH TABLE
  3
  4 076252  101726        ERR.TB: .WORD CALR1        ;CALL ALTERNATE PRINT STRING IN DM MEMORY IMAGE
  5 076254  101754                .WORD CALR2        ;PRINT AN SDI DIAGNOSE RESPONSE
  6 076256  102052                .WORD `ALR3        ;DECIDE WHETHER TO PRINT RBN
  7 076260  102066                .WORD CALR4        ;PRINT BASIC LINE WITHOUT UDA ADDRESS
  8 076262  102142                .WORD CALR5        ;PRINT BASIC LINE WITH UDA ADDRESS
  9 076264  102220                .WORD CALR6        ;CALL ALTERNATE PRINT STRING IN PDP-11 MEMORY
 10 076266  102234                .WORD CALR7        ;PRINT  "REPLACE UDA MODULE M7161"
 11 076270  102252                .WORD CALR8        ;PRINT  " UDASA CONTAINS  XXXXXX"
 12 076272  102270                .WORD CALR9        ;REPRINT LAST NUMBER
 13
 14         000011                ERR.SZ  = <. ERR.TB>/2
 15
 16 076274                TNAMES:
 18 076274  066670                .WORD BASN1
 19 076276  066714                .WORD BASN2
 20 076300  066734                .WORD BASN3
 23 076302  066754                .WORD BASN4
 25
```

```
 1                              ;BUILD TWO TABLES
 2                              ;       FIRST CONTAINING CONTROL CHARACTERS
 3                              ;       SECOND CONTAINING ROUTINE ADDRESSES
 4
 5                              ;HERE IS FIRST TABLE
 6
16 076304                      ERRC:
   076304      042                     .BYTE    ' "
   076305      101                     .BYTE    'A
   076306      104                     .BYTE    'D
   076307      110                     .BYTE    'H
   076310      117                     .BYTE    'O
   076311      116                     .BYTE    'N
   076312      122                     .BYTE    'R
   076313      123                     .BYTE    'S
17 076314      000                     .BYTE    0                       ;FOLLOW WITH A NULL BYTE
18                                     .EVEN
19
20                              ;HERE IS SECOND TABLE
21
31 076316                      ERRD:
   076316      076042                  .WORD    CON.QU
   076320      076062                  .WORD    CCN.A
   076322      076112                  .WORD    CON.D
   076324      076124                  .WORD    COl.H
   076326      076136                  .WORD    CON.O
   076330      076150                  .WORD    CON.N
   076332      076172                  .WORD    CON.R
   076334      076230                  .WORD    CON.S
```

```
   1
   2                                        ;TINIT
   3                                        ;
   4                                        ;INITIALIZE VARIABLES FOR TEST
   5                                        ;
   6                                        ;INPUTS:
   7                                        ;       R1   TEST NUMBER
   8                                        ;OUTPUTS:
   9                                        ;       LBUFS    CLEARED (DELETES ERROR LOG)
  10                                        ;       TNUM    TEST NUMBER FROM R1
  11                                        ;       FNUM    LAST LOADED TEST IN TNUM < 4
  12                                        ;       ALL REGISTERS CLOBERED
  13
  14 076336  0101. '  064444      TINIT:  MOV     R1,TNUM              ;SAVE TEST NUMBER
  15 076342  00475 '  106260              CALL    RESET                ;RESET ALL UDA'S
  16 076346  005037   064650              CLR     LBUFS                ;CLEAR ERROR LOG BUFFER POINTER
  17 076352  013737   064416  064412      MOV     FMEM,FFREE           ;INIT FREE
  18 076360  013737   064420  064414      MOV     FMEMS,FSIZE          ;INIT FSIZE
  20 076366  022701   000004              CMP     #4,R1                ;ARE WE DOING TEST 4 ?
  21 076372  001413                       BEQ     TIEXIT               ; IF SO, EXIT
  22 076374  020137   064442              CMP     R1,FNUM              ; IF FILE ALREADY IN MEMORY?
  23 076400  001410                       BEQ     TIEXIT               ; IF SO, EXIT
  24 076402  012705   001122              MOV     @<STORAG DMFRST>,R5  ; R5->ADDRESS TO STORE    DM FIRST ADDRESS
  25 076406  012737   002122  064432      MOV     @STORAG,DMPROG       ; SAVE DMPROG ADDRESS
  26 076414  004737   105630              CALL    RDREC                ; READ IN RECORD
  27 076420  103401                       BCS     TINITE               ; IF ERROR, REPORT
  28 076422  000207              TIEXIT: RETURN
  29
  30 076424                      TINITE:
     076424  104454                       TRAP    C$ERSF
     076426  000007                       .WORD   7
     076430  000000                       .WORD   0
     076432  074350                       .WORD   ERR007
  31                                                                   ;DO CLEAN UP TRAP
  32 076434  104444                       TRAP    C$DCLN
  41
```

C13

SFG 0158

```
    3                                  ;RNT4DM
    4                                  ;
    5                                  ;LOAD AND RUN A TEST 4 IN THE CONTROLLERS  RETURN WHEN ALL
    6                                  ;DM PROGRAMS HAVE TERMINATED.
    8                                  ;INPUTS:
    9                                  ;      TSTTAB   POINTER TO FIRST CONTROLLER TABLE
   10                                  ;      R1   NUMBER OF CONTROLLERS TO TEST
   11                                  ;OUTPUTS:
   12                                  ;      DMPROG - POINTER TO START OF DM PROGRAM IN MEMORY
   13                                  ;      Z SET IF NO CONTROLLERS SUCCESSFULLY STARTED
   14                                  ;ALL REGISTERS ARE USED AND PREVIOUS CONTENTS DESTROYED.
   15
   16 076436 005037 064450     RNT4DM: CLR     URNING
   17 076442 005037 064532             CLR     TYPCNT              ; CLEAR FLAGS
   18 076446 010137 064446             MOV     R1,URUN             ; URUN = # OF UNITS
   19 076452 012737 002122 064432      MOV     @STORAG,DMPROG      ; DMPROG -> WHERE EITHER TEST 4 IS LOADED
   20 076460 013737 064446 064452      MOV     URUN,UCNT
   21 076466 013705 064430             MOV     TSTTAB,R5           ; R5  > CONTROLLER TABLE
   37 076472 004737 076500             CALL    GTT452              ; GET TEST 4 FOR U52
   38 076476 00047.                    BR      STLDDM              ; GO START LOADING DM PROGRAMS
```

C13

```
    1
    2                              ;GTT452
    3                              ;
    4                              ;GET TEST 4 FOR UDA52
    5                              ;
    6  076500  012701  000005      GTT452: MOV   #5.,R1              ; R1 = T4 FOR 52 FNUM
    7  076504  012705  001122              MOV   #<STORAG-DMFRST>,R5
    8  076510  020137  064442              CMP   R1,FNUM             ; DMPROG ALREADY IN MEMORY?
    9  076514  001405                      BEQ   1$                  ; IF SO, EXIT
   10  076516  004737  105630              CALL  RDREC               ; ELSE, READ RECORD.
   11  076522  103002                      BCC   1$
   12  076524  000137  076424              JMP   TINITE              ; BRANCH IF ERROR
   13  076530  000207              1$:     RETURN
```

E13

```
  1                                          ;RUNDM
  2                                          ;
  3                                          ;
  4                                          ;LOAD AND RUN A DM PROGRAM IN THE CONTROLLERS. RETURN WHEN ALL
  5                                          ;DM PROGRAMS HAVE TERMINATED.
  6                                          ;
  7                                          ;INPUTS:
  8                                          ;      TSTTAB    POINTER TO FIRST CONTROLLER TABLE
  9                                          ;      R1     NUMBER OF CONTROLLERS TO TEST
 10                                          ;IMPLICIT INPUTS:
 11                                          ;      DMPROG - POINTER TO START OF DM PROGRAM IN MEMORY
 12                                          ;OUTPUTS:
 13                                          ;      Z SET IF NO CONTROLLERS SUCCESSFULLY STARTED
 14                                          ;ALL REGISTERS ARE USED AND PREVIOUS CONTENTS DESTROYED.
 15
 16 076532  010137  064446        RUNDM:  MOV      R1,URUN             ;SAVE NUMBER OF UNITS TO RUN
 17 076536  005037  064450                CLR      URNING              ;CLEAR NUMBER OF UNITS RUNNING
 18
 19                                          ;LOAD DM PROGRAM INTO EACH CONTROLLER
 20
 21 076542  013737  064446  064452  STLDDM: MOV      URUN,UCNT                   ;SET COUNTER OF UNITS
 22 076550  013705  064430                MOV      TSTTAB,R5                   ;GET FIRST CONTROLLER TABLE
 23 076554                         LDDM:
 27 076554  005065  000014                CLR      C.FLG(R5)           ;CLEAR ALL FLAGS
 29 076560  116537  000002  002074        MOVB     C.UNIT(R5),L$LUN          ;SEE IF UNIT TO BE TESTED
 30 076566  005765  000002                TST      C.UNIT(R5)
 31 076572  100405                         BMI      LDNEXT              ;IF NOT, DON'T LOAD THIS UNIT
 33 076574  004737  103410                CALL     LOADDM              ;LOAD THE DM PROGRAM
 34 076600  001402                         BEQ      LDNEXT              ;IF ERROR, GO TO NEXT CONTROLLER
 35 076602  005237  064450                INC      URNING              ;IF NO ERROR, COUNT UNIT RUNNING
 36 076606  062705  000046        LDNEXT: ADD      #C.SIZE,R5                ;MOVE TO NEXT CONTROLLER TABLE
 37 076612  005337  064452                DEC      UCNT                ;CHECK IF MORE CONTROLLERS
 38 076616  001356                         BNE      LDDM                ;LOAD NEXT
 39
 40                                          ;CHECK IF ANY CONTROLLERS LOADED
 41
 42 076620  005737  064450                TST      URNING              ;ANY UNITS LOADED?
 43
 44                                          ;THE DM PROGRAMS ARE NOW IN CONTROL
 45                                          ;RESPDM MUST BE CALLED TO RESPOND TO THEIR REQUESTS
 46
 47 076624  000207                         RETURN
```

F13

```
 1                                                ;RESPDM
 2                                                ;
 3                                                ;RESPOND TO DM REQUESTS. RETURN WHEN ALL DM PROGRAMS
 4                                                ;HAVE TERMINATED.
 5
 6
 7 076626  013705  064430           RESPDM:  MOV      TSTTAB,R5              ;GET CONTROLLER TABLE ADDRESS
 8 076632  013737  064446  064452            MOV      URUN,UCNT             ;SET COUNTER OF UNITS
 9 076640  016504  000016           RESPCT:  MOV      C.HCOM(R5),R4      ;GET HOST COMM AREA ADDRESS
10 076644  032765  000002  000014            BIT      @CT.RN,C.FLG(R5)      ;CHECK IF PROGRAM RUNNING
11 076652  001446                             BEQ      RSPNXT         ;IF NOT, LOOK AT NEXT
12 076654  116537  000002  002074            MOVB     C.UNIT(R5),L$LUN      ;STORE UNIT NUMBER UNDER TEST
13 076662  032765  000010  000014            BIT      @CT.MSG,C.FLG(R5)     ;SEE IF INTERRUPT RECEIVED
14 076670  001071                             BNE      RSPIN          ;IF SO, LOOK AT PACKET
15 076672  032765  000004  000014            BIT      @CT.CMD,C.FLG(R5)     ;SEE IF COMMAND HAS BEEN SENT
16 076700  001520                             BEQ      RSPOU          ;IF NOT, SEND ONE
17
18                                                ;CHECK IF UDA STILL RUNNING
19
20 076702  011503                             MOV      (R5),R3               ;GET ADDRESS OF UDAIP
21 076704  016301  000002                     MOV      2(R3),R1              ;LOOK AT UDASA REGISTER
22 076710  001405                             BEQ      RSPTM          ;IF ZERO, UDA STILL RUNNING
23                                                                    ;REPORT UDA HAS FATAL ERROR
24 076712  104455                             TRAP     C$ERDF
   076714  000036                             .WORD    30
   076716  000000                             .WORD    0
   076720  074720                             .WORD    ERR030
25 076722  000445                             BR       RSPDRP         ;DROP CONTROLLER FROM TESTING
26
27                                                ;CHECK FOR TIMEOUT OF RESPONSE
28
29 076724                           RSPTM:
39 076724  005737  064616                     TST      KW.CSR                ;SEE IF A CLOCK ON SYSTEM
40 076730  001416                             BEQ      RSPNTO         ;DON'T TIME IF NO CLOCK
41 076732  023765  064630  000042            CMP      KW.EL+2,C.TOH(R5)     ;COMPARE TO TIMEOUT COUNTER
42 076740  101005                             BHI  RSPTMO
43 076742  001011                             BNE      RSPNTO
44 076744  023765  064626  000040            CMP      KW.EL,C.TO(R5)
45 076752  103405                             BLO  RSPNTO            ;IF TOO MUCH TIME ELAPSED SINCE LAST INTERRUPT
46 076754                           RSPTMO:
   076754  104455                             TRAP     C$ERDF
   076756  000037                             .WORD    31
   076760  000000                             .WORD    0
   076762  074734                             .WORD    ERR031
47 076764  000424                             BR       RSPDRP         ;DROP CONTROLLER FROM TESTING
48 076766                           RSPNTO:
49                                                ;>>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<<
50 076766  104422                             TRAP     C$BRK
```

G13

```
 1
 2                                          ;CHECK FOR TIME TO PRINT STATISTICAL REPORT
 3
 4 076770  005737  064616          RSPNXT: TST     KW.CSR            ;ANY CLOCK ON SYSTEM?
 5 076774  001412                          BEQ     RSPNRP            ;BYPASS IF NOT
 6 076776  023737  064630  064634          CMP     KW.EL+2,STIME+2   ;  A STATISTICAL REPORT
 7 077004  101005                          BHI RSPRPT
 8 077006  001005                          BNE     RSPNRP
 9 077010  023737  064626  064632          CMP     KW.EL,STIME
10 077016  103401                          BLO RSPNRP
11 077020                          RSPRPT:                           ;PRINT A STATISTICAL REPORT
12 077020  104424                          TRAP    C$DRPT
13
14                                  ;SWITCH TO NEXT CONTROLLER
15
16 077022  062705  000046          RSPNRP: ADD     #C.SIZE,R5            ;MOVE TO NEXT TABLE
17 077026  005337  064452                  DEC     UCNT              ;CHECK IF MORE CONTROLLERS
18 077032  001302                          BNE     RESPCT            ;LOOK AT NEXT CONTROLLER
19 077034  000674                          BR      RESPCM            ;LOOK AT FIRST CONTROLLER AGAIN
20
21                                  ;REMOVE A CONTROLLER FROM TESTING
22
23 077036  042765  000012  000014  RSPDRP: BIC     #CT.RN+CT.MSG,C.FLG(R5) ;CLEAR PROGRAM RUNNING
24 077044  005337  064450                  DEC     URNING            ;REDUCE RUNNING CONTROLLERS COUNT
25 077050  001347                          BNE     RSPNXT            ;IF ANY STILL RUNNING, LOOK AT THEM
26 077052  000207                          RETURN                    ;ELSE RETURN TO TEST SECTION
```

H13

```
 1
 2                                      ;CONTROLLER HAS RESPONDED, LOOK AT MESSAGE PACKET
 3
 4                                      ;CHECK FOR PROPER OPCODE IN END PACKET
 5
 6 077054  012700  000204       RSPIN:  MOV     #OP.END+OP.SSD,R0        ;GET SEND DATA END PACKET OPCODE
 7 077060  032765  000020  000014       BIT     #CT.REQ,C.FLG(R5)       ;LOOK IF SEND DATA OR RECEIVE DATA
 8 077066  001402               BEQ     RSPMWR
 9 077070  012700  000205               MOV     #OP.END+OP.RSD,R0       ;CHANGE TO RECEIVE DATA END PACKET OPCODE
10 077074  120064  000030       RSPMWR: CMPB    R0,HC.MPK+P.OPCD(R4)   ;COMPARE TO OPCODE IN END PACKET
11 077100  001010               BNE     RSPERR
12
13                                      ;LOOK AT STATUS CODE
14
15 077102  032764  000037  000032       BIT     #ST.MSK,HC.MPK+P.STS(R4)    ;CHECK FOR STATUS CODE ST.SUC (ZERO)
16 077110  001004               BNE     RSPERR
17
18                                      ;CHECK FOR EXPECTED REFERENCE NUMBER
19
20 077112  026564  000044  000020       CMP     C.REF(R5),HC.MPK+P.CRF(R4)  ;CHECK IF CORRECT REF NUMBER
21 077120  001405               BEQ     RSPPTW
22 077122               RSPERR:
   077122  104455               TRAP    C$ERDF
   077124  000041               .WORD   33
   077126  000000               .WORD   0
   077130  014764               .WORD   ERR033
23 077132  000741               BR      RSPDRP                 ;DROP UNIT FROM TESTING
24
25                                      ;CHECK IF RESPONSE FROM SEND OR RECEIVE DATA COMMAND
26
27 077134  032765  000020  000014 RSPPTW: BIT   #CT.REQ,C.FLG(R5)        ;CHECK IF RESPONSE FROM DM PROGRAM
28 077142  001445               RSPOU:  BEQ     RSPOUT                 ;LOOK AT REQUEST NUMBER IF SO
```

```
 1
 2                                         ;MAINTENANCE READ END PACKET RECEIVED, LOOK AT REQUEST FROM DM PROGRAM
 3
 4 077144  016401  000206          RSPPT2: MOV     HC.BF2(R4),R1           ;GET REQUEST NUMBER
 5 077150  042701  007777                  BIC     #007777,R1             ;CHECK TYPE
 6 077154  022701  060000                  CMP     #DU.SPC,R1             ;IS SPECIAL TYPE SET?
 7 077160  001010                          BNE     1$                     ;IF NOT, ERROR
 8 077162  042764  170000  000206          BIC     #^C007777,HC.BF2(R4)   ;CLEAR TYPE
 9 077170  016401  000206                  MOV     HC.BF2(R4),R1          ;GET REQUEST NUMBER
10 077174  020127  000017                  CMP     R1,#DSPSIZ             ;CHECK IF IN EXPECTED RANGE
11 077200  103405                          BLO RSPPT3
12 077202                          1$:
   077202  104455                          TRAP    C$ERDF
   077204  000040                          .WORD   32
   077206  000000                          .WORD   0
   077210  074746                          .WORD   ERR032
13 077212  000711                          BR      RSPDRP                 ;DROP UNIT FROM TESTING
14
15 077214  012700  000004          RSPPT3: MOV     #OP.SSD,R0             ;BUILD A SEND DATA COMMAND PACKET
16 077220  004737  104124                  CALL    BLDCMD                 ;  FOR ANSWER TO DM PROGRAM
17 077224  012700  000100                  MOV     #HC.BF1,R0             ;POINT TO BUFFER IN PACKET
18 077230  004737  104256                  CALL    CLRBUF                 ;  AND CLEAR BUFFER
19 077234  010403                          MOV     R4,R3                  ;R3 POINTS TO COMMAND BUFFER
20 077236  062704  000106                  ADD     #HC.BSZ,R4             ;R4 POINTS TO MESSAGE BUFFER
21 077242  011401                          MOV     (R4),R1                ;GET REQUEST NUMBER
22 077244  012423                          MOV     (R4)+,(R3)+            ;PUT REQUEST NUMBER INTO COMMAND PACKET
23 077246  060101                          ADD     R1,R1                  ;DOUBLE REQUEST NUMBER
24 077250  004771  077360                  CALL    @RSPDSP(R1)            ;CALL REQUESTED ROUTINE
25 077254  001270                          BNE     RSPDRP                 ;ROUTINE RETURNS Z CLEAR TO DROP UNIT FROM TESTING
26                                                                        ; Z SET IF COMMAND READY TO SEND TO UNIT
27
28                                         ;SEND COMMAND BACK TO UDA
29
30 077256  042765  000010  000014  RSPOUT: BIC     #CT.MSG,C.FLG(R5)      ;CLEAR MESSAGE RECEIVED FLAG
31 077264  032765  000020  000014          BIT     #CT.REQ,C.FLG(R5)      ;CHECK WHICH COMMAND TO SEND
32 077272  001014                          BNE     RSPOU2                 ;BRANCH IF RESPONSE TO REQUEST
33
34 077274  012700  000005                  MOV     #OP.RSD,R0             ;BUILD RECEIVE DATA COMMAND
35 077300  004737  104124                  CALL    BLDCMD
36 077304  012700  000206                  MOV     #HC.BF2,R0             ;POINT TO MESSAGE BUFFER
37 077310  004737  104256                  CALL    CLRBUF                 ;  AND CLEAR IT
38 077314  052765  000020  000014          BIS     #CT.REQ,C.FLG(R5)      ;SET REQUEST BIT
39 077322  000403                          BR      RSPOU3
40
41 077324  042765  000020  000014  RSPOU2: BIC     #CT.REQ,C.FLG(R5)      ;CLEAR REQUEST BIT
42 077332                          RSPOU3:
43 077332  004737  104210                  CALL    SNDCMD                 ;SEND COMMAND TO UDA
44 077336  012700  000264                  MOV     #3.*60.,R0             ;SET TIMEOUT FOR 3 MINUTES
45 077342  010501                          MOV     R5,R1
46 077344  062701  000040                  ADD     #C.TO,R1               ;PUT TIME IN CONTROLLER TABLE
47 077350  004737  104530                  CALL    SETTO
48 077354  000137  076770                  JMP     RSPNXT                 ;NOW WAIT FOR END PACKET
```

```
    1
    2                                    ;RESPONSE REQUEST DISPATCH TABLE
    3
    4 077360  077416            RSPDSP: .WORD T1MSIZ         ; 0. SET UP FREE MEMORY FOR ADDRESS TESTING
    5 077362  077536                    .WORD T2DLL          ; 1. PROVIDE DIAGNOSTIC PROGRAM FOR DISK DRIVE
    6 077364  077702                    .WORD T2CMD          ; 2. GET MANUAL INTERVENTION COMMAND
    7 077366  100352                    .WORD T4MPRM         ; 3. TELL DATA PATTERN 16.
    8 077370  100374                    .WORD T4UPRM         ; 4. TELL UNIT PARAMETERS, CLEAR CONTENTS
    9 077372  100654                    .WORD T4BB1          ; 5. TELL BAD BLOCKS (FIRST 14)
   10 077374  100704                    .WORD T4BB2          ; 6. TELL BAD BLOCKS (LAST TWO)
   11 077376  100734                    .WORD T4SOFT         ; 7. ADD TO SOFT ERROR AND ECC COUNTS
   12 077400  100762                    .WORD T4SEEK         ; 8. ADD 1000 TO SEEK COUNT
   13 077402  101002                    .WORD T4MXFR         ; 9. ADD TO MEGABITS READ AND WRITE COUNTS
   14 077404  101144                    .WORD UTOTST         ;10. TELL WHICH DRIVES TO TEST
   15 077406  101250                    .WORD ERRMES         ;11. REPORT ERROR MESSAGE
   16 077410  101470                    .WORD ERRMC          ;12. REPORT ERROR MESSAGE AND COUNT HARD ERROR
   17 077412  101610                    .WORD MESSAG         ;13. PRINT A DESCRIPTIVE MESSAGE
   18 077414  101722                    .WORD DONE           ;14. MARK DM PROGRAM AS NO LONGER RUNNING
   19
   20         000017            DSPSIZ = <. RSPDSP>/2        ;LEGAL NUMBERS ARE LOWER THAN THIS
```

```
 1
 2                         ;NORMAL MAINTENANCE READ BUFFER DESCRIPTION
 3
 4                         ;BYTE OFFSET FROM
 5                         ;START OF BUFFER              +- - - - -------------------+
 6                         ;           0                 !       REQUEST NUMBER       !   USED TO SELECT ROUTINE
 7                         ;                             +- - ---- - ----- ----------+
 8                         ;           2                 !      DATA ARGUMENT #1      !   R4 CONTAINS THIS ADDRESS
 9                         ;                             +- --- ----------------------+
10                         ;           4                 !      DATA ARGUMENT #2      !
11                         ;                             +- - - - - - - - ----------+
12                         ;           6                 !      DATA ARGUMENT #3      !
13                         ;                             +- -- -- - - - -------------+
14                         ;           8                 !      DATA ARGUMENT #4      !
15                         ;                             +- - --------- --------------+
16                         ;          10                 !      DATA ARGUMENT #5      !
17                         ;                             +- -- - -- ---- -----+
18                         ;          12                 !      DATA ARGUMENT #6      !
19                         ;                             +- - - -- --- -------------+
20                         ;          14                 !      DATA ARGUMENT #7      !
21                         ;                             +- -- -- ---- --------------+
22                         ;          16                 !      DATA ARGUMENT #8      !
23                         ;                             +- - - --------------------+
24                         ;          18                 !      DATA ARGUMENT #9      !
25                         ;                             +- -- - - -- ---------------+
26                         ;          20                 !      DATA ARGUMENT #10     !
27                         ;                             +---- - --- ----------------+
28                         ;          22                 !      DATA ARGUMENT #11     !
29                         ;                             +- -  - - -  - ---------------+
30
31                         ;             .                       .
32
33                         ;             .                       .
34
35                         ;             .                       .
36
37                         ;                             +- -  - - ---- - ---------+
38                         ;          68                 !      DATA ARGUMENT #34     !
39                         ;                             +- -  -      -   ---------+
```

```
 1
 2                              ;NORMAL PSEUDO TERMINAL IN PACKET DESCRIPTION GIVEN IN RESPONSE TO ABOVE PACKET
 3
 4                              ;BYTE OFFSET FROM
 5                              ;START OF PACKET         +  -   -------------- -- --+
 6                              ;       0                !        REQUEST NUMBER         !   ECHOED FROM REQUEST PACKET
 7                              ;                        +- --   ------- -- ------ +
 8                              ;       2                !      DATA ARGUMENT #1         !   R3 CONTAINS THIS ADDRESS
 9                              ;                        +------  --  -  ----- --  -  +
10                              ;       4                !      DATA ARGUMENT #2         !   ALL DATA ARGUMENTS ARE RETURNED
11                              ;                        +- -    ---------- -   ---+    CONTAINING ZEROS UNLESS
12                              ;       6                !      DATA ARGUMENT #3         !   SPECIFICALLY INDICATED BY
13                              ;                        +  -- --  --  -- - ------- +   RESPONSE ROUTINE.
14                              ;       8                !      DATA ARGUMENT #4         !
15                              ;                        +----- - -  - - ----- -  -- -+
16                              ;      10                !      DATA ARGUMENT #5         !
17                              ;                        +  - -- ----- ----   - --+
18                              ;      12                !      DATA ARGU'  NT #6        !
19                              ;                        +------  ----   ------- - +
20                              ;      14                !      DATA ARGUMENT #7         !
21                              ;                        +-- -    -- --- ------- --- - - +
22                              ;      16                !      DATA ARGUMENT #8         !
23                              ;                        +- ------- -- -----------------+
24                              ;      18                !      DATA ARGUMENT #9         !
25                              ;                        + --- -- -----------------  -- ---+
26                              ;      20                !      DATA ARGUMENT #10        !
27                              ;                        + --.- -- --- ----------------+
28                              ;      22                !      DATA ARGUMENT #11        !
29                              ;                        +- -   ------ ------------- --+
30                              ;
31                              ;         .                        .
32                              ;
33                              ;         .                        .
34                              ;
35                              ;         .                        .
36                              ;                        +     - -- ---------------- +
37                              ;
38                              ;      68                !      DATA ARGUMENT #34        !
39                              ;                        +  --- -- ------  -- ---+
```

```
 1                                      ;T1MSIZ  -  DM REQUEST 0
 2                                      ;
 3                                      ;
 4                                      ;SET UP MEMORY FOR ADDRESS TESTING FROM UDA.
 5                                      ;PLACE ADDRESS OF EACH LOCATION INTO EACH LOCATION IN FREE
 6                                      ;MEMORY. RETURN FIRST LOCATION OF FREE MEMORY IN CMD.02 (LOW BITS)
 7                                      ;AND CMD.03 (HIGH BITS). RETURN LAST LOCATION OF FREE MEMORY IN
 8                                      ;CMD.04 AND CMD.05. ALSO RETURN FIRST EXISTANT LOCATION IN CMD.06
 9                                      ;AND CMD.07; LAST EXISTANT LOCATION IN CMD.08 AND CMD.09.
10                                      ;
11                                      ;INPUTS:
12                                      ;       R5 - CONTROLLER TABLE ADDRESS
13                                      ;       R4   MESSAGE PACKET DATA ADDRESS (POINTING TO MSG.02)
14                                      ;       R3 - COMMAND PACKET DATA ADDRESS (POINTING TO CMD.02)
15                                      ;OUTPUTS:
16                                      ;       COMMAND PACKET CONTAINING:
17                                      ;        (R3) LOW ADDRESS BITS OF FIRST WRITABLE ADDRESS
18                                      ;        2.(R3) HIGH ADDRESS BITS OF FIRST WRITABLE ADDRESS
19                                      ;        4.(R3) LOW ADDRESS BITS OF LAST WRITABLE ADDRESS
20                                      ;        6.(R3) HIGH ADDRESS BITS OF LAST WRITABLE ADDRESS
21                                      ;        8.(R3) LOW ADDRESS BITS OF FIRST READABLE ADDRESS
22                                      ;        10.(R3) HIGH ADDRESS BITS OF FIRST READABLE ADDRESS
23                                      ;        12.(R3) LOW ADDRESS BITS OF LAST READABLE ADDRESS
24                                      ;        14.(R3) HIGH ADDRESS BITS OF LAST READABLE ADDRESS
25                                      ;        Z SET
26
27 077416                      T1MSIZ:
29 077416  013701  064412              MOV     FFREE,R1            ;GET FIRST ADDRESS OF FREE MEMORY
30 077422  013702  064414              MOV     FSIZE,R2            ;GET SIZE
31
32                                      ;FILL MEMORY WITH ADDRESS PATTERN
33
34 077426  010111              MEMFIL: MOV     R1,(R1)             ;WRITE DATA INTO LOCATION
35 077430  062701  000002              ADD     #2,R1               ;INCREASE ADDRESS TO NEXT LOCATION
36 077434  005302                      DEC     R2                  ;COUNT THE WORDS
37 077436  001373                      BNE     MEMFIL              ;FILL ALL WORDS
38
39                                      ;SEND LOCATION OF FREE MEMORY TO UDA
40
41 077440  013723  064412              MOV     FFREE,(R3)+         ;LOAD FIRST ADDRESS OF FREE MEMORY
42 077444  005023                      CLR     (R3)+               ; HIGH ORDER BITS ARE ZERO
43 077446  013700  064414              MOV     FSIZE,R0            ;GET SIZE OF FREE MEMORY
44 077452  006300                      ASL     R0                  ;CONVERT TO BYTES
45 077454  063700  064412              ADD     FFREE,R0            ;COMPUTE LAST LOCATION
46 077460  162700  000002              SUB     #2,R0
47 077464  010023                      MOV     R0,(R3)+            ;LOAD LAST LOCATION
48 077466  005023                      CLR     (R3)+               ;CLEAR HIGH ORDER BITS
49
50                                      ;SEND LOCATION OF READABLE MEMORY
51
52 077470  005023                      CLR     (R3)+               ;SEND ZERO AS START OF READABLE MEMORY
53 077472  005023                      CLR     (R3)+
54 077474  013700  002120              MOV     L$HIMEM,R0          ;GET HIGH MEMORY ADDRESS
55 077500  005001                      CLR     R1                  ;CLEAR HIGH BITS
56 077502  006300                      ASL     R0                  ;SHIFT LEFT 6 PLACES
57 077504  006300                      ASL     R0
58 077506  006300                      ASL     R0
```

N⊥3

```
59 077510  006300              ASL     RO
60 077512  006300              ASL     RO
61 077514  006101              ROL     R1
62 077516  006300              ASL     RO
63 077520  006101              ROL     R1
64 077522  052700  000076      BIS     #76,RO        ;SET LOW ORDER BITS
65 077526  010023              MOV     RO,(R3)+      ;PUT INTO BUFFER
66 077530  010123              MOV     R1,(R3)+
68 077532  000264              SEZ
69 077534  000207              RETURN
```

B14

```
 1                                         ;T2DLL    DM REQUEST 1
 2                                         ;
 3                                         ;
 4                                         ;PROVIDE DIAGNOSTIC TO DOWNLINE LOAD INTO DISK DRIVE.
 5                                         ;
 6                                         ;THE UDA MAY BE USED TO GET THE DIAGNOSTIC IF THE SYSTEM LOAD DEVICE
 7                                         ;IS ON THE UDA. THIS ACTION WILL CAUSE A REINITIALIZATION OF THE  JOA
 8                                         ;AND THE RING STRUCTURE MOVED. SINCE THIS PROGRAM HAS NO WAY TO
 9                                         ;DETERMINE IF THE UDA IS USED. IT WILL ALWAYS ASSUME IT IS USED AND
10                                         ;WILL INITIALIZE AND RELOAD THE DM PROGRAM AFTER READING THE
11                                         ;DIAGNOSTIC. THE OUTPUTS OF THIS ROUTINE ARE STORED AND SENT TO THE
12                                         ;DM PROGRAM IN THE UTOTST REQUEST.
13                                         ;
14                                         ;INPUTS:
15                                         ;        R5   CONTROLLER TABLE ADDRESS
16                                         ;        R4   MESSAGE DATA ADDRESS
17                                         ;           (R4) DRIVE NUMBER
18                                         ;          2.(R4) A VALUE THE DM PROGRAM WISHES RETURNED
19                                         ;          4.(R4) REGION TO WHICH PROGRAM IS TO BE LOADED IN DISK
20                                         ;          6.(R4) 2 WORD PROGRAM NAME IN RAD50
21                                         ;        R3 - COMMAND DATA ADDRESS
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
38 077536                                  T2DLL:
40 077536  005037  064656                          CLR     DLL                     ;CLEAR CONTROL WORD
41 077542  012437  064660                          MOV     (R4)+,DLLDR             ;DRIVE NUMBER
42 077546  012437  064662                          MOV     (R4)+,DLLV              ;VALUE FROM DM
43 077552  012437  064664                          MOV     (R4)+,DLLR              ;REGION
44 077556  012437  064674                          MOV     (R4)+,DLLNAM            ;PROGRAM NAME
45 077562  012437  064676                          MOV     (R4)+,DLLNAM+2          ;   (TWO WORDS)
46
47                                                  ;RESET UDA AND READ DM PROGRAM
48
49 077566  005075  000000                          CLR     @(R5)                   ;RESET THE UDA
50 077572  013737  064412  064666                  MOV     FFREE,DLLADR            ;GET ADDRESS WHERE PROGRAM
51 077600  005037  064670                          CLR     DLLADR+2                ;  TO BE STORED
52 077604  013737  064414  064672                  MOV     FSIZE,DLLSIZ            ;SAVE CURRENT SIZE OF MEMORY
53 077612  004737  105570                          CALL    RDDLL                   ;READ DLL PROGRAM FROM DATA FILE
54 077616  103002                                  BCC     1$                      ;PROGRAM NOT FOUND IF CARRY SET
55 077620  005237  064656                          INC     DLL                     ;RETURN 1 IF PROGRAM FOUND
56 077624  005237  064656                  1$:     INC     DLL                     ;RETURN 2 IF PROGRAM NOT FOUND
57 077630  013737  064672  064414                  MOV     DLLSIZ,FSIZE            ;COMPUTE SIZE OF DLL PROGRAM
58 077636  013737  064412  064672                  MOV     FFREE,DLLSIZ            ;  AND RESTORE ORIGINAL FFREE
```

C14

```
59 077644  163737  064666  064672        SUB     DLLADR,DLLSIZ        ;  AND FSIZE VALUES
60 077652  013737  064666  064412        MOV     DLLADR,FFREE
61 077660  005726                         TST     (SP)                ;POP RETURN ADDRESS OFF STACK
62 077662  012701  000001                MOV     #1,P1               ;RUN THE DM PROGRAM AGAIN
63 077666  004737  076532                CALL    RJNDM
64 077672  001402                         BEQ     2$
65 077674  000137  076626                JMP     RFSPDM
66 077700  000207                 2$:     RETURN
```

```
     1                                          ;T2CMD   DM REQUEST 2
     2                                          ;
     3                                          ;GET MANUAL INTERVENTION COMMAND
     4                                          ;
     5                                          ;
     6                                          ;INPUTS:
     7                                          ;       R5 - CONTROLLER TABLE ADDRESS
     8                                          ;       R4 - MESSAGE DATA ADDRESS
     9                                          ;         (R4) DRIVE NUMBER
    10                                          ;       2.(R4) OPERATION CODE
    11                                          ;           0 ON FIRST REQUEST FOR DRIVE. ECHO OF PREVIOUS RESPONSE ALL OTHER TIMES.
    12                                          ;           IF OPERATION CODE = 2
    13                                          ;       4.(R4) DATA BYTE READ (TO BE PRINTED)
    14                                          ;       R3 - COMMAND DATA ADDRESS
    15                                          ;OUTPUTS:
    16                                          ;       COMMAND DATA FILLED WITH THE FOLLOWING:
    17                                          ;         (R3) OPERATION CODE
    18                                          ;               0 - EXIT
    19                                          ;               1 - WRITE
    20                                          ;               2 - READ
    21                                          ;               3 - DIAGNOSE
    22                                          ;           IF OPERATION CODE = 1, 2 OR 3
    23                                          ;       2.(R3) REGION NUMBER
    24                                          ;       4.(R3) OFFSET INTO REGION
    25                                          ;           IF OPERATION CODE = 1
    26                                          ;       6.(R3) DATA BYTE
    27                                          ;       Z SET IF DATA RETURNED
    28                                          ;       Z CLEAR IF DRIVE NUMBER NOT ON THIS CONTROLLER
    29 077702                                   T2CMD:
    31 077702   032737  000200  064400          BIT      @SM.MAN.SFPTBL+SO.BIT    ;LOOK AT MANUAL INTERVENTION MODE
    32 077710   001002                          BNE      T2CMDM                   ;EXIT IF NOT WANTED
    33 077712   000137  100334                  JMP      T2CMDX
    34 077716                                   T2CMDM:
       077716   104450                          TRAP     C$MANI
    35 077720   103406                          BCS      T2CMD0
    36 077722                                   T2CMD9:
       077722   004137  075664                  JSR      R1,LPNTF                 ;CALL    LPNTF PRINT ROUTINE
       077726   065757                          .WORD    T2WARN                   ;ADDRESS OF ASCIZ STRING
       077730   000000                          .WORD    ARG.CT                   ;ARGUMENT COUNT = 2
    37 077732   000137  100334                  JMP      T2CMDX
    38 077736   012401                 T2CMD0:  MOV      (R4)+,R1                 ;GET DRIVE NUMBER
    39 077740   012402                          MOV      (R4)+,R2                 ;GET OPERATION CODE
    40 077742   001022                          BNE      T2CMD2                   ;BRANCH IF NOT ZERO
    41 077744   004737  102274                  CALL     GTDRVT                   ;GET DRIVE TABLE ADDRESS
    42 077750   001401                          BEQ      1$                       ;CHECK IF DRIVE FOUND
    43 077752   000207                          RETURN                           ;RETURN WITH Z CLEAR IF NOT
    44
    45 077754                          1$:
       077754   011446                          MOV      (R4),(SP)                ;PUSH (R4) ON STACK
       077756   011546                          MOV      (R5),(SP)                ;PUSH (R5) ON STACK
       077760   016446  000002                  MOV      D.UNIT(R4),(SP)          ;PUSH D.UNIT(R4) ON STACK
       077764   004137  075664                  JSR      R1,LPNTF                 ;CALL    LPNTF PRINT ROUTINE
       077770   066056                          .WORD    T2CMS1                   ;ADDRESS OF ASCIZ STRING
       077772   000006                          .WORD    ARG.CT                   ;ARGUMENT COUNT = 2
    46 077774   005037  064642                  CLR      T2WRR                    ;CLEAR ALL STORAGE WORDS
    47 100000   005037  064644                  CLR      T2WRO
    48 100004   005037  064646                  CLR      T2DR
```

F 14

```
 49
 50 100010  022702  000002          T2CMD2:  CMP     #2,R2            ;SEE IF LAST OPERATION WAS READ
 51 100014  001027                           BNE     T2CMDQ           ;BRANCH IF NOT TO QUESTION
 52 100016  112700  000040                   MOVB    #' ,R0           ;STORE #'   IN R0 AND
    100022  004737  075506                   JSR     PC,PRINTC        ;PRINT THE CHARACTER.
 53 100026  013701  064642                   MOV     T2WRR,R1         ;PRINT REGION
 54 100032  004737  103010                   CALL    T2PNTW
 55 100036  013701  064644                   MOV     T2WRO,R1         ;PRINT OFFSET
 56 100042  004737  103010                   CALL    T2PNTW
 57 100046  112700  000057                   MOVB    #'/,R0           ;STORE #'/ IN R0 AND
    100052  004737  075506                   JSR     PC,PRINTC        ;PRINT THE CHARACTER.
 58 100056  012401                           MOV     (R4)+,R1         ;PRINT THE DATA
 59 100060  004737  103040                   CALL    T2PNTB
 60 100064  112700  000015                   MOVB    #CR,R0           ;STORE #CR IN R0 AND
    100070  004737  075506                   JSR     PC,PRINTC        ;PRINT THE CHARACTER.
 61
 62                                                  ;NOW ASK FOR COMMAND INPUT
 63
 64 100074                          T2CMDQ:
    100074  104443                           TRAP    C$GMAN
    100076  000406                           BR      10000$
    100100  064476                           .WORD   TEMP
    100102  000142                           .WORD   T$CODE
    100104  064760                           .WORD   T4OPT7
    100106  177777                           .WORD   -1
    100110  000001                           .WORD   T$LOLIM
    100112  000024                           .WORD   T$HILIM
    100114                          10000$:
 65 100114  012701  064476                   MOV     @TEMP,R1         ;GET POINTER TO STRING
 66 100120  112100                           MOVB    (R1)+,R0         ;GET COMMAND CHARACTER
 67 100122  022700  000105                   CMP     #'E,R0
 68 100126  001415                           BEQ     T2CMDV
 69 100130  022700  000104                   CMP     #'D,R0
 70 100134  001016                           BNE     T2CMD3
 71 100136  012713  000003                   MOV     #3,(R3)          ;STORE DIAGNOSE OPERATION CODE
 72 100142  004737  103122                   CALL    T2GNUM           ;GET REGION FROM COMMAND
 73 100146  001402                           BEQ     1$
 74 100150  010437  064646                   MOV     R4,T2DR
 75 100154  013763  064646  000002  1$:      MOV     T2DR,2(R3)
 76 100162  004737  103122          T2CMDV:  CALL    T2GNUM           ;MAKE SURE AT END OF LINE
 77 100166  001064                           BNE     T2CMDE
 78 100170  000461                           BR      T2CMDX
 79
 80                                                  ;COMMAND MUST BE EITHER READ OR WRITE
 81
 82 100172  012713  000002          T2CMD3:  MOV     #2,(R3)          ;CHECK IF READ
 83 100176  022700  000122                   CMP     #'R,R0
 84 100202  001415                           BEQ     T2CMDR
 85 100204  022700  000127                   CMP     # W,R0           ;CHECK IF WRITE
 86 100210  001053                           BNE     T2CMDE           ; IF NOT   ERROR
 87 100212  012713  000001                   MOV     #1,(R3)
 88 100216  004737  103122                   CALL    T2GNUM           ;GET DATA BYTE
 89 100222  001446                           BEQ     T2CMDE           ;ERROR IF NO DATA
 90 100224  162700  000002                   SUB     #2,R0
 91 100230  003043                           BGT     T2CMDE           ;OR GREATER THAN TWO DIGITS
 92 100232  010463  000006                   MOV     R4,6(R3)         ;STORE DATA BYTES IN BUFFER
 93 100236  013763  064642  000002  T2CMDR:  MOV     T2WRR,2(R3)      ;PUT REGION AND OFFSET
```

F 14

```
 94 100244   013763   064644   000004           MOV      T2WRO,4(R3)      ;  INTO BUFFER
 95 100252   021302                              CMP      (R3),R2          ;IF SO,
 96 100254   001002                              BNE      T2CMDN
 97 100256   005263   000004                     INC      4(R3)            ;  INCREMENT OFFSET
 98 100262   004737   1031_2            T2CMDN:  CALL     T2GNUM
 99 100266   001411                              BEQ      T2CMDW
100 100270   010463   00C002                     MOV      R4,2(R3)
101 100274   005063   000004                     CLR      4(R3)
102 100300   004737   103122                     CALL     T2GNUM
103 100304   001402                              BEQ      T2CMDW
104 100306   010463   000004                     MOV      R4,4(R3)
105 100312   004737   103122            T2CMDW:  CALL     T2GNUM
106 100316   001010                              BNE      T2CMDE
107 100320   016337   000002   064642            MOV      2(R3),T2WRR      ;SAVE REGION
108 100326   016337   000004   064644            MOV      4(R3),T2WRO      ;SAVE OFFSET
109 100334   000264                     T2CMDX:  SEZ
110 100336   000207                              RETURN
111 100340                              T2CMDE:
    100340   004137   075664                     JSR      R1,LPNTF         ;CALL   LPNTF PRINT ROUTINE
    100344   066445                              .WORD    T2CMS5           ;ADDRESS OF ASCIZ STRING
    100346   000000                              .WORD    ARG.CT           ;ARGUMENT COUNT * 2
112 100350   000651                              BR       T2CMDQ           ;GO ASK AGAIN
```

G14

```
 1                                    ;T4MPRM   DM REQUEST 3
 2                                    ;
 3                                    ;REQUEST FOR TEST 4 CONTENTS OF DATA PATTERN 16.
 4                                    ;
 5                                    ;INPUTS:
 6                                    ;      R5 - CONTROLLER TABLE ADDRESS
 7                                    ;      R4 - MESSAGE DATA ADDRESS
 8                                    ;            (NO DATA)
 9                                    ;      R3 - COMMAND DATA ADDRESS
10                                    ;OUTPUTS:
11                                    ;      COMMAND DATA FILLED WITH THE FOLLOWING:
12                                    ;        (R3) NUMBER OF WORDS IN DATA PATTERN 16
13                                    ;       2.(R3) DATA IN PATTERN 16
14                                    ;        )              "
15                                    ;      32.(R3)          "
16                                    ;      Z SET
17
18
19 100352  012701  000021   T4MPRM:  MOV      #17.,R1            ;GET COUNT
20 100356  012702  064554            MOV      #PAT16C,R2         ; AND ADDRESS OF PATTERN 16 PARAMETERS
21 100362  012223            1$:     MOV      (R2)+,(R3)+        ;COPY THE DATA TO BUFFER
22 100364  005301            DEC      R1
23 100366  001375            BNE      1$
24 100370  000264            SEZ                                 ;RETURN WITH Z SET
25 100372  000207            RETURN
```

H14

```
 1
 2                                        ;T4UPRM - DM REQUEST 4
 3                                        ;
 4                                        ;REQUEST FOR TEST 4 UNIT PARAMETERS
 5                                        ;
 6                                        ;INPUTS:
 7                                        ;        R5 - CONTROLLER TABLE ADDRESS
 8                                        ;        R4 - MESSAGE DATA ADDRESS
 9                                        ;          (R4) DRIVE NUMBER
10                                        ;         2.(R4) DRIVE SERIAL NUMBER
11                                        ;            )
12                                        ;         6.(R4)
13                                        ;         8.(R4) HDA SERIAL NUMBER
14                                        ;            )
15                                        ;        14.(R4)
16                                        ;        R3 - COMMAND DATA ADDRESS
17                                        ;OUTPUTS:
18                                        ;        COMMAND DATA FILLED WITH THE FOLLOWING:
19                                        ;          (R3) PARAMETER BITS (1 FOR TRUE)
20                                        ;                BIT      14 - INITIAL WRITE
21                                        ;                BIT      13 - DIAGNOSTIC CYLINDERS
22                                        ;                BIT      12 - ECC CORRECTION
23                                        ;                BIT      11 - READ ONLY
24                                        ;                BIT      10 - WRITE ONLY
25                                        ;                BIT       9 - RETRIES
26                                        ;                BIT       8 - TRACK/GROUP AND CYLINDERS SPECIFIED
27                                        ;                BIT       7 - (NOT USED)
28                                        ;                BIT       6 - SEQUENTIAL SEEKS
29                                        ;                BIT       5 - BEGIN/END SETS SPECIFIED
30                                        ;                BIT       4 - TRACK SPECIFIED (0   GROUPS SPECIFIED)
31                                        ;                     HAS MEANING ONLY WHEN BIT 5 IS ZERO
32                                        ;                BIT       3   WRITE CHECKS ENABLED
33                                        ;                BIT       2 - WRITE CHECKS ALWAYS
34                                        ;                BIT       1 - DATA COMPARES ENABLED
35                                        ;                BIT       0 - DATA COMPARE ALWAYS
36                                        ;         2.(R3) DATA PATTERN NUMBER
37                                        ;        IF PARAMETER BIT 5 SET
38                                        ;         4.(R3) COUNT OF BEGIN/END SETS
39                                        ;         6.(R3) BEGIN BLOCK (2 WORDS) THEN END BLOCK (2 WORDS)
40                                        ;            )      1 TO 4 SETS
41                                        ;            )         OR
42                                        ;            )      IF COUNT OF BEGIN/END BLOCKS = 0
43                                        ;        36.(R3)   START CYLINDER (2 WORDS) THEN END CYLINDER (2 WORDS)
44                                        ;                  END CYLINDER A NEGATIVE VALUE IF TO TEST ENTIRE AREA
45                                        ;        IF PARAMETER BIT 5 CLEAR
46                                        ;         4.(R3) STARTING CYLINDER
47                                        ;         6.(R3)    (2 WORDS)
48                                        ;         8.(R3) ENDING CYLINDER (2 WORDS)
49                                        ;        10.(R3)    NEGATIVE FOR ALL CYLINDERS
50                                        ;        12.(R3) NUMBER OF TRACKS OR GROUPS SPECIFIED
51                                        ;        14.(R3) 1 TO 7 TRACK OR GROUP NUMBERS
52                                        ;            )       DETERMINED BY PARAMETER BIT 4
53                                        ;        26.(R3)
54                                        ;        Z SET IF DATA RETURNED
55                                        ;        Z CLEAR IF UNIT NUMBER NOT ON THIS CONTROLLER
```

```
    1
    2 100374  012401                   T4UPRM: MOV     (R4)+,R1            ;GET DRIVE NUMBER
    3 100376  010402                           MOV     R4,R2               ;SAVE DATA ADDRESS
    4 100400  004737  102274                   CALL    GTDRVT              ;GET DRIVE TABLE ADDRESS
    5 100404  001122                           BNE     T4UPRX              ;CHECK IF DRIVE FOUND
    6 100406  012264  000200                   MOV     (R2)+,D.SERN(R4)    ;COPY DRIVE SERIAL NUMBER TO DRIVE TABLE
    7 100412  012264  000202                   MOV     (R2)+,D.SERN+2(R4)
    8 100416  012264  000204                   MOV     (R2)+,D.SERN+4(R4)
   14 100422  016401  000004                   MOV     D.PRM(R4),R1        ;GET PARAMETER BITS
   15 100426  042701  140200                   BIC     #D.ZERO,R1          ;CLEAR SOME BITS
   16 100432  032737  000020  064440           BIT     #ISTRTH,IFLAGS      ;FIRST TIME TEST 4 B  NG RUN.
   17 100440  001406                           BEQ     1$                  ;BRANCH IF NOT, ELSE
   18 100442  032737  040000  064400           BIT     #SM.IW,SFPTBL+SO.BIT ;GET INITIAL WRITE BIT.
   19 100450  001402                           BEQ     1$
   20 100452  052701  040000                   BIS     #D.IW,R1            ;MOVE INTO PARAMETER BITS
   21 100456  010123              1$:          MOV     R1,(R3)+            ;PUT INTO BUFFER
   22 100460  016423  000006                   MOV     D.PAT(R4),(R3)+     ;PUT PATTERN NUMBER IN BUFFER
   23 100464  032701  000040                   BIT     #D.BE,R1            ;CHECK BEGIN/END PARAMETER BIT
   24 100470  001411                           BEQ     3$                  ;BRANCH IF NOT SET
   25
   26                                  ;RETURN BEGIN/END SETS
   27
   28 100472  012701  000021                   MOV     #4*4+1,R1           ;COUNT OF SETS TIMES WORDS PER SET PLUS COUNT WORD
   29 100476  010402                           MOV     R4,R2               ;GET INDEX INTO DRIVE TABLE
   30 100500  062702  000112                   ADD     #C.BEC,R2
   31 100504  012223              2$:          MOV     (R2)+,(R3)+         ;TRANSFER THE BEGIN/END SETS
   32 100506  005301                           DEC     R1
   33 100510  001375                           BNE     2$
   34 100512  000457                           BR      T4UPRX
   35
   36 100514  032764  000400  000004 3$:       BIT     #D.CYL,D.PRM(R4)    ;LOOK AT D.CYL BIT
   37 100522  001441                           BEQ     8$                  ;BRANCH IF NOT SET
   38
   39                                  ;RETURN TRACKS/GROUPS AND CYLINDERS
   40
   41 100524  005764  000112                   TST     D.BEC(R4)           ;CHECK IF ANY TRACKS/GROUPS
   42 100530  001421                           BEQ     6$                  ;BRANCH IF NONE
   43 100532  012701  000004                   MOV     #4,R1               ;COUNT OF CYLINDER WORDS
   44 100536  010402                           MOV     R4,R2
   45 100540  062702  000154                   ADD     #D.BCYL,R2
   46 100544  012223              4$:          MOV     (R2)+,(R3)+         ;CYLINDERS
   47 100546  005301                           DEC     R1
   48 100550  001375                           BNE     4$
   49 100552  012701  000010                   MOV     #8.,R1
   50 100556  010402                           MOV     R4,R2
   51 100560  062702  000112                   ADD     #D.BEC,R2
   52 100564  012223              5$:          MOV     (R2)+,(R3)+         ;TRACKS/GROUPS
   53 100566  005301                           DEC     R1
   54 100570  001375                           BNE     5$
   55 100572  000427                           BR      T4UPRX
   56
   57                                  ;RETURN CYLINDERS ONLY
   58
   59 100574  052763  000040  177774 6$:       BIS     #D.BE, 4(R3)        ;SET D.BE FOR DM PROGRAM
   60 100602  005023                           CLR     (R3)+               ;SEND ZERO BEGIN/END COUNT
   61 100604  012701  000004                   MOV     #4,R1
   62 100610  010402                           MOV     R4,R2
```

```
63 100612  062702  000154              ADD     #D.BCYL,R2
64 100616  012223              7$:      MOV     (R2)+,(R3)+        ;CYLINDERS
65 100620  005301                       DEC     R1
66 100622  001375                       BNE     7$
67 100624  000412                       BR      T4UPRX
68
69                                       ;RETURN ENTIRE AREA
70
71 100626  052763  000040  177774  8$:  BIS     #D.BE,-4(R3)       ;SET D.BE FOR DM PROGRAM
72 100634  005023                       CLR     (R3)+              ;BEGIN/END COUNT OF ZERO
73 100636  005023                       CLR     (R3)+              ;START CYLINDER OF ZERO
74 100640  005023                       CLR     (R3)+
75 100642  005023                       CLR     (R3)+              ;END CYLINDER NEGATIVE
76 100644  012723  177777              MOV     #-1,(R3)+
77 100650  000264                       SEZ
78 100652  000207              T4UPRX:  RETURN
```

```
    1
    2                                  ;T4BB1    DM REQUEST 5
    3                                  ;
    4                                  ;REQUEST FOR FIRST 14 BAD BLOCKS
    5                                  ;
    6                                  ;INPUTS:
    7                                  ;         R5 - CONTROLLER TABLE ADDRESS
    8                                  ;         R4 - MESSAGE DATA ADDRESS
    9                                  ;            (R4) DRIVE NUMBER
   10                                  ;         R3 - COMMAND DATA ADDRESS
   11                                  ;OUTPUTS:
   12                                  ;         COMMAND DATA FILLED WITH BAD BLOCKS
   13                                  ;            (R3) COUNT OF BAD BLOCKS
   14                                  ;          2.(R3) BAD BLOCK 1 (LOW)
   15                                  ;          4.(R3)             (HIGH)
   16                                  ;             )
   17                                  ;             )
   18                                  ;         56.(R3) BAD BLOCK 14 (LOW)
   19                                  ;         58.(R3)              (HIGH)
   20                                  ;         Z SET IF DATA RETURNED
   21                                  ;         Z CLEAR IF DRIVE NUMBER NOT ON THIS CONTROLLER
   22
   23 100654  011401              T4BB1:  MOV     (R4),R1               .GET DRIVE NUMBER
   24 100656  004737  102274              CALL    GTDRVT                ;GET DRIVE TABLE ADDRESS
   25 100662  001007                      BNE     T4BB1E                ;CHECK IF DRIVE FOUND
   26 100664  062704  000010              ADD     #D.BB,R4              ;INCREASE ADDRESS TO DATA TO COPY
   27 100670  012701  000035              MOV     #<1+<14.*2>>,R1       ;GET COUNT OF WORDS
   28 100674  012423              1$:     MOV     (R4)+,(R3)+           ;COPY THE WORDS
   29 100676  005301                      DEC     R1
   30 100700  001375                      BNE     1$
   31 100702  000207              T4BB1E: RETURN
```

```
 1
 2                                    ;T4BB2 - DM REQUEST 6
 3                                    ;
 4                                    ;REQUEST LAST TWO BAD BLOCKS
 5                                    ;
 6                                    ;INPUTS:
 7                                    ;       R5 - CONTROLLER TABLE ADDRESS
 8                                    ;       R4 - MESSAGE DATA ADDRESS
 9                                    ;          (R4) DRIVE NUMBER
10                                    ;       R3 - COMMAND DATA ADDRESS
11                                    ;OUTPUTS:
12                                    ;       COMMAND DATA FILLED WITH BAD BLOCKS 15 AND 16
13                                    ;       Z SET IF DATA RETURNED
14                                    ;       Z CLEAR IF UNIT NUMBER NOT ON THIS CONTROLLER
15
16 100704  011401            T4BB2:  MOV     (R4),R1               ;GET DRIVE NUMBER
17 100706  004737  102274            CALL    GTDRVT                ;GET DRIVE TABLE ADDRESS
18 100712  001007                    BNE     T4BB2E                ;CHECK IF DRIVE FOUND
19 100714  062704  000102            ADD     #D.BB15,R4            ;INCREASE ADDRESS TO DATA TO COPY
20 100720  012701  000004            MOV     #4,R1                 ;GET COUNT OF WORDS
21 100724  012423            1$:     MOV     (R4)+,(R3)+           ;COPY THE WORDS
22 100726  005301                    DEC     R1
23 100730  001375                    BNE     1$
24 100732  000207            T4BB2E: RETURN
```

M14

```
 1
 2                                        ;T4SOFT - DM REQUEST 7
 3                                        ;
 4                                        ;ADD TO SOFT ERROR AND ECC COUNTS
 5                                        ;
 6                                        ;INPUTS:
 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
15                                        ;       R3 - COMMAND DATA ADDRESS
16
17 100734  012401              T4SOFT: MOV     (R4)+,R1           ;GET DRIVE NUMBER
18 100736  010402                      MOV     R4,R2              ;SAVE DATA ADDRESS
19 100740  004737  102274              CALL    GTDRVT             ;GET DRIVE TABLE ADDRESS
20 100744  001005                      BNE     1$                 ;CHECK IF DRIVE FOUND
21 100746  062264  000172              ADD     (R2)+,D.SERR(R4)   ;ADD TO SOFT ERROR COUNT
22 100752  062264  000176              ADD     (R2)+,D.ECCC(R4)   ;ADD TO ECC COUNT
26 100756  000264                      SEZ                        ;EXIT
27 100760  000207              1$:     RETURN
```

N14

```
 1
 2 100762                         T4SEEK:
 3                               ;       DM REQUEST 8.
 4                               ;
 5                               ;RECORD 1000 SEEKS COMPLETED ON DRIVE
 6                               ;
 7                               ;INPUTS:
 8                               ;       R5 - CONTROLLER TABLE ADDRESS
 9                               ;       R4    MESSAGE DATA ADDRESS
10                               ;          (R4) DRIVE NUMBER
11                               ;       R3    COMMAND DATA ADDRESS
12                               ;
13 100762 011401                         MOV     (R4),R1         ; GET DRIVE NUMBER
14 100764 004737  102274                 CALL    GTDRVT          ; GET DRIVE TABLE ADDRESS
15 100770 001003                         BNE     SEKERE          ; CHECK IF DRIVE FOUND
16 100772 005264  000174                 INC     D.SEEK(R4)      ; COUNT THE BITS TRANSFERRED
17 100776 000264                         SEZ                     ; NORMAL RETURN
18 101000 000207                 SEKERE: RETURN
```

```
  1                                      ;T4MXFR    DM REQUEST 9.
  2                                      ;
  3                                      ;
  4                                      ;RECORD 1M BITS TRANSFERRED ON UNIT. COMPARE TO TRANSFER LIMIT AND
  5                                      ;REPORT LIMIT REACHED.
  6                                      ;
  7                                      ;INPUTS:
  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                                      ;OUTPUTS:
 15                                      ;          (R3) BIT 15 SET IF TRANSFER LIMIT REACHED
 16                                      ;        MESSAGE PRINTED IF TRANSFER LIMIT REACHED
 17                                      ;        Z CLEAR IF DRIVE NUMBER NOT ON THIS CONTROLLER
 18
 19 101002  010402                       T4MXFR: MOV    R4,R2                   ;GET MESSAGE DATA ADDRESS
 20 101004  011401                               MOV    (R4),R1                 ;GET DRIVE NUMBER
 21 101006  004737  102274                       CALL   GTDRVT                  ;GET DRIVE TABLE ADDRESS
 22 101012  001053                               BNE    MXFERE                  ;CHECK IF DRIVE FOUND
 23 101014  005764  000002                       TST    D.UNIT(R4)              ;SEE IF UNIT HAS BEEN DROPPED
 24 101020  100003                               BPL    1$                      ;CONTINUE IF STILL TO BE TESTED
 26 101022  052713  100000                       BIS    #BIT15,(R3)             ;TELL DM PROGRAM TO STOP TESTING THIS UNIT
 27 101026  000444                               BR     MXFERX                  ;  AND EXIT WITHOUT ADDING TO ADDING TO COUNTS
 28
 29 101030                               1$:
 44 101030  066264  000002  000166               ADD    2(R2),D.XFRR(R4)        ;ADD MEGABITS READ
 45 101036  066264  000004  000164               ADD    4(R2),D.XFRW(R4)        ;ADD MEGABITS WRITTEN
 46 101044  005737  064376                        TST    SFPTBL.SO.XL            ;SEE IF LIMIT SPECIFIED
 47 101050  001433                               BEQ    MXFERX                  ;BRANCH IF NOT
 48 101052  026437  000166  064376               CMP    D.XFRR(R4),SFPTBL.SO.XL ;CHECK IF LIMIT REACHED
 49 101060  103427                               BLO    MXFERX                  ;BRANCH IF LIMIT NOT REACHED
 50 101062  104421                               TRAP   C$RFLA
 51 101064  032700  000040                       BIT    #IDU,R0                 ;SEE IF DROPPING UNITS IS INHIBITED
 52 101070  001023                               BNE    MXFERX
 53 101072  052713  100000                       BIS    #BIT15,(R3)             ;SET DROP UNIT BIT
 54 101076  042765  000010  000014               BIC    #CT.MSG,C.FLG(R5)       ;CLEAR MESSAGE RECEIVED FLAG
 55 101104  011446                               MOV    (R4),(SP)               ;PUSH (R4) ON STACK
    101106  011546                               MOV    (R5),(SP)               ;PUSH (R5) ON STACK
    101110  016446  000002                       MOV    D.UNIT(R4),(SP)         ;PUSH D.UNIT(R4) ON STACK
    101114  004137  075704                       JSR    R1,LPNTX                ;CALL   LPNTX PRINT ROUTINE
    101120  065713                               .WORD  MESSG                   ;ADDRESS OF ASCIZ STRING
    101122  000006                               .WORD  ARG.CT                  ;ARGUMENT COUNT . 2
 56 101124  004737  106362                       CALL   RNTIME                  ;PRINT RUNTIME
 57 101130  004137  075704                       JSR    R1,LPNTX                ;CALL   LPNTX PRINT ROUTINE
    101134  065170                               .WORD  MXFERP                  ;ADDRESS OF ASCIZ STRING
    101136  000000                               .WORD  ARG.CT                  ;ARGUMENT COUNT . 2
 58 101140  000264                       MXFERX: SEZ                            ;NORMAL RETURN
 59 101142  000207                       MXFERE: RETURN
```

```
   1
   2                                          ;UTOTST - DM REQUEST 10
   3                                          ;
   4                                          ;TELL DM PROGRAM WHICH DRIVES ARE SELECTED FOR TESTING
   5                                          ;AND CLEAR STATISTICS IN DRIVE TABLE
   6                                          ;
   7                                          ;INPUTS:
   8                                          ;       R5 - CONTROLLER TABLE ADDRESS
   9                                          ;       R4 - MESSAGE DATA ADDRESS
  10                                          ;            (NO DATA)
  11                                          ;       R3    COMMAND DATA ADDRESS
  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
  19 101144  010504            UTOTST:  MOV     R5,R4                   ;GET ADDRESS OF CONTROLLER TABLE
  20 101146  062704   000020            ADD     #C.DR0,R4               ;BUMP TO DRIVE TABLE POINTERS
  21 101152  012702   000010            MOV     #8.,R2                  ;GET COUNT OF PORTS
  22 101156  012400            UTOT1:   MOV     (R4)+,R0                ;SEE IF DRIVE TABLE POINTER EXISTS
  23 101160  001415            BEQ     UTOT2                   ;BRANCH IF NOT
  24 101162  005760   000002            TST     D.UNIT(R0)              ;LOOK IF UNIT AVAILABLE FOR TESTING
  25 101166  100410            BMI     UTOT1A
  27 101170  011023            MOV     (R0),(R3)+              ;LOAD DRIVE NUMBER FROM TABLE
  28 101172  062700   000164            ADD     #D.XFRW,R0              ;CLEAR STATISTICS IN DRIVE TABLE
  29 101176  012701   000011            MOV     #<D.SIZE-D.XFRW>/2,R1
  30 101202  005020            1$:     CLR     (R0)+
  31 101204  005301            DEC     R1
  32 101206  001375            BNE     1$
  33 101210  005302            UTOT1A:  DEC     R2                      ;COUNT THE DRIVE TABLES
  34 101212  001361            BNE     UTOT1                   ;REPEAT FOR EACH TABLE
  35 101214  012723   100000            UTOT2:   MOV     #BIT15,(R3)+            ;TERMINATE LIST
  36 101220  013723   064656            MOV     DLL,(R3)+               ;GET DLL CONTROL WORD
  37 101224  001407            BEQ     UTOT4                   ; IF NON ZERO
  38 101226  012701   064660            MOV     #DLLDR,R1               ;   TRANSFER ALL DLL WORDS INTO BUFFER
  39 101232  012702   000020            MOV     #<DLLNAM+4-DLLDR>,R2
  40 101236  012123            UTOT3:   MOV     (R1)+,(R3)+
  41 101240  005302            DEC     R2
  42 101242  001375            BNE     UTOT3
  43 101244  000264            UTOT4:   SEZ
  44 101246  000207            RETURN                          ;RETURN WITH Z SET
```

```
  1                                      ;ERRMES     DM REQUEST 11
  2                                      ;
  3                                      ;PRINT AN ERROR MESSAGE
  4                                      ;
  5                                      ;
  6                                      ;INPUTS:
  7                                      ;          R5    CONTROLLER TABLE ADDRESS
  8                                      ;          R4    MESSAGE DATA ADDRESS
  9                                      ;              (R4)   ERROR PC IN DM PROGRAM
 10                                      ;            2.(R4)   <15:14> ERROR TYPE
 11                                      ;                     <13:0 > ERROR NUMBER
 12                                      ;            4.(R4)   DRIVE NUMBER ( -1 IF NOT GIVEN)
 13                                      ;            6.(R4)   MESSAGE POINTER
 14                                      ;            8.(R4)   OPTIONAL PARAMETERS FOR ERROR PRINT ROUTINE
 15                                      ;           10.(R4)                    "
 16                                      ;               )                      "
 17                                      ;               )                      "
 18                                      ;           58.(R4)                    "
 19                                      ;          R3 - COMMAND DATA ADDRESS
 20                                      ;OUTPUTS:
 21                                      ;          COMMAND PACKET CONTAINING THE FOLLOWING:
 22                                      ;              (R3) - BIT 15 SET IF FATAL ERROR TO INDICATE DRIVE SHOULD NO LONGER BE TESTED
 23                                      ;          Z SET TO INDICATE DATA RETURNED
 24                                      ;          Z CLEAR IF DRIVE NUMBER NOT ON THIS CONTROLLER
 25 101250                              ;ERRMES:
 36 101250 005764 000002                       TST     2(R4)                   ;CHECK IF FATAL ERROR
 37 101254 100406                              BMI     5$                      ;BRANCH IF NOT
 38 101256 104421                              TRAP    C$RFLA
 39 101260 032700 000040                       BIT     #IDU.R0                 ;SEE IF ALLOWED TO DROP UNITS
 40 101264 001014                              BNE     6$                      ;BRANCH IF NOT
 52 101266 052713 100000                       BIS     #BIT15,(R3)             ;SET DROP DRIVE BIT
 53 101272 016400 000002            5$:        MOV     2(R4),R0                ;SEE IF SOFT ERROR
 54 101276 005100                              COM R0
 55 101300 032700 140000                       BIT     #140000,R0
 56 101304 001004                              BNE     6$                      ;BRANCH IF NOT
 57 101306 032737 000400 064400               BIT     #SM.SSF,SO.BIT+SFPTBL   ;SEE IF SOFT ERRORS SUPPRESSED
 58 101314 001063                              BNE     ERRMSX                  ;DON'T PRINT IF SO
 59 101316                            6$:
 60 101316 042765 000010 000014               BIC     #CT.MSG.C.FLG(R5)       ;CLEAR MESSAGE RECEIVED FLAG
 62 101324 022737 000004 064444               CMP     #4,TNUM                 ;ARE WE DOING DISK EXERCISER TEST ?
 63 101332 001004                              BNE     7$                      ;BRANCH IF NOT
 65 101334 032737 001000 064400               BIT     #SM.LOG,SFPTBL+SO.BIT   ; SEE IF LOG BEING USED
 66 101342 001005                              BNE     ERRMSL
 67 101344 004737 103250            7$:        CALL    PNTERR                  ;IF NOT, PRINT THE ERROR MESSAGE
 68 101350 103045                              BCC ERRMSX                      ;IF DRIVE HASN'T BEEN DROPPED, PRINT
 72 101352 000244                              CLZ                             ;ELSE RETURN
 73 101354 000207                              RETURN
```

E15

```
    1
    2 101356  005737  064650            ERRMSL: TST     LBUFS              ;SEE IF LOG BUFFER ESTABLISHED
    3 101362  001016                            BNE     1$                 ;   LBUFS CONTAINS ADDRESS IF ESTABLISHED
    4 101364  013701  064432                    MOV     DMPROG,R1
    5 101370  005721                            TST     (R1)+              ; LBUFS < (DMPROG)+2
    6 101372  010137  064650                    MOV     R1,LBUFS
    7 101376  010137  064652                    MOV     R1,LBUFN
    8 101402  067701  163024                    ADD     @DMPROG,R1
    9 101406  005741                            TST     (R1)               ; LBUFE <- (LBUFS) + ((DMPROG))   2
   10 101410  010137  064654                    MOV     R1,LBUFE
   11 101414  005037  064442                    CLR     FNUM
   12 101420  013701  064652            1$:     MOV     LBUFN,R1           ;GET ADDRESS OF DATA STORAGE AREA
   13 101424  062737  000106  064652            ADD     #HC.BSZ,LBUFN      ;ADD BYTES OF STORAGE NEEDED
   14 101432  023737  064652  064654            CMP     LBUFN,LBUFE        ;SEE IF ENOUGH ROOM
   15 101440  103007                            BHIS    3$                 ;   BRANCH IF NOT
   16 101442  010521                            MOV     R5,(R1)+           ;STORE CONTROLLER TABLE ADDRESS
   17 101444  012700  000042                    MOV     #<HC.BSZ-2>/2,R0   ;GET COUNT OF REST OF DATA IN WORDS
   18 101450  012421                    2$:     MOV     (R4)+,(R1)+        ;STORE DATA
   19 101452  005300                            DEC     R0
   20 101454  001375                            BNE     2$
   21 101456  000402                            BR      ERRMSX
   22 101460  010137  064652            3$:     MOV     R1,LBUFN           ;RESTORE OLD VALUE OF LBUFN
   23 101464  000264            ERRMSX: SEZ
   24 101466  000207                            RETURN
```

```
 1                                 ;ERRMC   DM REQUEST 12.
 2                                 ;
 3                                 ;REPORT AN ERROR MESSAGE IDENTICAL TO DM REQUEST ERRMES
 4                                 ;THEN ADD ONE TO THE ERROR COUNT FOR THE DRIVF AND SEE IF
 5                                 ;ERROR LIMIT REACHED.
 6
 7                                 ;
 8                                 ;INPUTS:
 9                                 ;         R5 - CONTROLLER TABLE ADDRESS
10                                 ;         R4 - MESSAGE DATA ADDRESS
11                                 ;              (R4)   ERROR PC IN DM PROGRAM
12                                 ;            2.(R4)   < 9:8 > ERROR TYPE
13                                 ;                     < 7:0 > ERROR NUMBER
14                                 ;            4.(R4)   DRIVE NUMBER ( 1 IF NOT GIVEN)
15                                 ;            6.(R4)   <15:12> TYPE
16                                 ;                     <11:0 > MESSAGE POINTER
17                                 ;            8.(R4)   OPTIONAL PARAMETERS FOR ERROR PRINT ROUTINE
18                                 ;           10.(R4)                    "
19                                 ;             )                       "
20                                 ;             )                       "
21                                 ;           58.(R4)                   "
22                                 ;         R3 - COMMAND DATA ADDRESS
23                                 ;OUTPUTS:
24                                 ;         COMMAND PACKET CONTAINING THE FOLLOWING:
25                                 ;           (R3) BIT 15 SET IF ERROR COUNT REACHED
26                                 ;                TO INDICATE DRIVE SHOULD NO LONGER BE TESTED.
27                                 ;         Z CLEAR IF DRIVE NUMBER NOT ON THIS CONTROLLER
28                                 ;         Z SET TO INDICATE DATA RETURNED
29
30 101470                         ERRMC:
   101470   010446                         MOV     R4, (SP)                 .;PUSH R4 ON STACK
31 101472   004737   101250                CALL    ERRMES                   ; CALL REQUEST ERRMES
32 101476   012604                         MOV     (SP)+,R4                 ;;POP STACK INTO R4
33 101500   005713                         TST     (R3)                     ;SEE IF UNIT ALREADY TO BE DROPPED
34 101502   100436                         BMI     3$                       ;   IF SO, JUST EXIT NOW
35 101504   016401   000004                MOV     4(R4),R1                 ; GET DRIVE NUMBER
36 101510   016402   000002                MOV     2(R4),R2                     ;GET ERROR TYPE
37 101514   004737   102274                CALL    GTDRVT                   ; GET DRIVE TABLE
38 101520   001031                         BNE     5$                       ; EXIT IF NO TABLE FOR UNIT
39 101522   042702   037777                BIC     #^C140000,R2
40 101526   022702   100000                CMP     #100000,R2               ;CHECK IF HARD ERROR
41 101532   001022                         BNE     3$                       ;BRANCH IF NOT
42 101534   005264   000170                INC     D.HERR(R4)               ; COUNT THE ERROR
43 101540   026437   000170   064374       CMP     D.HERR(R4),SFPTBL+SO.EL  ; CHECK IF AT LIMIT
44 101546   103414                         BLO     3$                       ; IF LIMIT REACHED, BRANCH
45 101550   104421                         TRAP    C$RFLA
46 101552   032700   000040                BIT     #IDU,R0                  ;SEE IF DROPPING UNITS INHIBITED
47 101556   001010                         BNE     3$                       ;BRANCH IF SO
55 101560   016446   000002                MOV     D.UNIT(R4), (SP)             ;PUSH D.UNIT(R4) ON STACK
   101564   004137   075704                JSR     R1,LPNTX                 ;CALL   LPNTX PRINT ROUTINE
   101570   065245                         .WORD   ERRLIM                   ;ADDRESS OF ASCIZ STRING
   101572   000002                         .WORD   ARG.CT                   ;ARGUMENT COUNT + 2
56 101574   052713   100000                BIS     #BIT15,(R3)              ;SET STOP TESTING BIT
62 101600   000264                 3$:     SEZ                              ; SET Z FOR NORMAL RETURN
63 101602   000207                         RETURN                          ; RETURN TO CALLING PROGRAM
64
65 101604   000244                 5$:     CLZ                              ; FLAG AS ERROR
```

66 101606 000207   RETURN   ; RETURN TO CALLING PROGRAM

',

```
 1                                              ;MESSAG - DM REQUEST 13.
 2
 3                                              ;
 4                                              ;PRINT A MESSAGE WITH HEADER AS FOLLOWS:
 5                                              ;   "UNIT XX UDA AT XXXXXX DRIVE XXX RUNTIME HH:MM:SS '
 6                                              ;ENTIRE MESSAGE IS PRINTED WITH PRINTX CALLS.
 7                                              ;
 8                                              ;INPUTS:
 9                                              ;          R5    CONTROLLER TABLE ADDRESS
10                                              ;          R4 - MESSAGE DATA ADDRESS
11                                              ;           (R4) DRIVE NUMBER
12                                              ;         2.(R4) MESSAGE POINTER
13                                              ;         2.(R4) MESSAGE POINTER
14                                              ;         4.(R4) OPTIONAL MESSAGE PARAMETERS
15                                              ;             )
16                                              ;             )
17                                              ;        58.(R4) COMMAND DATA ADDRESS
18
19 101610  042765  000010  000014  MESSAG: BIC   #CT.MSG.C.FLG(R5)      ;CLEAR MESSAGE RECEIVED FLAG
20 101616  012401                          MOV   (R4)+,R1               ;GET DRIVE NUMBER
21 101620  010446                          MOV   R4,-(SP)               ;;PUSH R4 ON STACK
22 101622  004737  102274                  CALL  GTDRVT                 ;GET DRIVE TABLE ADDRESS
23 101626  001033                          BNE   1$                     ;CHECK IF DRIVE FOUND
24 101630  005764  000002                  TST   D.UNIT(R4)             ;IF UNIT DROPPED FROM TESTING
25 101634  100430                          BMI   1$                     ;   DON'T PRINT ANYTHING
26 101636  011446                          MOV   (R4),-(SP)             ;PUSH (R4) ON STACK
   101640  011546                          MOV   (R5),-(SP)             ;PUSH (R5) ON STACK
   101642  016446  000002                  MOV   D.UNIT(R4),-(SP)       ;PUSH D.UNIT(R4) ON STACK
   101646  004137  075704                  JSR   R1,LPNTX               ;CALL  LPNTX PRINT ROUTINE
   101652  065713                          .WORD MESSG                  ;ADDRESS OF ASCIZ STRING
   101654  000006                          .WORD ARG.CT                 ;ARGUMENT COUNT + 2
27 101656  004737  106362                  CALL  RNTIME                 ;GET RUNTIME PARAMETERS
28 101662  012604                          MOV   (SP)+,R4               ;;POP STACK INTO R4
29 101664  012402                          MOV   (R4)+,R2               ;GET MESSAGE POINTER
30 101666  006302                          ASL   R2                     ;DOUBLE TO MAKE BYTE OFFSET
31 101670  063702  064432                  ADD   DMPROG,R2              ;ADD TO START OF MESSAGE STRINGS
32 101674  067702  162532                  ADD   @DMPROG,R2             ;ADD SIZE OF MAIN PROGRAM
33 101700  105712                          TSTB  (R2)                   ;CHECK FIRST BYTE
34 101702  001001                          BNE   2$                     ;IF ZERO
35 101704  005202                          INC   R2                     ;   INCREMENT TO NEXT BYTE
36 101706  004737  075772          2$:     CALL  OSTRNG                 ;OUTPUT ACCORDING TO STRING
37 101712  0 0264                           SEZ
38 101714  000207                          RETURN
39 101716                          1$:
   101716  012604                          MOV   (SP)+,R4               ;;POP STACK INTO R4
40 101720  000207                          RETURN
```

I15

```
 1
 2                                      ;DONE - DM REQUEST 14
 3                                      ;
 4                                      ;MARK DM PROGRAM AS NO LONGER RUNNING
 5                                      ;
 6                                      ;INPUTS:
 7                                      ;       R5 - CONTROLLER TABLE ADDRESS
 8                                      ;       R4 - MESSAGE DATA ADDRESS
 9                                      ;           (NO DATA)
10                                      ;       R3 - COMMAND DATA ADDRESS
11                                      ;OUTPUTS:
12                                      ;       Z CLEAR TO DROP UNIT FROM TESTING
13
14 101722  000244                      DONE:   CLZ                     ;DROP UNIT FROM TESTING
15 101724  000207                              RETURN
```

```
 1                              .SBTTL   PRE PROGRAMMED SUBROUTINES
 2
 3                           ;PRE PROGRAMMED ROUTINE 1
 4                           ;
 5                           ;CALL ALTERNATE PRINT STRING IN DM PROGRAM IMAGE
 6
 7 101726                   CALR1:
   101726  010246              MOV     R2,(SP)         ;;PUSH R2 ON STACK
 8 101730  012402              MOV     (R4)+,R2        ;GET NEW STRING POINTER
 9 101732  006302              ASL     R2              ;DOUBLE FOR WORD COUNT
10 101734  063702  064432      ADD     DMPROG,R2       ;ADD START OF STRING STORAGE
11 101740  067702  162466      ADD     @DMPROG,R2      ;ADD SIZE OF MAIN PROGRAM
12 101744  004737  075772      CALL    OSTRNG          ;OUTPUT USING THIS STRING
13 101750  012602              MOV     (SP)+,R2        ;;POP STACK INTO R2
14 101752  000207              RETURN                  ;NOW CONTINUE THE OLD STRING
```

K15

```
    1
    2                                      ;PRE PROGRAMMED ROUTINE 2
    3                                      ;
    4                                      ;PRINT AN SDI DIAGNOSE RESPONSE
    5
    6 101754                      CALR2:
      101754  010246                       MOV    R2,-(SP)         ;;PUSH R2 ON STACK
    7 101756  012402                       MOV    (R4)+,R2         ;GET COUNTS
    8 101760  010246                       MOV    R2,-(SP)         ;;PUSH R2 ON STACK
    9 101762  042702  177400               BIC    @177400,R2       ;GET BINARY COUNT
   10 101766  001414                       BEQ    2$               ;BYPASS BINARY IF COUNT IS ZERO
   11 101770  012700  000020      1$:      MOV    @16.,R0          ;RADIX IS HEX
   12 101774  012701  000040               MOV    @32.,R1          ;32 BIT NUMBERS
   13 102000  004737  102450               CALL   PNTNUS           ;PRINT THE NUMBER
   14 102004  112700  000015               MOVB   @CR,R0           ;STORE @CR IN R0 AND
      102010  004737  075506               JSR    PC,PRINTC        ;PRINT THE CHARACTER.
   15 102014  005302                       DEC    R2
   16 102016  001364                       BNE    1$
   17 102020                      2$:
      102020  012601                       MOV    (SP)+,R1         ;;POP STACK INTO R1
   18 102022  000301                       SWAB R1                 ;GET ASCII COUNT
   19 102024  042701  177400               BIC    @177400,R1
   20 102030  001406                       BEQ    3$               ;BYPASS IS COUNT IS ZERO
   21 102032  004737  076066               CALL   CON.A1           ;PRINT THE ASCII
   22 102036  112700  000015               MOVB   @CR,R0           ;STORE @CR IN R0 AND
      102042  004737  075506               JSR    PC,PRINTC        ;PRINT THE CHARACTER.
   23 102046                      3$:
      102046  012602                       MOV    (SP)+,R2         ;;POP STACK INTO R2
   24 102050  000207                       RETURN
```

L15

CZUDCEO UDA & DISK DRV DIAG MACRO V05.00 Wednesday 04-Jan 84 16:12 Page 147
PRE PROGRAMMED SUBROUTINES

```
 1
 2                                          ;PRE-PROGRAMMED ROUTINE 3
 3                                          ;
 4                                          ;DECIDE WHETHER TO PRINT RBN
 5
 6                                          ;FOUR PARAMETERS ARE PROVIDED FOR THIS ROUTINE. THE FIRST PARAMETER
 7                                          ;SHOULD BE CHECKED TO SEE IF BIT 7 IS SET:
 8                                          ;  IF SET   TURN INTO A CALL TO ROUTINE 1 (WHICH WILL USE OTHER 3 PARAMETERS)
 9                                          ;  IF CLEAR - SKIP OVER NEXT 3 PARAMETERS AND END ROUTINE
10
11 102052 032724 000200    CALR3:  BIT     #BIT7,(R4)+          ;CHECK BIT 7 IN FIRST PARAMETER WORD
12 102056 001323                   BNE     CALR1                ;IF SET, TURN INTO A CALR1
13 102060 062704 000006            ADD     #6,R4                ;ELSE, SKIP OVER NEXT 3 PARAMETERS
14 102064 000207                   RETURN
```

M15

```
   1
   2                                      ;PRE-PROGRAMMED ROUTINE 4
   3                                      ;
   4                                      ;PRINT BASIC LINE FOR HOST PROGRAM ERROR WITHOUT UDA ADDRESS
   5                                      ;THEN SWITCH TO EXTENDED FORMAT
   6
   7 102066                       CALR4:
     102066  012746  067047              MOV     #BAS,-(SP)         ;PUSH #BAS ON STACK
     102072  012746  067047              MOV     #BAS,-(SP)         ;PUSH #BAS ON STACK
     102076  012746  067047              MOV     #BAS,-(SP)         ;PUSH #BAS ON STACK
     102102  012746  066651              MOV     #BASNO,(SP)        ;PUSH #BASNO ON STACK
     102106  004137  075674              JSR     R1.LPNTB           ;CALL   LPNTB PRINT ROUTINE
     102112  067050                      .WORD   BASLN              ;ADDRESS OF ASCIZ STRING
     102114  000010                      .WORD   ARG.CT             ;ARGUMENT COUNT * 2
   8 102116  004737  106362              CALL    RNTIME
   9 102122  112700  000015              MOVB    #CR,R0             ;STORE #CR IN R0 AND
     102126  004737  075506              JSR     PC.PRINTC          ;PRINT THE CHARACTER.
  10 102132  012737  075612  075770      MOV     #P),PTYPE
  11 102140  000207                      RETURN
```

```
 1                                         ;PRE-PROGRAMMED ROUTINE 5
 2                                         ;
 3                                         ;PRINT BASIC LINE FOR HOST PROGRAM ERROR WITH UDA ADDRESS
 4                                         ;THEN SWITCH TO EXTENDED FORMAT
 5
 6
 7 102142                       CALR5:
   102142  012746  067047              MOV     #BAS,-(SP)          ;PUSH #BAS ON STACK
   102146  012746  067047              MOV     #BAS,-(SP)          ;PUSH #BAS ON STACK
   102152  011546                      MOV     (R5),-(SP)          ;PUSH (R5) ON STACK
   1C2154  012746  067013              MOV     #BASL2,(SP)         ;PUSH #BASL2 ON STACK
   102160  012746  066651              MOV     #BASNO,(SP)         ;PUSH #BASNO ON STACK
   102164  004137  075674              JSR     R1,LPNTB            ;CALL    LPNTB PRINT ROUTINE
   102170  067050                      .WORD   BASLN               ;ADDRESS OF ASCIZ STRING
   102172  000012                      .WORD   ARG.CT              ;ARGUMENT COUNT * 2
 8 102174  004737  106362              CALL    RNTIME
 9 102200  112700  000015              MOVB    #CR,RO              ;STORE #CR IN RO AND
   102204  004737  075506              JSR     PC,PRINTC           ;PRINT THE CHARACTER.
10 102210  012737  075612  075770      MOV     #PX,PTYPE
11 102216  000207                      RETURN
```

```
   1
   2                            ;PRE PROGRAMMED ROUTINE 6
   3                            ;
   4                            ;CALL ALTERNATE PRINT ROUTINE IN PDP 11 MEMORY
   5
   6 102220                     CALR6:
     102220  010246                   MOV     R2,(SP)         ;;PUSH R2 ON STACK
   7 102222  012402                   MOV     (R4)+,R2        ;GET NEW STRING POINTER
   8 102224  004737  075772           CALL    OSTRNG          ;OUTPUT USING THIS STRING
   9 102230  012602                   MOV     (SP)+,R2        ;;POP STACK INTO R2
  10 102232  000207                   RETURN                  ;NOW CONTINUE THE OLD STRING
```

C16

```
     1
     2                                      ;PRE PROGRAMMED ROUTINE 7
     3                                      ;
     4                                      ;PRINT "REPLACE UDA MODULE M7161'
     5
     6 102234                              CALR7:
       102234   010246                             MOV      R2, (SP)           ;;PUSH R2 ON STACK
     7 102236   012702   074215                    MOV      #XFRU,R2
     8 102242   004737   075772                    CALL     OSTRNG
     9 102246   012602                             MOV      (SP)+,R2           ;;POP STACK INTO R2
    10 102250   000207                             RETURN
```

D16

```
 1
 2                                      ;PRE PROGRAMMED ROUTINE 8
 3                                      ;
 4                                      ;PRINT    "   UDASA CONTAINS   XXXXXX"
 5
 6 102252                              CALR8:
   102252   010246                         MOV      R2, (SP)              ;;PUSH R2 ON STACK
 " 102254   012702   074164               MOV      @XSA,R2
 8 102260   004737   075772               CALL     OSTRNG
 9 102264   012602                         MOV      (SP)+,R2             ;;POP STACK INTO R2
10 102266   000207                         RETURN
```

E16

```
   1
   2                              ;PRE-PROGRAMMED ROUTINE 9
   3                              ;
   4                              ;      REPRINT LAST NUMBER
   5                              ;              R4 -> TABLE
   6 102270  005744      CALR9:   TST      (R4)
   7 102272  000207               RETURN
```

F 16

```
 1
 2                                      .GTDRVT
 3                                      ;
 4                                      ;GET DRIVE TABLE POINTER
 5                                      ;
 6                                      ;INPUTS:
 7                                      ;       R5   CONTROLLER TABLE ADDRESS
 8                                      ;       R1 - DRIVE NUMBER
 9                                      ;OUTPUTS:
10                                      ;       R4 - DRIVE TABLE ADDRESS
11                                      ;       L$LUN - LOADED WITH UNIT NUMBER OF DRIVE
12                                      ;       Z CLEAR IF DRIVE TABLE NOT FOUND AFTER ERROR PRINTED
13
14 102274                              GTDRVT:
   102274  010246                              MOV     R2,-(SP)           ;;PUSH R2 ON STACK
15 102276  010504                              MOV     R5,R4              ;GET CONTROLLER TABLE ADRESS
16 102300  062704  000020                      ADD     #C.DR0,R4          ;ADD OFFSET TO DRIVE TABLE ADDRESS
17 102304  012702  000010                      MOV     #8.,R2             ;GET COUNT OF DRIVES
18 102310  005714                      1$:     TST     (R4)               ;CHECK IF AN ADDRESS HERE
19 102312  001406                              BEQ     3$
20 102314  027401  000000                      CMP     @(R4),R1           ;COMPARE DRIVE NUMBERS
21 102320  001412                              BEQ     4$                 ;BRANCH IF A MATCH
22 102322  005724                      2$:     TST     (R4)+              ;BUMP ADDRESS
23 102324  005302                              DEC     R2
24 102326  001370                              BNE     1$                 ;LOOK AT ALL OF THEM
25 102330                              3$:
   102330  104455                              TRAP    C$ERDF
   102332  000043                              .WORD   35
   102334  000000                              .WORD   0
   102336  075000                              .WORD   ERRO35
26 102340  012602                              MOV     (SP)+,R2           ;;POP STACK INTO R2
27 102342  000244                              CLZ                        ;CLEAR Z AS ERROR FLAG
28 102344  000207                              RETURN
29
30 102346  011404                      4$:     MOV     (R4),R4            ;(ET ADDRESS OF TABLE
31 102350  116437  000002  002074              MOVB    D.UNIT(R4),L$LUN   ;(ET UNIT NUMBER
32 102356  012602                              MOV     (SP)+,R2           ;;POP STACK INTO R2
33 102360  000264                              SEZ                        ;(ET Z FLAG
34 102362  000207                              RETURN
```

```
 1                              ;GETCNT
 2                              ;
 3                              ;
 4                              ;GET COUNT IN NEXT CHARACTERS OF STRING POINTED TO BY R2.
 5                              ;NUMBER WILL BE IN DECIMAL. IF NO NUMBER, RETURN A
 6                              ;DEFAULT OF 1.
 7                              ;
 8                              ;INPUTS:
 9                              ;     R2 - POINTER TO ASCII STRING
10                              ;OUTPUTS:
11                              ;     R1    NUMBER READ OR A ONE
12                              ;     R2 - POINTING TO CHARACTER AFTER NUMBER
13
14 102364                      GETCNT:
   102364  010046                      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
15 102366  005001                      CLR     R1              ;START WITH ZERO COUNT
16 102370  121227  000060      GETCNX: CMPB    (R2),#'0        ;CHECK IF CHARACTER A DIGIT
17 102374  103415                      BLO     GETCDN          ;BRANCH IF LOWER THAN ZERO
18 102376  121227  000071              CMPB    (R2),#'9
19 102402  101012                      BHI     GETCDN          ;BRANCH IF HIGHER THAN NINE
20 102404  006301                      ASL     R1              ;MULTIPLY NUMBER BY 10
21 102406  010100                      MOV     R1,R0           ;  SAVE 2N
22 102410  006301                      ASL     R1              ;  COMPUTE 4N
23 102412  006301                      ASL     R1              ;  COMPUTE 8N
24 102414  060001                      ADD     R0,R1           ;  8N + 2N = 10N
25 102416  112200                      MOVB    (R2)+,R0        ;GET DIGIT FROM STING
26 102420  162700  000060              SUB     #'0,R0          ;GET RID OF ASCII
27 102424  060001                      ADD     R0,R1           ;ADD TO NUMBER
28 102426  000760                      BR      GETCNX          ;GO TO NEXT CHARACTER
29 102430  005701              GETCDN: TST     R1              ;CHECK IF NUMBER IS ZERO
30 102432  001001                      BNE     GETCXX          ;IF ZERO, CHANGE
31 102434  005201                      INC     R1              ;  TO DEFAULT OF ONE
32 102436                      GETCXX:
   102436  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
33 102440  000207                      RTS     PC
```

H16

```
 1
 2                                          ;PNTNUM
 3                                          ;
 4                                          ;PRINT A NUMBER
 5                                          ;
 6                                          ;INPUTS:
 7                                          ;       R1 - RADIX OF NUMBER
 8                                          ;       R2 - ASCII STRING TO COUNT OF BITS IN NUMBER
 9                                          ;       R4 - POINTER TO NUMBER (LOW WORD)
10                                          ;OUTPUTS:
11                                          ;       NUMBER IS PRINTED. LEADING ZEROS ARE PRINTED EXCEPT FOR
12                                          ;           DECIMAL NUMBERS (LEFT JUSTIFIED).
13                                          ;       R0 - CONTENTS DESTROYED
14
15 102442 010100            PNTNUM: MOV    R1,R0                   ;SAVE RADIX
16 102444 004737 102364             JSR    PC,GETCNT               ;GET COUNT OF BITS
17 102450                   PNTNUS:
   102450 010246                    MOV    R2,-(SP)                ;;PUSH R2 ON STACK
   102452 010346                    MOV    R3,-(SP)                ;;PUSH R3 ON STACK
   102454 010546                    MOV    R5,-(SP)                ;;PUSH R5 ON STACK
18 102456 012403                    MOV    (R4)+,R3                ;GET ONE PARAMETER WORD
19 102460 005005                    CLR    R5                      ;CLEAR STORAGE FOR OTHER
20 102462 020127 000020             CMP    R1,#16.                 ;MORE THAN 16 BITS IN NUMBER?
21 102466 003401                    BLE    1$
22 102470 012405                    MOV    (R4)+,R5                ;YES, GET SECOND PARAMETER WORD
23 102472                   1$:
   102472 010446                    MOV    R4,-(SP)                ;;PUSH R4 ON STACK
24 102474 010504                    MOV    R5,R4                   ;PUT HIGH WORD IN R4
25 102476 012702 000020             MOV    #16.,R2                 ;COMPUTE BITS NOT WANTED
26 102502 160102                    SUB    R1,R2                   ;BY SUBTRACTING BITS TO USE
27 102504 002002                    BGE    2$                      ;FROM 16.
28 102506 062702 000020             ADD    #16.,R2                 ;IF NEGATIVE, ADD 16 FOR FIRST WORD
29 102512 001414            2$:     BEQ    6$                      ;IF ZERO, NO BITS NEED BE CLEARED
30 102514 012705 100000             MOV    #BIT15,R5               ;START MASK WITH SIGN BIT SET
31 102520 005302            3$:     DEC    R2                      ;COUNT BITS IN MASK
32 102522 001402                    BEQ    4$
33 102524 006205                    ASR    R5                      ;SHIFT MORE BITS TO RIGHT
34 102526 000774                    BR     3$
35 102530 020127 000020     4$:     CMP    R1,#16.                 ;MORE THAN 16 BITS IN NUMBER?
36 102534 003402                    BLE    5$
37 102536 040504                    BIC    R5,R4                   ;YES, CLEAR IN HIGH WORD
38 102540 000401                    BR     6$
39 102542 040503            5$:     BIC    R5,R3                   ;NO, CLEAR IN LOW WORD
40 102544 004737 102704     6$:     JSR    PC,DIVIDE               ;DIVIDE BY RADIX IN R0
41 102550 010546                    MOV    R5,-(SP)                ;;PUSH R5 ON STACK
42 102552 005202                    INC    R2                      ;COUNT DIGITS ON STACK
43 102554 005703                    TST    R3                      ;CHECK IF QUOTIENT IS ZERO
44 102556 001372                    BNE    6$
45 102560 005704                    TST    R4
46 102562 001370                    BNE    6$
47 102564 020027 000012             CMP    R0,#10.                 ;IF RADIX IS DECIMAL
48 102570 001423                    BEQ    10$                     ; JUST GO PRINT DIGITS ON STACK
49 102572 010103                    MOV    R1,R3                   ;OTHERWISE COMPUTE NUMBER OF LEADING ZEROS
50 102574 162700 000014             SUB    #12.,R0                 ;DIVIDEND IS BITS IN NUMBER
51 102600 003002                    BGT    7$                      ;DIVISOR IS BITS PER DIGIT PRINTED
52 102602 012700 000003             MOV    #3,R0                   ;   (3 OR 4)
53 102606 004737 102704     7$:     JSR    PC,DIVIDE
```

I16

```
54 102612 005705                        TST     R5              ;IF REMAINDER NOT ZERO
55 102614 001401                        BEQ     8$              ;INCREMENT QUOTIENT
56 102616 005203                        INC     R3
57 102620 160203              8$:        SUB     R2,R3           ;SUBTRACT DIGITS ON STACK
58 102622 001406                        BEQ     10$             ;NO LEADING ZEROS IF ZERO
59 102624                     9$:
   102624 112700 000060                  MOVB    0'0,R0          ;STORE 0'0 IN R0 AND
   102630 004737 075506                  JSR     PC,PRINTC       ;PRINT THE CHARACTER.
60 102634 005303                        DEC     R3
61 102636 001372                        BNE     9$              ;REPEAT UNTIL COUNT REACHES ZERO
62 102640                     10$:
   102640 012605                         MOV     (SP)+,R5        ;;POP STACK INTO R>
63 102642 062705 000060                  ADD     0'0,R5          ;CNVERT TO ASCII DIGIT
64 102646 020527 000071                  CMP     R5,0'9          ;IF GREATER THAN A 9
65 102652 003402                         BLE     11$             ; CONVERT TO A OR HIGHER
66 102654 062705 000007                  ADD     0<'A-'9-1>,R5   ; FOR HEX DIGIT
67 102660                     11$:
   102660 110500                         MOVB    R5,R0           ;STORE R5 IN R0 AND
   102662 004737 075506                  JSR     PC,PRINTC       ;PRINT THE CHARACTER.
68 102666 005302                        DEC     R2              ;REPEAT FOR ALL DIGITS
69 102670 001363                        BNE     10$             ; ON STACK
70 102672 012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
   102674 012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
   102676 012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
   102700 012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
71 102702 000207                        RTS     PC
```

```
 1
 2                                    ;DIVIDE
 3                                    ;
 4                                    ;DIVIDE A 32 BIT UNSIGNED NUMBER BY A 16 BIT UNSIGNED NUMBER.
 5                                    ;REPLACE DIVIDEND WITH QUOTIENT AND RETURN REMAINDER.
 6                                    ;WILL NOT CHECK FOR DIVIDE B: ZERO.
 7                                    ;
 8                                    ;INPUTS:
 9                                    ;      R3    LOW 16 BITS OF DIVIDEND
10                                    ;      R4 -  HIGH 16 BITS OF DIVIDEND
11                                    ;      RO    DIVISOR
12                                    ;OUTPUTS:
13                                    ;      R3 -  LOW 16 BITS OF QUOTIENT
14                                    ;      R4 -  HIGH 16 BITS OF QUOTIENT
15                                    ;      R5 -  REMAINDER
16
17 102704                            DIVIDE:
   102704   010246                            MOV    R2,-(SP)          ;;PUSH R2 ON STACK
18 102706   012702   000040                   MOV    #32.,R2           ;SET UP SHIFT COUNT
19 102712   ^05005                            CLR    R5                ;START WITH ZERO REMAINDER
20 102714   C06303                    1$:     ASL    R3                ;SHIFT LEFT INTO R5
21 102716   006104                            ROL    R4
22 102720   006105                            ROL    R5
23 102722   020005                            CMP    RO,R5             ;WILL DIVISOR GO INTO REMAINDER
24 102724   101002                            BHI    2$                ;ONLY SUBTRACT IF IT WILL
25 102726   160005                            SUB    RO,R5             ;SUBTRACT DIVISOR
26 102730   005203                            INC    R3                ;PUT A ONE INTO QUOTIENT
27 102732   005302                    2$:     DEC    R2                ;COUNT THE SHIFTS
28 102734   001367                            BNE    1$
29 102736   012602                            MOV    (SP)+,R2          ;;POP STACK INTO R2
30 102740   000207                            RTS    PC
```

```
  1
  2                                  ;DIV10
  3                                  ;
  4                                  ;DIVIDE A 64 BIT UNSIGNED NUMBER BY A 10.
  5                                  ;REPLACE DIVIDEND WITH QUOTIENT AND RETURN REMAINDER.
  6                                  ;WILL NOT CHECK FOR DIVIDE BY ZERO.
  7                                  ;
  8                                  ;INPUTS:
  9                                  ;      R1 - LOW  16 BITS OF DIVIDEND
 10                                  ;      R2 - NEXT 16 BITS OF DIVIDEND
 11                                  ;      R3 - NEXT 16 BITS OF DIVIDEND
 12                                  ;      R4 - HIGH 16 BITS OF DIVIDEND
 13                                  ;OUTPUTS:
 14                                  ;      R1 - QUOTIENT.
 15                                  ;      R2 - QUOTIENT.
 16                                  ;      R3 - QUOTIENT.
 17                                  ;      R4 - QUOTIENT.
 18                                  ;      R5   REMAINDER
 19
 20 102742                  DIV10:
    102742  010046                   MOV    R0,-(SP)     ;;PUSH R0 ON STACK
 21 102744  012700  000100           MOV    #64.,R0      ;SET UP SHIFT COUNT
 22 102750  005005                   CLR    R5           ;START WITH ZERO REMAINDER
 23 102752  006301           1$:     ASL    R1
 24 102754  006102                   ROL    R2           ;SHIFT LEFT INTO R5
 25 102756  006103                   ROL    R3
 26 102760  006104                   ROL    R4
 27 102762  006105                   ROL    R5
 28 102764  022705  000012           CMP    #10.,R5      ;SILL DIVISOR GO INTO REMAINDER?
 29 102770  101003                   BHI    2$           ;ONLY SUBTRACT IF IT WILL
 30 102772  162705  000012           SUB    #10.,R5      ;SUBTRACT DIVISOR
 31 102776  005201                   INC    R1           ;PUT A ONE INTO QUOTIENT
 32 103000  005300           2$:     DEC    R0           ;COUNT THE SHIFTS
 33 103002  001363                   BNE    1$
 34 103004  012600                   MOV    (SP)+,R0     ;;POP STACK INTO R0
 35 103006  000207                   RTS    PC           ;  <R4,R3,R2,R1> AND REMAINDER IN R5
```

L16

```
                                          ;PRINT HEX·NUMBERS WITH LEADING SPACE
 2
 3
 4  103010                                T2PNTW:
    103010   112700   000040                MOVB    #' ,RO           ;STORE #   IN RO AND
    103014   004737   075506                JSR     PC,PRINTC        ;PRINT THE CHARACTER.
 5  103020   010146                         MOV     R1, (SP)         ;;PUSH R1 ON STACK
 6  103022   000301                         SWAB R1
 7  103024   004737   103050                CALL    T2PNT            ;PRINT HIGH TWO DIGITS
 8  103030   012601                         MOV     (SP)+,R1         ;;POP STACK INTO R1
 9  103032   004737   103050                CALL    T2PNT            ;PRINT LOW TWO DIGITS
10  103036   000207                         RETURN
11
12  103040                                T2PNTB:
    103040   112700   000040                MOVB    #' ,RO           ;STORE #'   IN RO AND
    103044   004737   075506                JSR     PC,PRINTC        ;PRINT THE CHARACTER.
13
14                                        ;PRINT TWO HEX DIGITS FROM NUMBER IN R1
15
16  103050                                T2PNT:
    103050   010146                         MOV     R1,-(SP)         ;;PUSH R1 ON STACK
17  103052   006001                         ROR     R1                    ;SHIFT TO GET HIGH DIGIT
18  103054   006001                         ROR     R1
19  103056   006001                         ROR     R1
20  103060   006001                         ROR     R1
21  103062   004737   103070                CALL    T2PNTO           ;PRINT TWO DIGITS
22  103066   012601                         MOV     (SP)+,R1         ;;POP STACK INTO R1
23  103070   042701   177760       T2PNTO:  BIC     #+C17,R1              ;CLEAR OTHER BITS
24  103074   062701   000060                ADD     # 0,R1           ;CONVERT TO ASCII CHARACTER
25  103100   020127   000071                CMP     R1,#'9           ;IF GREATER THAN A 9
26  103104   003402                         BLE T2PNTD               ;  CONVERT TO A OR HIGHER
27  103106   062701   000007                ADD     #<'A-'9-1>,R1    ;  FOR HEX DIGIT
28  103112                                T2PNTD:
    103112   110100                         MOVB    R1,RO            ;STORE R1 IN RO AND
    103114   004737   075506                JSR     PC,PRINTC        ;PRINT THE CHARACTER.
29  103120   000207                         RETURN
```

M16

```
  1                                    ;T2GNUM
  2                                    ;
  3                                    ;
  4                                    ;GET A HEX DIGIT FROM AN ASCII INPUT STRING
  5                                    ;
  6                                    ;INPUTS:
  7                                    ;       R1 - STRING POINTER
  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
 13 103122   005000          T2GNUM:  CLR     R0                      ;CLEAR DIGIT COUNT
 14 103124   105711                   TSTB    (R1)                    ;CHECK IF END OF LINE
 15 103126   001442                   BEQ     T2GNX                   ;REPORT NULL CHARACTER FOUND
 16 103130   121127   000040          CMPB    (R1),#'                 ;CHECK IF A SPACE
 17 103134   001002                   BNE     T2GND1                  ;IF SO, IGNORE IT
 18 103136   005201                   INC     R1
 19 103140   000770                   BR      T2GNUM
 20 103142   005004          T2GND1:  CLR     R4                      ;CLEAR NUMBER STORAGE
 21 103144                   T2GND2:
    103144   010246                   MOV     R2,-(SP)                ;;PUSH R2 ON STACK
 22 103146   112102                   MOVB    (R1)+,R2                ;GET CHARACTER
 23 103150   162702   000060          SUB     #'0,R2                  ;CONVERT TO HEX DIGIT
 24 103154   100431                   BMI     T2GNE
 25 103156   020227   000011          CMP     R2,#9.
 26 103162   003410                   BLE T2GND3
 27 103164   020227   000021          CMP     R2,#<'A-'0>
 28 103170   103423                   BLO T2GNE
 29 103172   020227   000026          CMP     R2,#<'F-'0>
 30 103176   101020                   BHI T2GNE
 31 103200   162702   000007          SUB     #<'A-'9-1>,R2
 32 103204   006304          T2GND3:  ASL     R4
 33 103206   006304                   ASL     R4
 34 103210   006304                   ASL     R4
 35 103212   006304                   ASL     R4
 36 103214   050204                   BIS     R2,R4
 37 103216   005200                   INC     R0
 38 103220   012602                   MOV     (SP)+,R2                ;;POP STACK INTO R2
 39 103222   105711                   TSTB    (R1)
 40 103224   001403                   BEQ     T2GNX
 41 103226   121127   000040          CMPB    (R1),#'
 42 103232   001344                   BNE     T2GND2
 43 103234   005700          T2GNX:   TST     R0
 44 103236   000207                   RETURN
 45
 46 103240                   T2GNE:
    103240   012602                   MOV     (SP)+,R2                ;;POP STACK INTO R2
    103242   012600                   MOV     (SP)+,R0                ;;POP STACK INTO R0
 47 103244   000137   100340          JMP     T2CMDE
```

```
  1
  3                                 ;PNTERR
  4                                 ;
  5                                 ;PRINT ERROR MESSAGE FROM DM PROGRAM REQUEST 11 OR 12.
  6                                 ;
  7                                 ;INPUTS:
  8                                 ;       R5    CONTROLLER TABLE ADDRESS
  9                                 ;       R4    MESSAGE DATA ADDRESS
 10                                 ;       R3 -  COMMAND DATA ADDRESS
 11                                 ;OUTPUTS:
 12                                 ;       ERROR MESSAGE PRINTED
 13                                 ;       BIT     15 SET IN COMMAND DATA IF DRIVE HAS BEEN DROPPED
 14
 15 103250                         PNTERR:
    103250  010046                         MOV    R0,-(SP)            ;;PUSH R0 ON STACK
    103252  010146                         MOV    R1,-(SP)            ;;PUSH R1 ON STACK
    103254  010246                         MOV    R2,-(SP)            ;;PUSH R2 ON STACK
 16 103256  005764  000004                 TST    4(R4)               ;GET DRIVE NUMBER
 17 103262  002004                         BGE    1$                  ;CHECK IF BIT 15 SET
 18 103264  116537  000002  002074         MOVB   C.UNIT(R5),L$LUN    ;IF SO, GET UNIT FROM CONTROLLER TABLE
 19 103272  000416                         BR     2$
 20 103274                         1$:
    103274  010446                         MOV    R4,-(SP)            ;;PUSH R4 ON STACK
 21 103276  016401  000004                 MOV    4(R4),R1            ;GET DRIVE NUMBER
 22 103302  004737  102274                 CALL   GTDRVT              ;GET DRIVE TABLE ADDRESS
 23 103306  001036                         BNE    5$                  ;IF UNIT DROPPED, EXIT
 24 103310  005764  000002                 TST    D.UNIT(R4)          ;SEE IF UNIT HAS BEEN DROPPED FROM TESTING
 25 103314  100004                         BPL    3$                  ;  PROCEED IF STILL TO BE TESTED
 26 103316  052713  100000                 BIS    #BIT15,(R3)         ;TELL DM PROGRAM TO STOP TESTING THIS UNIT
 32 103322  012604                         MOV    (SP)+,R4            ;;POP STACK INTO R4
 33 103324  000423                         BR     4$
 34 103326                         3$:
    103326  012604                         MOV    (SP)+,R4            ;;POP STACK INTO R4
 35 103330  012702  064402         2$:     MOV    #ERRTYP,R2          ;GET POINTER TO ERROR TABLE
 36 103334  016412  000002                 MOV    2(R4),(R2)          ;GET ERROR TYPE
 37 103340  006112                         ROL    (R2)
 38 103342  006112                         ROL    (R2)
 39 103344  006112                         ROL    (R2)
 40 103346  042722  177774                 BIC    #^C3,(R2)+          ;CLEAR LOW 2 BITS
 41 103352  016412  000002                 MOV    2(R4),(R2)
 42 103356  042722  140000                 BIC    #140000,(R2)+       ;MASK LOW 14 BITS
 43 103362  005022                         CLR    (R2)+               ;CLEAR MESSAGE POINTER
 44 103364  012712  075250                 MOV    #ERR.TN,(R2)        ;GET ROUTINE NUMBER
 45 103370  104460                         TRAP   C$ERROR
 46 103372  000241                         CLC                        ;DRIVE HAS NOT BEEN DROPPED
 47 103374                         4$:
    103374  012602                         MOV    (SP)+,R2            ;;POP STACK INTO R2
    103376  012601                         MOV    (SP)+,R1            ;;POP STACK INTO R1
    103400  012600                         MOV    (SP)+,R0            ;;POP STACK INTO R0
 48 103402  000207                         RETURN
 49 103404  000261         5$:             SEC                        ;DRIVE HAS BEEN DROPPED
 50 103406  000772                         BR     4$
```

```
      1                                         ;LOADDM
      2                                         ;
      3                                         ;
      4                                         ;LOAD AND START A DM PROGRAM INTO A CONTROLLER
      5                                         ;
      6                                         ;INPUTS:
      7                                         ;       R5      CONTROLLER TABLE ADDRESS
      8                                         ;IMPLICIT INPUTS:
      9                                         ;       DMPROG   POINTER TO START OF DM PROGRAM IN MEMORY
     10                                         ;OUTPUTS:
     11                                         ;       IF LOAD SUCCEEDS    Z CLEAR
     12                                         ;                          CONTROLLER TABLE MARKED LOADED
     13                                         ;       IF ERROR - Z SET
     14
     15  103410                       LOADDM:
     32  103410   016504  000004      2$:       MOV     C.VEC(R5),R4          ;GET VECTOR OF UDA
     33  103414   042704  177000                BIC     #^C<CT.VEC>,R4
     34  103420   010501                        MOV     R5,R1                ;GET INTERRUPT SERVICE LINK
     35  103422   062701  000010                ADD     #C.JSR,R1
     36  103426   012746  000340                MOV     #PRIO7,(SP)
         103432   010146                        MOV     R1,-(SP)
         103434   010446                        MOV     R4,-(SP)
         103436   012746  000003                MOV     #3,-(SP)
         103442   104437                        TRAP    C$SVEC
         103444   062706  000010                ADD     #10,SP
     37
     38  103450   006204                        ASR     R4                   ;INITIALIZE UDA WITH SMALLEST
     39  103452   006204                        ASR     R4                   ;POSITION VECTOR FOR UDA
     40  103454   004737  104612                CALL    UDAINT               ; RING BUFFER AND INTERRUPTS ENABLED
     41  103460   001002                        BNE     3$                   ; BRANCH IF NO ERROR
     42  103462   000137  104120                JMP     LOADER               ; ELSE, JUMP IF AN ERROR
     43
     44
     45                                         . NOW CHECK IF THE CONTROLLER IS A UDA50-A
     46  103466   013703  105406      3$:       MOV     SSTEP4,R3            ; GET SAVED VALUE OF UDA INIT STEP 4
     47  103472   010301                        MOV     R3,R1                ; R3 HAS STEP 4 INFO
     48  103474   042701  177760                BIC     #^C<SA.MCV>,R1       ; R1 = MICRO CODE LEVEL
     49  103500   042703  177417                BIC     #^C<SA.CNT>,R3       ; R3 = CNT MODE
     50  103504   006003                        ROR     R3
     51  103506   006003                        ROR     R3
     52  103510   006003                        ROR     R3
     53  103512   006003                        ROR     R3
     54  103514   032703  000017                BIT     #<SA.CNT/16.>,R3     ; CHECK WITH CONTROLLER MODEL
     55  103520   001010                        BNE     4$                   ; IF ZERO, UDA50(M7161)//IF NOT ZERO UDA50 A
     56  103522   052765  100000  000002        BIS     #BIT15,C.UNIT(R5)    ; AND MARK AS DO NOT EXECUTE (M 161)
     57  103530   104455                        TRAP    C$ERDF
         103532   000016                        .WORD   14
         103534   000000                        .WORD   0
         103536   074362                        .WORD   ERRO14
     58  103540   000567                        BR      LOADER               ; EXIT
     59  103542   020127  000003      4$:       CMP     R1,#3                ; IS MICROCODE VERSION UP TO DATE?
     60  103546   103004                        BCC     5$                   ; IF SO, BRANCH
     61  103550   104455                        TRAP    C$ERDF
         103552   000016                        .WORD   14
         103554   000000                        .WORD   0
         103556   074362                        .WORD   ERRO14
     62  103560   004737  075446      5$:       CALL    HCOMM                ;ALLOCATE SPACE FOR HOST COMM AREA
```

D 1

```
64 103564  023727  064444  000001              CMP    TNUM,#1                      ;IF TEST NUMBER 1
65 103572  001440                              BEQ    LOADT1                       ;  DO SPECIAL LOAD
67 103574  017701  160632                      MOV    @DMPROG,R1                   ;GET SIZE OF PROGRAM
68 103600  012700  000002          LOADB:      MOV    #OP.ESP,R0                   ;BUILD EXECUTE SUPPLIED PROGRAM COMMAND PACKET
69 103604  004737  104124                      CALL   BLDCMD
70 103610  013764  064432  000040              MOV    DMPROG,HC.CPK+P.UADR(R4)     ;LOAD MAIN PROGRAM ADDRESS
71 103616  010164  000034                      MOV    R1,HC.CPK+P.BCNT(R4)         ;  AND SIZE
72 103622  013764  064432  000054              MOV    DMPROG,HC.CPK+P.OVRL(R4)     ;LOAD OVERLAY ADDRESS
73 103630  067764  160576  000054              ADD    @DMPROG,HC.CPK+P.OVRL(R4)
82 103636  004737  104210                      CALL   SNDCMD                       ;SEND COMMAND TO UDA
83 103642  004737  104320                      CALL   WAITMS                       ;WAIT FOR MESSAGE RESPONSE
84 103646  032764  000037  000032              BIT    #ST.MSK,HC.MPK+P.STS(R4)     ;CHECK FOR ERRORS
85 103654  001115                              BNE    LOADE1
86 103656  042765  000024  000014              BIC    #CT.CMD+CT.REQ,C.FLG(R5)     ;CLEAR COMMAND OUTSTANDING FLAG
87 103664  052765  000002  000014              BIS    #CT.RN,C.FLG(R5)             ;SET DM PROGRAM RUNNING FLAG
91 103672  000207                              RETURN
```

```
 1
 2                                  ;LOAD DM PROGRAM FROM MEMORY SPACE TESTED DURING
 3                                  ;INITIALIZATION IN TEST 1
 4
 5 103674  017704  160532  LOADT1:  MOV    @DMPROG,R4      ;GET SIZE OF DM PROGRAM IN BYTES
 6 103700  162704  000040           SUB    #DMMAIN,R4
 7 103704  013700  064432           MOV    DMPROG,R0       ;GET ADDRESS OF DM PROGRAM
 8 103710  062700  000040           ADD    #DMMAIN,R0
 9 103714  005001                    CLR    R1             ;START WITH OFFSET OF ZERO
10
11 103716  012703  000214  LT1L1:   MOV    #<HC.BSZ*2>,R3  ;GET SIZE OF BOTH BUFFERS
12 103722  020403                    CMP    R4,R3          ;IF FEWER BYTES REMAINING IN PROGRAM
13 :03724  103001                    BHIS   LT11
14 103726  010403                    MOV    R4,R3          ;USE ACTUAL BYTE COUNT
15 103730                   LT11:
   103730  010346                    MOV    R3,-(SP)       ;;PUSH R3 ON STACK
16 103732  013702  064412           MOV    FFREE,R2       ;GET ADDRESS OF BUFFER
17 103736  162702  000214           SUB    #<HC.BSZ*2>,R2
18 103742  010246                    MOV    R2,-(SP)       ;;PUSH R2 ON STACK
19 103744  012022           LT1L2:   MOV    (R0)+,(R2)+    ;MOVE DATA TO BUFFER
20 103746  162703  000002           SUB    #2,R3          ;COUNT BYTES
21 103752  001374                    BNE    LT1L2
22 103754  012602                    MOV    (SP)+,R2       ;;POP STACK INTO R2
23 103756  012603                    MOV    (SP)+,R3       ;;POP STACK INTO R3
24 103760  004737  104006           CALL   LOAD           ;LOAD INTO UDA
25 103764  001455                    BEQ    LOADER         ;IF ERROR, GET OUT NOW
26 103766  006203                    ASR    R3             ;CONVERT BYTES TO WORDS
27 103770  060301                    ADD    R3,R1          ;INCREASE OFFSET FOR NEXT BUFFER
28 103772  006303                    ASL    R3             ;CONVERT WORDS TO BYTES
29 103774  160304                    SUB    R3,R4          ;REDUCE REMAINING BYTE COUNT
30 103776  001347                    BNE    LT1L1          ;GET NEXT BUFFER
31 104000  012701  000040           MOV    #DMMAIN,R1     ;GET A BYTE COUNT OF HEADER ONLY
32 104004  000675                    BR     LOADB          ;NOW START
```

F1

```
 1                                          ;LOAD
 2                                          ;
 3                                          ;ISSUE DOWNLINE LOAD COMMAND TO UDA. CHECK THAT LOAD
 4                                          ;HAPPENS WITHOUT ERROR.
 5                                          ;
 6                                          ;
 7                                          ;INPUTS:
 8                                          ;        R1 - OFFSET FOR DM PROGRAM
 9                                          ;        R2 - ADDRESS OF BUFFER CONTAINING PROGRAM
10                                          ;        R3 - SIZE OF BUFFER IN BYTES
11                                          ;        R5 - CONTROLLER TABLE ADDRESS
12                                          ;OUTPUTS:
13                                          ;        Z CLEAR IF NO ERROR
14                                          ;        Z SET IF ERROR AND ERROR REPORTED
15
16 104006                                  LOAD:
   104006   010046                            MOV    R0,-(SP)                   ;;PUSH R0 ON STACK
   104010   010346                            MOV    R3,-(SP)                   ;;PUSH R3 ON STACK
   104012   010446                            MOV    R4,-(SP)                   ;;PUSH R4 ON STACK
17 104014   012700   000031                   MOV    #OP.MWR,R0                 ;GET DOWNLINE LOAD COMMAND
18 104020   004737   104124                   CALL   BLDCMD                     ;BUILD COMMAND PACKET
19 104024   010264   000040                   MOV    R2,HC.CPK+P.UADR(R4)       ;STUFF IN BUFFER ADDRESS
20 104030   010364   000034                   MOV    R3,HC.CPK+P.BCNT(R4)       ;STUFF IN BYTE COUNT
21 104034   010164   000060                   MOV    R1,HC.CPK+P.RGOF(R4)       ;STUFF IN OFFSET
22 104040   012764   000001   000054          MOV    #1,HC.CPK+P.RGID(R4)       ;STUFF IN REGION ID 1
23 104046   004737   104210                   CALL   SNDCMD                     ;SEND COMMAND TO UDA
24 104052   004737   104320                   CALL   WAITMS                     ;WAIT FOR MESSAGE RESPONSE
25 104056   001420                            BEQ    LOADER                     ; IF FAILED, EXIT
26 104060   032764   000037   000032          BIT    #ST.MSK,HC.MPK+P.STS(R4)   ;LOOK FOR ANY ERROR
27 104066   001010                            BNE    LOADE1
28 104070   042765   000004   000014          BIC    #CT.CMD,C.FLG(R5)          ;CLEAR COMMAND ISSUED
29 104076   012604                            MOV    (SP)+,R4                   ;;POP STACK INTO R4
   104100   012603                            MOV    (SP)+,R3                   ;;POP STACK INTO R3
   104102   012600                            MOV    (SP)+,+                    ;;POP STACK INTO R0
30 104104   000244                            CLZ                              ;CLEAR Z TO INDICATE NO ERROR
31 104106   000207                            RETURN
```

G1

```
   1
   2                                      ;UDA FAILED TO DOWNLINE LOAD DM PROGRAM
   3
   4 104110                      LOADE1:
     104110   104455                      TRAP      C$ERDF
     104112   000042                      .WORD     34
     104114   000000                      .WORD     0
     104116   074772                      .WORD     ERRO34
   5 104120   000264             LOADER:  SEZ                            ;SET Z TO INDICATE ERROR OCCURRED
   6 104122   000207                      RETURN
```

```
 1                                   ;BLDCMD
 2
 3                                   ;
 4                                   ;BUILD A COMMAND IN COMMAND PACKET
 5                                   ;
 6                                   ;INPUTS:
 7                                   ;       R5 - CONTROLLER TABLE ADDRESS
 8                                   ;       R0 - COMMAND CODE
 9                                   ;OUTPUTS:
10                                   ;       R4 - ADDRESS OF HOST COMM AREA
11                                   ;       COMMAND PACKET CONTAINING REF NUMBER AND OPCODE. ALL OTHER FIELDS CLEARED.
12                                   ;       CMD REFERRENCE NUMBER IN CONTROLLER TABLE INCREMENTED AND RESULT
13                                   ;       IN COMMAND PACKET.
14                                   ;       R0 - CONTENTS DESTROYED
15
16 104124                  BLDCMD:
   104124  010146                  MOV     R1,-(SP)                ;;PUSH R1 ON STACK
   104126  010046                  MOV     R0,-(SP)                ;;PUSH R0 ON STACK
17 104130  016504  000016          MOV     C.HCOM(R5),R4           ;GET ADDRESS OF HOST COMM AREA
18 104134  010400                  MOV     R4,R0                   ;COPY TO R0
19 104136  062700  000014          ADD     #HC.CEV,R0              ;COMPUTE ADDRESS OF COMMAND ENVELOPE
20 104142  012720  000060          MOV     #HC.PSZ,(R0)+           ;LOAD PACKET LENGTH
21 104146  012701  001000          MOV     #DUP,R1                 ;LOAD DIAG CIRCUIT IDENTIFIER
22 104152  022716  000031          CMP     #OP.MWR,(SP)            ;IF CODE IS MAINTENANCE WRITE
23 104156  001002                  BNE     BLDCO                   ;   GET OTHER CIRCUIT IDENTIFIER
24 104160  012701  177777          MOV     #DIAG,R1
25 104164  010120          BLDCO:  MOV     R1,(R0)+                ;PUT IDENTIFIER INTO PACKET
26 104166  012701  000030          MOV     #<HC.PSZ>/2,R1          ;GET WORDS TO CLEAR
27 104172  005020          BLDC1:  CLR     (R0)+                   ;CLEAR PACKET
28 104174  005301                  DEC     R1
29 104176  001375                  BNE     BLDC1
30 104200  012664  000030          MOV     (SP)+,HC.CPK+P.OPCD(R4)         ;;POP STACK INTO HC.CPK+P.OPCD(R4)
31 104204  012601                  MOV     (SP)+,R1                ;;POP STACK INTO R1
32 104206  000207                  RETURN
```

```
 1
 2                                          ;SNDCMD
 3                                          ;
 4                                          ;SEND A COMMAND TO THE UDA.
 5                                          ;CLEAR THE RESPONSE PACKET. MARK BOTH PACKETS AVAILABLE TO THE
 6                                          ;UDA. SET COMMAND ISSUED BIT IN CONTROLLER TABLE AND INITIALIZE
 7                                          ;TIMEOUT COUNTER.
 8                                          ;
 9                                          ;INPUTS:
10                                          ;     R5 - CONTROLLER TABLE ADDRESS
11                                          ;OUTPUTS:
12                                          ;     R4 - ADDRESS OF HOST COMM AREA
13
14
15 104210  016504  000016          SNDCMD: MOV    C.HCOM(R5),R4             ;LOAD R4 WITH HOST COMM AREA ADDRESS
16 104214  005265  000044                  INC    C.REF(R5)                ;INCREMENT CMD REFERENCE NUMBER
17 104220  016564  000044  000020          MOV    C.REF(R5),HC.CPK+P.CRF(R4) ;PUT IN PACKET
18 104226  012764  140000  000006          MOV    #RG.CWN+RG.FLG,HC.MCT(R4)  ;MARK MESSAGE PACKET AVAILABLE
19 104234  012764  100000  000012          MOV    #RG.OWN,HC.CCT(R4)       ;MARK COMMAND TO UDA
20 104242  005775  000000                  TST    @(R5)                    ;TELL UDA COMMAND IS THERE
21 104246  052765  000004  000014          BIS    #CT.CMD,C.FLG(R5)        ;MARK COMMAND ISSUED
22 104254  000207                          RETURN
```

J1

```
 1
 2                                      ;CLRBUF
 3                                      ;
 4                                      ;CLEAR THE SPECIFIED DATA BUFFER IN THE HOST COMM AREA
 5                                      ;AND LOAD BUFFER DESCRIPTOR IN COMMAND PACKET TO THE BUFFER
 6                                      ;
 7                                      ;INPUTS:
 8                                      ;       R5 - CONTROLLER TABLE ADDRESS
 9                                      ;       R4 - ADDRESS OF HOST COMM AREA
10                                      ;       RO - OFFSET INTO HOST COMM AREA TO DATA BUFFER
11                                      ;OUTPUTS:
12                                      ;       DATA BUFFER CLEARED
13                                      ;       COMMAND PACKET POINTING TO BUFFER
14                                      ;       BYTE COUNT SET TO SIZE OF BUFFER
15                                      ;       R4   ADDRESS OF DATA BUFFER
16
17 104256                              CLRBUF:
   104256  010046                              MOV     RO,-(SP)                    ;;PUSH RO ON STACK
   104260  010146                              MOV     R1,-(SP)                    ;;PUSH R1 ON STACK
18 104262  060400                              ADD     R4,RO                       ;ADD START OF HOST COMM AREA TO OFFSET
19 104264  010064  000040                      MOV     RO,HC.CPK+P.UADR(R4)        ;PUT BUFFER ADDRESS IN COMMAND PACKET
20 104270  012764  000106  000034              MOV     #HC.BSZ,HC.CPK+P.BCNT(R4) ;PUT SIZE OF BUFFER IN COMMAND PACKET
21 104276  010004                              MOV     RO,R4                       ;PUT BUFFER ADDRESS IN R4
22 104300  012701  000043                      MOV     #<HC.BSZ>/2,R1              ;GET SIZE OF BUFFER IN WORDS
23 104304  005020                      CLRBFL: CLR     (RO)+                       ;CLEAR ALL THE WORDS
24 104306  005301                              DEC     R1
25 104310  001375                              BNE     CLRBFL
26 104312  012601                              MOV     (SP)+,R1                    ;;POP STACK INTO R1
   104314  012600                              MOV     (SP)+,RO                    ;;POP STACK INTO RO
27 104316  000207                              RETURN
```

K1

```
  1
  2                                     ;WAITMS
  3                                     ;
  4                                     ;WAIT FOR UDA TO RESPOND WITH A MESSAGE PACKET
  5                                     ;
  6                                     ;INPUTS:
  7                                     ;       R5 - ADDRESS OF CONTROLLER TABLE
  8                                     ;OUTPUTS:
  9                                     ;       Z CLEAR IF NO ERROR
 10                                     ;       Z SET IF ERROR, MESSAGE PRINTED
 11
 12 104320                     WAITMS:
    104320  010046                      MOV     RO,-(SP)              ;;PUSH RO ON STACK
    104322  010146                      MOV     R1, (SP)              ;;PUSH R1 ON STACK
 13 104324  012700  000036              MOV     @30.,RO               ;SET TIME OUT VALUE OF 30 SECONDS
 14 104330  010501                      MOV     R5,R1                 ;POINT TO TIME OUT COUNTER
 15 104332  062701  000040              ADD     #C.TO,R1
 16 104336  004737  104530              CALL    SETTO
 17 104342  011500              1$:     MOV     (R5),RO               ;GET ADDRESS OF UDAIP REGISTER
 18 104344  032765  000010  000014      BIT     #CT.MSG,C.FLG(R5)     ;LOOK IF INTERRUPT OCCURRED
 19 104352  001030                      BNE     3$                    ;BRANCH IF SO
 20 104354  016001  000002              MOV     2(RO),R1              ;LOOK AT UDASA REGISTER
 21 104360  001034                      BNE     4$                    ;BRANCH IF ERROR CODE PRESENT
 32                                                                   ;>>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<<
 33 104362  104422                      TRAP    C$BRK
 34 104364  005737  064616              TST     KW.CSR                ;SEE IF A CLOCK ON SYSTEM
 35 104370  001764                      BEQ     1$
 36 104372  023765  064630  000042      CMP     KW.EL+2,C.TOH(R5)     ;CHECK IF TIMEOUT HAS HAPPENED
 37 104400  101005                      BHI     2$
 38 104402  001357                      BNE     1$
 39 104404  023765  064626  000040      CMP     KW.EL,C.TO(R5)
 40 104412  103753                      BLO     1$
 41 104414                      2$:
    104414  104455                      TRAP    C$ERDF
    104416  000044                      .WORD   36
    104420  000000                      .WORD   O
    104422  075016                      .WORD   ERRO36
 42 104424  012601                      MOV     (SP)+,R1              ;;POP STACK INTO R1
    104426  012600                      MOV     (SP)+,RO              ;;POP STACK INTO RO
 43 104430  000264                      SEZ
 44 104432  000207                      RETURN
 45
 46 104434  042765  000010  000014  3$: BIC     #CT.MSG,C.FLG(R5)     ;CLEAR MESSAGE RECEIVED FLAG
 47 104442  012601                      MOV     (SP)+,R1              ;;POP STACK INTO R1
    104444  012600                      MOV     (SP)+,RO              ;;POP STACK INTO RO
 48 104446  000244                      CLZ                           ;GIVE NO ERROR RETURN
 49 104450  000207                      RETURN
 50 104452                      4$:
    104452  104455                      TRAP    C$ERDF
    104454  000045                      .WORD   37
    104456  000000                      .WORD   O
    104460  075030                      .WORD   ERRO37
 51 104462  012601                      MOV     (SP)+,R1              ;;POP STACK INTO R1
    104464  012600                      MOV     (SP)+,RO              ;;POP STACK INTO RO
 52 104466  000264                      SEZ
 53 104470  000207                      RETURN
```

```
 1                                       ;APRINT
 2                                       ;
 3                                       ;
 4                                       ;CONVERT AN 18 BIT ADDRESS STORED IN TWO WORDS INTO A FORMAT
 5                                       ;THAT WILL ALLOW PRINTING OF THE 18 BIT NUMBER.
 6                                       ;
 7                                       ;INPUTS:
 8                                       ;        RO    ADDRESS OF TWO WORD BLOCK CONTAINING ADDRESS.
 9                                       ;              FIRST WORD CONTAINING LOW 16 BITS.
10                                       ;              SECOND WORD CONTAINING HIGH 2 BITS.
11                                       ;OUTPUTS:
12                                       ;        R1 - HIGH 3 BITS OF ADDRESS
13                                       ;        R2    LOW 15 BITS OF ADDRESS
14
15 104472 016001 000002   APRINT: MOV    2(RO),R1              ;GET HIGH 2 BITS
16 104476 006301                  ASL    R1                    ;SHIFT LEFT
17 104500 011002                  MOV    (RO),R2               ;GET LOW 16 BITS
18 104502 100001                  BPL    APRIZ                 ;IF 16TH BIT SET
19 104504 005201                  INC    R1                    ;PLACE IT IN WITH HIGH 2 BITS
20 104506 000207          APRIZ:  RETURN
```

M1

PRE PROGRAMMED SUBROUTINES

```
 1
 2                                      ;NXMI
 3                                      ;
 4                                      ;NON-EXISTANT MEMORY SERVICE ROUTINE
 5                                      ;
 6                                      ;INPUTS:
 7                                      ;        NXMAD SET TO ZERO
 8                                      ;OUTPUTS:
 9                                      ;        NXMAD SET TO ONES IF NON EXISTANT TRAP OCCURED
10
11 104510                      NXMI::
12 104510  012737  177777  064636              MOV     # 1,NXMAD
13 104516                      L10036:
   104516  000002                      RTI
```

N⊥

```
 1
 2                                    ;UDASRV
 3                                    ;
 4                                    ;UDA INTERRUPT SERVICE ROUTINE. MARKS UDA CONTROLLER TABLE THAT AN
 5                                    ;INTERRUPT HAS BEEN RECEIVED.
 6                                    ;
 7                                    ;THIS ROUTINE IS CALLED BY A [JSR RO,UDASRV] INSTRUCTION FROM WITHIN
 8                                    ;THE CONTROLLER TABLE. THE PC STORED IN RO IS THE ADDRESS OF THE C.FLG
 9                                    ;WORD IN THE CONTROLLER TABLE. THE STACK CONTAINS THE SAVED CONTENTS
10                                    ;OF RO FOLLOWED BY THE INTERRUPTED PC AND PS.
11                                    ;
12                                    ;INPUTS:
13                                    ;     RO   ADDRESS OF C.FLG WORD IN CONTROLLER TABLE
14                                    ;     STACK - SAVED CONTENTS OF RO
15                                    ;OUTPUTS:
16                                    ;     CT.CMD CLEARED AND CT.MSG SET IN C.FLG WORD OF CONTROLLER TABLE
17                                    ;     RO - RESTORED FROM STACK
18
19 104520                            UDASRV::
20 104520  052710  000010                    BIS     #CT.MSG,(RO)           ;SET CT.MSG
21 104524  012600                             MOV     (SP)+,RO               ;POP STACK INTO RO
22 104526                            L10037:
   104526  000002                             RTI
```

```
     1                                 ;SETTO
     2                                 ;
     3                                 ;SET TIMEOUT COUNTER TO SOME NUMBER OF SECONDS FROM CURRENT TIME.
     4                                 ;
     5                                 ;INPUTS:
     6                                 ;     RO - NUMBER OF SECONDS FOR TIMEOUT
    10                                 ;     R1 - ADDRESS WHERE TWO WORD TIME TO BE PUT
    11                                 ;OUTPUTS:
    12                                 ;     RO - CONTENTS DESTROYED
    13                                 ;     R1 - INCREMENTED BY 2
    14
    15
    16                                 ;COMPUTE CLOCK TICKS TIL TIMEOUT
    17
    28 104530                         SETTO:
       104530  010246                        MOV     R2, (SP)        ;;PUSH R2 ON STACK
       104532  010346                        MOV     R3, (SP)        ;;PUSH R3 ON STACK
    30 104534  005002                        CLR     R2              ;CLEAR PRODUCT
    31 104536  013703  064624                MOV     KW.HZ,R3        ;GET MULTIPLICAND
    46 104542  006200                 SETOO: ASR     RO              ;SHIFT MULTIPLIER TO RIGHT
    47 104544  103001                        BCC     SETO1           ;IF A ONE BIT SHIFTED OUT
    48 104546  060302                        ADD     R3.R2           ;   ADD MULTIPLICAND TO PRODUCT
    49 104550  006303                 SETO1: ASL     R3              ;DOUBLE THE MULTIPLICAND
    50 104552  005700                        TST     RO
    51 104554  001372                        BNE     SETOO           ;CONTINUE UNTIL MULTIPLIER IS ZERO
    53                                        ;GET CURRENT TIME
    54
    55 104556  013700  064626        SETO2:  MOV     KW.EL.RO        ;GET TIME
    56 104562  013703  064630                MOV     KW.EL+2,R3
    57 104566  020037  064626                CMP     RO,KW.EL        ;IF CHANGED DURING RETRIEVAL
    58 104572  001371                        BNE     SETO2           ;   GET IT AGAIN
    59
    60
    61                                        ;ADD TIME TIL TIMEOUT
    62 104574  060200                        ADD     R2,RO           ;ADD
    63 104576  005503                        ADC     R3
    67
    68                                        ;PUT RESULT IN STORAGE
    69
    70 104600  010021                        MOV     RO,(R1)+
    71 104602  010311                        MOV     R3,(R1)
    72
    76 104604  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
       104606  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
    78 104610  000207                        RETURN
```

C2

```
  1                                  ;UDAINT
  2                                  ;
  3                                  ;FUNCTIONAL DESCRIPTION:
  4                                  ;      SUBROUTINE TO INITIALIZE A UDA AND BRING IT ON LINE.
  5                                  ;      ALL STEPS ARE CHECKED. AN ERROR MESSAGE IS REPORTED IF ANY ERROR
  6                                  ;      DETECTED.
  7                                  ;
  8                                  ;
  9                                  ;INPUTS:
 10                                  ;      R5 - ADDRESS OF CONTROLLER TABLE.
 11                                  ;      R4 - LENGTH, INTERRUPT AND VECTOR FIELDS TO SEND TO UDA
 12                                  ;IMPLICIT INPUTS:
 13                                  ;      FFREE - FIRST FREE ADDRESS OF MEMORY. THIS ADDRESS IS GIVEN TO UDA
 14                                  ;              AS START OF RING BUFFER.
 15                                  ;      FSIZE - SIZE OF FREE MEMORY AVAILABLE IN WORDS.
 16                                  ;OUTPUTS:
 17                                  ;      R1 - SIZE OF RING BUFFER IN WORDS IF NO ERROR.
 18                                  ;      R4 - ADDRESS OF UDAIP REGISTER IN UDA.
 19                                  ;      R5 - UNCHANGED.
 20                                  ;
 21                                  ;      Z CLR   IF NO ERR.
 22                                  ;      Z SET   IF ANY ERROR REPORTED
 23                                  ;
 24                                  ;CHECK IF ENOUGH FREE MEMORY FOR RING BUFFER
 25
 26 104612                          UDAINT:
    104612  010346                          MOV     R3, (SP)        ;;PUSH R3 ON STACK
 27 104614  010400                          MOV     R4,R0           ;GET MESSAGE LENGTH
 28 104616  000300                          SWAB    R0
 29 104620  042700  177770                  BIC     #177770,R0
 30 104624  004737  105550                  JSR     PC,CLOG         ;COMPUTE LOGARITHMIC VALUE
 31 104630  010102                          MOV     R1,R2           ;SAVE RESULT IN R2
 32 104632  010400                          MOV     R4,R0           ;GET COMMAND LENGTH
 33 104634  000300                          SWAB    R0
 34 104636  006000                          ROR     R0
 35 104640  006000                          ROR     R0
 36 104642  006000                          ROR     R0
 37 104644  042700  177770                  BIC     #177770,R0
 38 104650  004737  105550                  JSR     PC,CLOG         ;COMPUTE LOGARITHMIC VALUE
 39 104654  060201                          ADD     R2,R1           ;ADD THE TWO RESULTS
 40 104656  006301                          ASL     R1              ;MULTIPLY BY 2 WORDS PER RING
 41 104660  062701  000002                  ADD     #<HC.ISZ>/2,R1  ;ADD SPACE FOR INTERRUPT INDICATORS
 42 104664  020137  064414                  CMP     R1,FSIZE        ;COMPARE WITH SIZE OF FREE MEMORY
 43 104670  101402                          BLOS    1$
 44 104672  000137  075410                  JMP     FMERR           ;FATAL ERROR IF NOT ENOUGH MEMORY
 45
 46                                          ;FILL HOST COMMUNICATION AREA WITH ALL ONES
 47
 48 104676  013702  064412           1$:    MOV     FFREE,R2        ;GET FIRST ADDRESS OF RING BUFFER
 49 104702  010103                          MOV     R1,R3           ;GET SIZE OF RING BUFFER
 50 104704  012722  177777           2$:    MOV     #-1,(R2)+       ;WRITE ONES TO BUFFER
 51 104710  005303                          DEC     R3              ;COUNT THE WORDS IN BUFFER
 52 104712  003374                          BGT     2$              ;LOOP UNTIL ENTIRE BUFFER WRITTEN
 53
 54                                          ;DO THE INITIALIZATION
 55
 56 104714  004737  105070                  JSR     PC,UDAIST       ;DO FIRST THREE STEPS
```

```
57 104720 103460                    BCS    9$            ;GET OUT IF UDA MICROCODE REPORTED FAILURE
58 104722 012364 000002             MOV    (R3)+,2(R4)   ;WRITE NEXT WORD TO UDASA REGISTER
59 104726 012700 000310             MOV    #200.,R0      ;GET TRY COUNTER
60 104732 016402 000002      3$:    MOV    2(R4),R2      ;LOOK AT UDASA
61 104736 001410                    BEQ    5$
62 104740 100005                    BPL    4$
63 104742 104455                    TRAP   C$ERDF
   104744 000030                    .WORD  24
   104746 000000                    .WORD  0
   104750 074602                    .WORD  ERRO24
64 104752 000443                    BR     9$
65
66 104754 005300             4$:    DEC    R0
67 104756 001365                    BNE    3$
68 104760 010264 000002      5$:    MOV    R2,2(R4)      ;WRITE 0 TO UDASA (PURGE)
69 104764 011402                    MOV    (R4),R2       ;READ FROM UDAIP (POLL)
70 104766 004737 105410             JSR    PC,UDARSP     ;WAIT FOR STEP OR ERROR BIT
71 104772 103433                    BCS    9$            ;GET OUT IF UDA MICROCODE REPORTED FAILURE
72 104774 010146                    MOV    R1,(SP)       ;;PUSH R1 ON STACK
73 104776 004733                    JSR    PC,@(R3)+     ;CALL LAST ROUTINE
74 105000 012601                    MOV    (SP)+,R1      ;;POP STACK INTO R1
75
76                                  ;CHECK HOST COMMUNICATION AREA FOR ALL ZEROS
77
78 105002 013702 064412             MOV    FFREE,R2      ;GET FIRST ADDRESS OF RING BUFFER
79 105006 010103                    MOV    R1,R3         ;GET SIZE OF RING BUFFER
80 105010 005722             6$:    TST    (R2)+         ;CHECK WORD IN BUFFER
81 105012 001003                    BNE    7$            ;GO TO ERROR REPORTER IF NOT ZERO
82 105014 005303                    DEC    R3            ;COUNT THE WORDS IN BUFFER
83 105016 003374                    BGT    6$            ;LOOP UNTIL ALL WORDS CHECKED
84 105020 000405                    BR     8$
85
86 105022                    7$:
   105022 104455                    TRAP   C$ERDF
   105024 000027                    .WORD  23
   105026 000000                    .WORD  0
   105030 074516                    .WORD  ERRO23
87 105032 000413                    BR     9$
88
89                                  ;SEND GO BIT TO UDASA REGISTER TO END INITIALIZATION
90
91
100 105034 016500 000006     8$:    MOV    C.BST(R5),R0  ;GET BURST VALUE
102 105040 006300                    ASL    R0           ;SHIFT TO POSITION
103 105042 006300                    ASL    R0
104 105044 052700 000001             BIS    #SA.GO,R0    ;SET THE GO BIT
105 105050 010064 000002             MOV    R0,2(R4)     ;SEND TO UDA
106 105054 012603                    MOV    (SP)+,R3     ;;POP STACK INTO R3
107 105056 000244                    CLZ                 ;CLEAR Z AS NO ERROR INDICATION
108 105060 000207                    RTS    PC
109
110                                  ;ERROR RETURN
111
112 105062                    9$:
   105062 012603                    MOV    (SP)+,R3     ;;POP STACK INTO R3
113 105064 000264                    SEZ                 ;SET Z TO INDICATE ERROR OCCURRED
114 105066 000207                    RTS    PC
```

E2

```
 1
 2                                              ;UDAIST
 3                                              ;
 4                                              ;START THE INITIALIZATION PROCESS ON THE SELECTED UDA.
 5                                              ;STOP BEFORE WRITING THE THIRD WORD SO UDA DOES NOT
 6                                              ;ATTEMPT ANY UNIBUS TRANSFERS.
 7                                              ;
 8                                              ;INPUTS:
 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
14 105070                            UDAIST:                             ;>>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<<
15 105070 104422                              TRAP   C$BRK
16 105072 010146                              MOV    R1,-(SP)            ;;PUSH R1 ON STACK
17 105074 052704 100000                       BIS    #SA.STP,R4          ;SET STEP BIT IN DATA WORD
18 105100 010437 105300                       MOV    R4,SND.S1           ;LOAD SEND DATA FOR STEP 1 OF UDA INIT
19 105104 013737 064412 105304                MOV    FFREE,SND.S2        ;GET MEMORY ADDRESS AND
20 105112 062737 000004 105304                ADD    #HC.MSG,SND.S2      ;LOAD SEND DATA FOR STEP 2 OF UDA INIT
21 105120 012737 100000 105310                MOV    #SA.TST,SND.S3      ;LOAD SEND DATA FOR STEP 3 OF UDA INIT
22
23                                              ;START THE INITIALIZATION BY WRITING TO UDAIP REGISTER
24
25 105126 016504 000000                       MOV    C.UADR(R5),R4       ;GET ADDRESS OF UDAIP REGISTER
26 105132 005037 064636                       CLR    NXMAD               ;CLEAR MEMORY ERROR FLAG
27                                                                       ;SETUP TIMEOUT ERROR VECTOR
28 105136 012746 000340                       MOV    #PRI07,-(SP)
   105142 012746 104510                       MOV    #NXMI,-(SP)
   105146 012746 000004                       MOV    #ERRVEC,(SP)
   105152 012746 000003                       MOV    #3,-(SP)
   105156 104437                              TRAP   C$SVEC
   105160 062706 000010                       ADD    #10,SP
29 105164 005764 000002                       TST    2(R4)               ;ACCESS UDASA REGISTER
30 105170 005014                              CLR    (R4)                ;WRITE TO UDAIP
31                                                                       ;RETURN TIMEOUT ERROR VECTOR
32 105172 012700 000004                       MOV    #ERRVEC,R0
   105176 104436                              TRAP   C$CVEC
33 105200 005737 064636                       TST    NXMAD               ;SEE IF A MEMORY ERROR OCCURRED
34 105204 001406                              BEQ    1$
35 105206 104455                              TRAP   C$ERDF
   105210 000046                              .WORD  38
   105212 000000                              .WORD  0
   105214 075044                              .WORD  ERR038
36 105216 000261                              SEC
37 105220 000424                              BR     4$
38
39                                              ;SET UP LOOP PARAMEIERS TO EXECUTE THE FOUR STEPS OF INITIALIZATION
40
41 105222 012737 004000 105546    1$:         MOV    #SA.S1,UDARSD       ;STORE RESPONSE MASK
42 105230 012703 105276                       MOV    #INITBL,R3          ;GET INDEX TO UDA SEND/REPOND INITIALIZE TABLE
43
44                                              ;WAIT FOR AND CHECK RESPONSE DATA
45
46 105234 004737 105410          2$:          JSR    PC,UDARSP           ;WAIT FOR STEP OR ERROR BITS
47 105240 103414                              BCS    4$                  ;EXIT IF ERROR
48 105242 004733                              JSR    PC,@(R3)+           ;CALL RESPONSE CHECKER FOR STEP
```

F 2

```
 49 105244 103412                    BCS    4$              ;GET OUT IF ERROR
 50 105246 006337 105546             ASL    UDARSD          ;SHIFT TO NEXT STEP BIT
 51 105252 032737 04.000 105546      BIT    #SA.S4,UDARSD   ;CHECK IF NOW AT STEP 4
 52 105260 001003                    BNE    3$              ;GET OUT IF SO
 53 105262 012364 000002             MOV    (R3)+,2(R4)     ;WRITE DATA TO UDASA REGISTER
 54 105266 000762                    BR     2$              ;STAY IN LOOP
 55
 56 105270 000241            3$:     CLC                    ;CLEAR CARRY FOR NO ERROR INDICATION
 57 105272                   4$:
    105272 012601                    MOV    (SP)+,R1        ;;POP STACK INTO R1
 58 105274 000207                    RTS    PC
 59
 60                                  ;DATA TO BE SENT AND RECEIVED BY UDA INITIALIZATION
 61
 62 105276 105314            INITBL: .WORD  RSP.S1          ;1ST WORD RESPONSE CHECK ROUTINE
 63 105300 000000            SND.S1: .WORD  0               ;1ST WORD TO SEND TO UDASA
 64 105302 105322                    .WORD  RSP.S2          ;2ND WORD RESPONSE CHECK ROUTINE
 65 105304 000000            SND.S2: .WORD  0               ;2ND WORD TO SEND TO UDASA
 66 105306 105342                    .WORD  RSP.S3          ;3RD WORD RESPONSE CHECK ROUTINE
 67 105310 000000            SND.S3: .WORD  0               ;3RD WORD TO SEND TO UDASA
 68 105312 105360                    .WORD  RSP.S4          ;4TH WORD RESPONSE CHECK ROUTINE
 69
 70                                  ;RESPONSE CHECK FOR FIRST WORD (STEP 1) FROM UDASA
 71                                  ;CHECK FOR PROPER CONTROLLER TYPE
 72
 73 105314 012701 004400     RSP.S1: MOV    #SA.S1+SA.DI,R1 ;SET STEP ONE BIT
 74 105320 000422                    BR     RSP.CK          ;NOW DO A RESPONSE CHECK
 75
 76                                  ;RESPONSE CHECK FOR SECOND WORD (STEP 2) FROM UDASA
 77                                  ;CHECK FOR ECHO OF INTI AND VECTOR
 78
 79 105322 013701 105300     RSP.S2: MOV    SND.S1,R1       ;GET WORD SENT TO UDASA
 80 105326 000301                    SWAB   R1              ;GET HIGH 8 BITS
 81 105330 042701 177400             BIC    #177400,R1
 82 105334 052701 010000             BIS    #SA.S2,R1       ;SET STEP 2 BIT
 83 105340 000412                    BR     RSP.CK          ;NOW DO A RESPONSE CHECK
 84
 85                                  ;RESPONSE CHECK FOR THIRD WORD (STEP 3) FROM UDASA
 86                                  ;CHECK FOR ECHO OF MESSAGE AND COMMAND RING LENGTHS
 87
 88 105342 013701 105300     RSP.S3: MOV    SND.S1,R1       ;GET WORD SENT TO UDASA
 89 105346 042701 177400             BIC    #177400,R1      ;JUST LOW 8 BITS
 90 105352 052701 020000             BIS    #SA.S3,R1       ;SET STEP 3 BIT
 91 105356 000403                    BR     RSP.CK          ;NOW DO A RESPONSE CHECK
 92
 93                                  ;RESPONSE CHECK FOR FOURTH WORD (STEP 4) FROM UDASA
 94                                  ;CHECK FOR ECHO OF PURGE AND LFAIL BITS
 95
 96 105360 010201            RSP.S4: MOV    R2,R1           ;GET RESPONSE FROM UDA AND
 97 105362 010237 105406             MOV    R2,SSTEP4       ;SAVE STEP 4 VALUF.
 98
 99                                  ;RESPONSE CHECK, COMPARE EXPECTED DATA IN R1 WITH ACTUAL DATA IN R2
100
101 105366 020102            RSP.CK: CMP    R1,R2           ;COMPARE THE DATA
102 105370 001405                    BEQ    1$              ;EXIT IF COMPARED CORRECTLY
103                                  ;ERROR, UDA DID NOT RETURN CORRECT DATA IN
104                                  ;        UDASA REGISTER DURING INITIALIZATION
```

G2

```
105 105372  104455                    TRAP     C$ERDF
    105374  000031                    .WORD    25
    105376  000000                    .WORD    0
    105400  074616                    .WORD    ERRO25
106 105402  000261                    SEC
107 105404  000207          1$:       RTS      PC
108
109 105406  000000          SSTEP4:   .WORD    0                    ;SAVE STEP 4 VALUE HERE
```

```
  1
  2                                          ;UDARSP
  3                                          ;
  4                                          ;WAIT FOR UDA TO RESPOND WITH DATA IN UDASA REGISTER.
  5                                          ;EITHER STEP BIT FROM MASK IN LOCATION UDARSD OR ERROR BIT
  6                                          ;WILL CAUSE A TERMINATION.
  7                                          ;AN ERROR MESSAGE WILL BE PRINTED IF THE UDA DOES NOT RESPOND
  8                                          ;IN 10 SECONDS OR IF ERROR SETS.
  9                                          ;
 10                                          ;INPUTS·
 11                                          ;       UDASRD - MASK OF STEP BIT TO LOOK FOR
 12                                          ;       R5 - ADDRESS OF CONTROLLER TABLE
 13                                          ;       R4 - ADDRESS OF UDAIP REGISTER
 14                                          ;OUTPUTS:
 15                                          ;       ERROR MESSAGE IF TIME OUT ON RESPONSE OR ERROR BIT SETS
 16                                          ;       R2 - DATA FROM UDASA REGISTER
 17                                          ;       CARRY SET IF ERROR BIT SETS OR TIME OUT
 18
 19 105410                         UDARSP:
    105410  010146                           MOV    R1,-(SP)            ;;PUSH R1 ON STACK
 20 105412  052737  100000  105546           BIS    #SA.ERR,UDARSD      ;SET ERROR BIT IN MASK WORD
 21 105420  012700  000012                   MOV    #10.,R0             ;SET UP FOR 10 SECOND TIMEOUT
 22 105424  010501                           MOV    R5,R1               ;POINT TO COUNTER IN CONTROLLER TABLE
 23 105426  062701  000040                   ADD    #C.TO,R1
 24 105432  004737  104530                   JSR    PC,SETTO
 25 105436  012601                           MOV    (SP)+,R1            ;;POP STACK INTO R1
 26 105440  033764  105546  000002  1$:      BIT    UDARSD,2(R4)        ;LOOK AT ERROR AND STEP BIT
 27 105446  001024                           BNE    3$                  ;BRANCH IF EITHER SET
 28                                          ;>>>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<<
 29 105450  104422                           TRAP   C$BRK
 30 105452  005737  064616                   TST    KW.CSR              ;SEE IF CLOCK ON SYSTEM
 31 105456  001770                           BEQ    1$
 32 105460  023765  064630  000042           CMP    KW.EL+2,C.TOH(R5)   ;CHECK IF TIME OUT OCCURRED
 33 105466  101005                           BHI    2$
 34 105470  001363                           BNE    1$
 35 105472  023765  064626  000040           CMP    KW.EL,C.TO(R5)
 36 105500  103757                           BLO    1$
 37 105502  016402  000002          2$:      MOV    2(R4),R2            ;GET REGISTER CONTENTS
 38 105506  104455                           TRAP   C$ERDF
    105510  000026                           .WORD  22
    105512  000000                           .WORD  0
    105514  074470                           .WORD  ERRO22
 39 105516  000407                           BR     4$
 40
 41                                          ;CHECK IF ERROR BIT SET
 42
 43 105520  016402  000002          3$:      MOV    2(R4),R2            ;GET REGISTER CONTENTS
 44 105524  100006                           BPL    5$                  ;EXIT IF ERROR NOT SET
 45 105526  104455                           TRAP   C$ERDF
    105530  000025                           .WORD  21
    105532  000000                           .WORD  0
    105534  074430                           .WORD  ERRO21
 46 105536  000261                  4$:      SEC
 47 105540  000207                           RTS    PC
 48
 49                                          ;NORMAL EXIT
 50
```

I2

```
   51 105542  000241        5$:         CLC                                      ;CLEAR CARRY AS NO ERROR INDICATION
   52 105544  000207                    RTS      PC
   53
   54                                             ;LOCATION FOR STEP BIT MASK
   55
   56 105546  000000        UDARSD: .WORD   0                                    ;LOAD BY CALLING ROUTINE
```

J2

```
    1
    2                              ;CLOG
    3                              ;
    4                              ;COMPUTE LOGARITHMIC VALUE OF NUMBER TO BASE 2.
    5                              ;
    6                              ;INPUTS:
    7                                      RO - LOGARITHM TO BE CONVERTED
    8                              ;OUTPUTS:
    9                              ;       R1 - VALUE OF 2 RAISED TO POWER OF INPUT NUMBER
   10
   11 105550                      CLOG:
      105550  010046                      MOV     RO,-(SP)        ;;PUSH RO ON STACK
   12 105552  005001                      CLR     R1              ;SET UP ZERO START VALUE
   13 105554  000261                      SEC                     ;WITH CARRY READY TO SHIFT IN
   14 105556  006101              1$:     ROL     R1              ;SHIFT TO LEFT
   15 105560  005300                      DEC     RO              ;UNTIL RO
   16 105562  100375                      BPL     1$              ;GOES NEGATIVE
   17 105564  012600                      MOV     (SP)+,RO        ;;POP STACK INTO RO
   18 105566  000207                      RTS     PC
```

K2

CZUDCEO UDA & DISK DRV DIAG MACRO VO5.00 Wednesday 04 Jan-84 16:12 Page 178

```
 1
 2                                ;RDDLL
 3                                ;
 4                                ;READ DISK DRIVE DOWNLINE LOAD PROGRAM INTO MEMORY
 5                                ;
 6                                ;INPUTS:
 7                                ;      DLLNAM - NAME OF PROGRAM IN RAD50 (TWO WORDS)
 8                                ;OUTPUTS:
 9                                ;      FREE MEMORY CONTAINING PROGRAM
10                                ;      CARRY CLEAR IF NO ERROR, CARRY SET IF PROGRAM NOT FOUND
11
16 105570  012701  000006  RDDLL:  MOV     #6.,R1          ;TYPE OF PROGRAM IN DATA FILE
17 105574  004737  105630          CALL    RDREC           ;READ PROGRAM INTO MEMORY
18 105600  006101                  ROL     R1                  ;PRESERVE CARRY STATE IN R1
19 105602  004737  105612          CALL    CLOSEF          ;  WHILE CLOSING THE DATA FILE
20 105606  006001                  ROR     R1                  ;  AS NORMAL POSITION IS LOST
21 105610  000207                  RETURN
```

L2

SFO 0231

```
 1
 2                                      ;CLOSEF
 3                                      ;
 4                                      ;CLOSE DATA FILE FOR DM PROGRAMS
 5                                      ;
 6                                      ;INPUTS:
 7                                      ;      FILOPN - ZERO IF FILE NOT OPEN
 8                                      ;OUTPUTS:
 9                                      ;      NONE
10
12 105612  005737  064474   CLOSEF: TST     FILOPN          ;SEE IF FILE CURRENTLY OPEN
13 105616  001403                     BEQ     1$
14 105620  104435                     TRAP    C$CLOS
15 105622  005037  064474             CLR     FILOPN          ;AND MARK AS SO
16 105626  000207            1$:     RETURN
```

M2

```
 1
 2                                       ;RDREC
 3                                       ;
 4        .                              ;READ A RECORD FROM THE INPUT FILE. PLACE DATA INTO FREE MEMORY.
 5                                       ;
 6                                       ;INPUTS:
 7                                       ;       R1 - FILE TYPE
 8                                       ;               1   UDA52 TEST 1 DM PROGRAM
 9                                       ;               2 - UDA52 TEST 2 DM PROGRAM
10                                       ;               3 - UDA52 TEST 3 DM PROGRAM
11                                       ;               4 - TEST 4 QUESTIONS
12                                       ;               5   UDA52 TEST 4 DM PROGRAM
13                                       ;               6 - DRIVE DIAGNOSTIC DOWNLINE LOAD PROGRAM
14                                       ;       DLLNAM   IF R1 CONTAINS 6, TWO WORDS AT THIS ADDRESS CONTAIN
15                                       ;                NAME OF PROGRAM IN RAD50.
16                                       ;       R5 - ADJUSTED ADDRESS WHERE TO BRING DATA INTO.
17                                       ;OUTPUTS:
18                                       ;       DATA FROM RECORD IN MEMORY
19                                       ;       CARRY CLEAR IF NO ERROR. CARRY SET IF ERROR
20
21 105630                               RDREC:
   105630  010046                               MOV      R0,-(SP)          ;;PUSH R0 ON STACK
   105632  010146                               MOV      R1,-(SP)          ;;PUSH R1 ON STACK
   105634  010246                               MOV      R2,-(SP)          ;;PUSH R2 ON STACK
   105636  010346                               MOV      R3,-(SP)          ;;PUSH R3 ON STACK
   105640  010446                               MOV      R4,-(SP)          ;;PUSH R4 ON STACK
   105642  010546                               MOV      R5, (SP)          ;;PUSH R5 ON STACK
22 105644  005037  064442                       CLR      FNUM
23 105650  005737  064474                       TST      FILOPN            ;SEE IF FILE ALREADY OPEN
24 105654  001005                               BNE      RDSTS
25 105656  012700  064456                       MOV      #FNAME,R0
   105662  104434                               TRAP     C$OPEN
26 105664  005237  064474                       INC      FILOPN            ;AND MARK AS OPEN
27 105670  005105                       RDSTS:   COM      R5                ;COMPLEMENT LOAD ADDRESS (SEARCH MODE)
28 105672                               RDST:                              ;>>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<<
29 105672  104422                               TRAP     C$BRK
30                                                                         ;READ A BYTE FROM FILE
31 105674  104426                               TRAP     C$GETB
   105676  110004                               MOVB     R0,R4
32 105700  005704                               TST      R4                ;IF ZERO
33 105702  001773                               BEQ      RDST              ; KEEP READING
34 105704  022704  000001                       CMP      #1,R4             ;WHEN NOT ZERO
35 105710  001142                               BNE      RWRDE1            ; IT BETTER BE A ONE
36                                                                         ;READ A BYTE FROM FILE
37 105712  104426                               TRAP     C$GETB
38 105714  060004                               ADD      R0,R4
39 105716  005700                               TST      R0                ; IF ZERO, PROCESS DATA
40 105720  001431                               BEQ      RDDAT
41 105722  020001                               CMP      R0,R1             ;CHECK IF TYPE OF FILE LOOKING FOR
42 105724  103427                               BLO      RDDAT             ;IF TOO SOON IN FILE, KEEP SEARCHING
43 105726  101121                               BHI      RDERR             ;IF PAST TYPE, GIVE ERROR RETURN
44 105730  004737  106150                       CALL     FWORD             ;GET NEXT TWO WORDS
45 105734  013702  064472                       MOV      FDATA,R2
46 105740  004737  106150                       CALL     FWORD
47                                                                         ;READ A BYTE FROM FILE
48 105744  104426                               TRAP     C$GETB
49 105746  060004                               ADD      R0,R4             ;ADD TO COMPUTED SUM
```

```
 50 105750   105704                        TSTB    R4              ;SEE IF THIS SUM IS ZERO
 51 105752   001121                        BNE     RWRDE1          ;  IF NOT, REPORT CHECKSUM ERROR
 52 105754   020127   000006               CMP     R1,#6           ;IF FILE TYPE IS A 6
 53 105760   001007                        BNE     1$
 54 105762   023702   064674               CMP     DLLNAM,R2       ;  MATCH THE PROGRAM NAME
 55 105766   001341                        BNE     RDST            ;KEEP SEARCHING IF NOT DESIRED PROGRAM
 56 105770   023737   064676   064472      CMP     DLLNAM+2,FDATA
 57 105776   001335                        BNE     RDST
 58 106000   005105               1$:      COM     R5              ;GET STORAGE ADDRESS
 59 106002   000733                        BR      RDST            ;SWITCH FROM SEARCH TO STORE MODE
 60
 61 106004   004737   106150      RDDAT:   CALL    FWORD           ;READ BYTE COUNT
 62 106010   013703   064472               MOV     FDATA,R3        ;SAVE IN R3
 63 106014   004737   106150               CALL    FWORD           ;READ LOAD ADDRESS
 64 106020   162703   000006               SUB     #6,R3           ;SUBTRACT BYTES ALREADY READ FROM BYTE COUNT
 65 106024   001431                        BEQ     RWORDT          ;IF RESULT IS ZERO, THIS IS A TRANSFER BLOCK
 66 106026   005705                        TST     R5              ;IF IN SEARCH MODE,
 67 106030   100413                        BMI     ·$              ;  BYPASS TRANSFER ADDRESS COMPUTATION
 68 106032   013701   064472               MOV     FDATA,R1        ;GET LOAD ADDRESS
 69 106036   060501                        ADD     R5,R1           ; R1 -> REAL STARTING ADDRESS
 70 106040   020127   002122               CMP     R1,#STORAG      ; R1 MUST BE GREATER THAN STORAG
 71 106044   103452                        BLO     RDERR           ; IF NOT, ERROR
 72 106046   060301                        ADD     R3,R1           ;ADD BYTES IN RECORD
 73 106050   022701   064342               CMP     #<STORAG+STOSIZ>,R1  ; R1 MUST BE LESS THAN ENDING ADDRESS
 74 106054   103446                        BLO     RDERR           ; IF NOT, ERROR
 75 106056   160301                        SUB     R3,R1
 76 106060                        1$:                              ;READ A BYTE FROM FILE
 77 106060   104426                        TRAP    C$GETB
 78 106062   005705                        TST     R5              ;IF IN SEARCH MODE,
 79 106064   100401                        BMI     2$              ;  BYPASS DATA STORAGE
 80 106066   110021                        MOVB    R0,(R1)+        ;STORE IN MEMORY
 81 106070   060004               2$:      ADD     R0,R4           ;UPDATE CHECKSUM
 82 106072   005303                        DEC     R3              ;COUNT THE BYTE
 83 106074   001371                        BNE     1$              ;GET THEM ALL
 84                                                                ;READ A BYTE FROM FILE
 85 106076   104426                        TRAP    C$GETB
 86 106100   060004                        ADD     R0,R4           ;ADD
 87 106102   105704                        TSTB    R4              ;IF CHECKSUM CORRECT,
 88 106104   001672                        BEQ     RDST            ;  THEN GO READ NEXT RECORD
 89 106106   000443                        BR      RWRDE1          ;  ELSE REPORT ERROR
 90
 91 106110                        RWORDT:                          ;READ A BYTE FROM FILE
 92 106110   104426                        TRAP    C$GETB
 93 106112   060004                        ADD     R0,R4           ;ADD TO COMPUTED CHECKSUM
 94 106114   105704                        TSTB    R4              ;CHECK LOW BYTE OF SUM
 95 106116   001037                        BNE     RWRDE1          ;BRANCH IF CHECKSUM ERROR
 96 106120   005705                        TST     R5              ;IF IN SEARCH MODE,
 97 106122   100663                        BMI     RDST            ;  KEEP ON SEARCHING
 98 106124   012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
    106126   012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
    106130   012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
    106132   012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
    106134   012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
    106136   012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 99 106140   010137   064442               MOV     R1,FNUM
100 106144   000241                        CLC
101 106146   000207                        RETURN
```

B3

```
102
103  106150                         FWORD:                              ;READ A BYTE FROM FILE
104  106150  104426                         TRAP    C$GETB
105  106152  060004                         ADD     R0,R4               ;UPDATE CHECKSUM ERROR
106  106154  110037  064472                 MOVB    R0,FDATA            ;START TO BUILD WORD
107                                                                      ;READ A BYTE FROM FILE
108  106160  104426                         TRAP    C$GETB
109  106162  060004                         ADD     R0,R4               ;UPDATE CHECKSUM
110  106164  110037  064473                 MOVB    R0,FDATA+1          ;COMPLETE WORD
111  106170  000207                         RETURN
112
113  106172  004737  105612  RDERR:         CALL    CLOSEF              ;CLOSE FILE AS POSITION IS LOST
114  106176  012605                         MOV     (SP)+,R5            ;;POP STACK INTO R5
     106200  012604                         MOV     (SP)+,R4            ;;POP STACK INTO R4
     106202  012603                         MOV     (SP)+,R3            ;;POP STACK INTO R3
     106204  012602                         MOV     (SP)+,R2            ;;POP STACK INTO R2
     106206  012601                         MOV     (SP)+,R1            ;;POP STACK INTO R1
     106210  012600                         MOV     (SP)+,R0            ;;POP STACK INTO R0
115  106212  000261                         SEC                         ;ERROR RETURN, FILE NOT FOUND
116  106214  000207                         RETURN
117
118  106216                         RWRDE1:
     106216  104454                         TRAP    C$ERSF
     106220  000005                         .WORD   5
     106222  000000                         .WORD   0
     106224  074336                         .WORD   ERR005
119                                                                      ;DO CLEAN UP TRAP
120  106226  104444                         TRAP    C$DCLN
```

```
   1
   2                                  ;KW11I
   3                                  ;
   4                                  ;CLOCK INTERRUPT SERVICE ROUTINE
   5
   6 106230                           KW11I::
   7 106230  062737  000001  064626           ADD    #1,KW.EL              ;COUNT THE INTERRUPT
   8 106236  005537  064630                   ADC KW.EL+2
   9 106242  012777  000105  156346           MOV    @KW.OUT,@KW.CSR     ;RESTART THE CLOCK
  10 106250                           L10040:
     106250  000002                           RTI
  11
  12 106252                           INTSRV::
  13 106252  005237  064454                   INC    INTRCV               ; FLAG INTERRUPT AS RECEIVED
  14 106256                           L10041:
     106256  000002                           RTI
```

D3

CZUDCEO UDA & DISK DRV DIAG MACRO V05.00 Wednesday 04 Jan 84 16:12 Page 182
PRE-PROGRAMMED SUBROUTINES   SFQ 0236

```
 1
 2                                         ;RESET
 3                                         ;
 4                                         ; RESET ALL UDA-50S IN THE CONTROLLER TABLES
 5                                         ;
 6                                         ; INPUTS:
 7                                         ;     IPADRS    CONTAINS ALL IP ADDRESSES
 8                                         ; OUTPUTS:
 9                                         ;     NONE
10                                         ;
11
12 106260  005037  064636       RESET:  CLR     NXMAD           ;CLEAR NON EXISTANT MEMORY ADDRESS
13 106264  010346                        MOV     R3, (SP)        ;;PUSH R3 ON STACK
   106266  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
14                                                               ;SETUP TIMEOUT ERROR VECTOR
15 106270  012746  000340                MOV     #PRI07,-(SP)
   106274  012746  104510                MOV     #NXMI,-(SP)
   106300  012746  000004                MOV     #ERRVEC,-(SP)
   106304  012746  000003                MOV     #3,-(SP)
   106310  104437                        TRAP    C$SVEC
   106312  062706  000010                ADD     #10,SP
16 106316  012703  000010                MOV     #8.,R3          ;R3 = COUNTER OF ENTRIES
17 106322  012704  064534                MOV     #IPADRS,R4      ;R4 -> IP ADDRESS
18 106326  005714                1$:     TST     (R4)            ;IS THERE AN ENTRY?
19 106330  001403                        BEQ     2$              ;IF NOT, DONE
20 106332  005034                        CLR     @(R4)+          ;INIT UDA
21 106334  005303                        DEC     R3              ;MAKE SURE WE DO NOT EXTEND OVER AREA
22 106336  001373                        BNE     1$              ;IF NOT DONE, BRANCH
23 106340  005737  064616        2$:     TST     KW.CSR          ;SEE IF CLOCK PRESENT,
24 106344  001403                        BEQ     3$              ;BRANCH IF NOT, ELSE
25 106346  012777  000105  156242        MOV     #KW.OUT,@KW.CSR ;START THE CLOCK.
26 106354                        3$:
   106354  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
   106356  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
27 106360  000207                        RETURN
```

E3

```
  1                                  ;RNTIME
  2                                  ;
  3                                  ;
  4                                  ;PRINT RUNTIME
  5                                  ;
  6                                  ;INPUTS:
  7                                  ;       KW.EL - CONTAINS ELAPSED TIM
  8                                  ;       KW.HZ - HERTZ OF CLOCK
  9                                  ;OUTPUTS:
 10                                  ;       IF CLOCK ON SYSTEM:
 11                                  ;       "  RNTIME HH:MM:SS " PRINTED
 12                                  ;       IF NO CLOCK: ONE SPACE IS PRINTED
 13
 14 106362 005737 064616    RNTIME: TST     KW.CSR              ;CHECK IF A CLOCK PRESENT
 15 106366 001465                   BEQ     RNTIMX              ;BRANCH IF NOT
 16 106370 010046                   MOV     R0,-(SP)            ;;PUSH R0 ON STAC'
    106372 010346                   MOV     R3,-(SP)            ;;PUSH R3 ON STACK
    106374 010446                   MOV     R4,-(SP)            ;;PUSH R4 ON STACK
    106376 010546                   MOV     R5,-(SP)            ;;PUSH R5 ON STACK
 17 106400 013703 064626           MOV     KW.EL,R3                    ;GET ELAPSED TIME
 18 106404 013704 064630           MOV     KW.EL+2,R4
 19 106410 013700 064624           MOV     KW.HZ,R0                    ;GET SPEED OF CLOCK
 20 106414 004737 102704           CALL    DIVIDE              ;COMPUTE SECONDS OF ELAPSED TIME
 21 106420 012700 000074           MOV     #60.,R0             ;NOW DIVIDE BY 60
 22 106424 004737 102704           CALL    DIVIDE              ; TO COMPUTE MINUTES
 23 106430 010546                   MOV     R5,-(SP)            ;;PUSH R5 ON STACK
 24 106432 004737 102704           CALL    DIVIDE              ;DIVIDE BY 60 AGAIN
 25 106436 010346                   MOV     R3,(SP)             ;PUSH R3 ON STACK
    106440 004137 075722           JSR     R1,LPNT      ;CALL  LPNT PRINT ROUTINE
    106444 065045                   .WORD   RNTIM               ;ADDRESS OF ASCIZ STRING
    106446 000002                   .WORD   ARG.CT              ;ARGUMENT COUNT + 2
 26 106450 020527 000011           CMP     R5,#9.              ;IF MINUTES 9 OR LESS
 27 106454 003004                   BGT     1$
 28 106456 112700 000060           MOVB    #'0,R0              ;STORE #'0 IN R0 AND
    106462 004737 075506           JSR     PC,PRINTC           ;PRINT THE CHARACTER.
 29 106466                  1$:
    106466 010546                   MOV     R5,-(SP)            ;PUSH R5 ON STACK
    106470 004137 075722           JSR     R1,LPNT      ;CALL  LPNT PRINT ROUTINE
    106474 065070                   .WORD   RNTIM1              ;ADDRESS OF ASCIZ STRING
    106476 000002                   .WORD   ARG.CT              ;ARGUMENT COUNT + 2
 30 106500 012605                   MOV     (SP)+,R5            ;;POP STACK INTO R5
 31 106502 020527 000011           CMP     R5,#9.              ;IF 9 OR LESS
 32 106506 003004                   BGT     2$
 33 106510 112700 000060           MOVB    #'0,R0              ;STORE #'0 IN R0 AND
    106514 004737 075506           JSR     PC,PRINTC           ;PRINT THE CHARACTER.
 34 106520                  2$:
    106520 010546                   MOV     R5,(SP)             ;PUSH R5 ON STACK
    106522 004137 075722           JSR     R1,LPNT      ;CALL  LPNT PRINT ROUTINE
    106526 065076                   .WORD   RNTIM2              ;ADDRESS OF ASCIZ STRING
    106530 000002                   .WORD   ARG.CT              ;ARGUMENT COUNT + 2
 35 106532 012605                   MOV     (SP)+,R5            ;;POP STACK INTO R5
    106534 012604                   MOV     (SP)+,R4            ;;POP STACK INTO R4
    106536 012603                   MOV     (SP)+,R3            ;;POP STACK INTO R3
    106540 012600                   MOV     (SP)+,R0            ;;POP STACK INTO R0
 36 106542                  RNTIMX:
    106542 112700 000040           MOVB    #' ,R0              ;STORE #   IN R0 AND
    106546 004737 075506           JSR     PC,PRINTC           ;PRINT THE CHARACTER.
```

F3

37 106552 000207   RETURN

```
     1
     2                                       ;WCHNG
     3                                       ;
     4                                       ;          WAIT UNTIL UDASA CHANGES FROM WHAT IS IN WCHNGD
     5
     6 106554  012700  000012      WCHNG:    MOV     #10.,RO          ;SET TIMEOUT FOR 10 SECONDS
     7 106560  010501                        MOV     R5,R1            ;POINT TO CONTROLLER TABLE
     8 106562  062701  000040                ADD     #C.TO,R1
     9 106566  004737  104530                CALL    SETTO
    10 106572  026437  000002  106650  1$:   CMP     2(R4),WCHNGD     ;SEE IF CHANGED
    11 106600  001022                        BNE     2$
    12                                       ;>>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<<
    13 106602  104422                        TRAP    C$BRK
    14 106604  005737  064616                TST     KW.CSR           ;SEE IF CLOCK ON SYSTEM
    15 106610  001770                        BEQ     1$
    16 106612  023765  064630  000042        CMP     KW.EL   .C.TOH(R5)   ;CHECK IF TIME OUT OCCURRED
    17 106620  101005                        BHI 3$
    18 106622  001363                        BNE     1$
    19 106624  023765  064626  000040        CMP     KW.EL,C.TO(R5)
    20 106632  103757                        BLO 1$
    21 106634                       3$:
       106634  104455                        TRAP    C$ERDF
       106636  000033                        .WORD   27
       106640  000000                        .WORD   0
       106642  074654                        .WORD   ERR027
    22 106644  000264                        SEZ                      ; FLAG AS ERROR
    23 106646  000207              2$:        RETURN                   ; RETURN TO CALLING PROGRAM
    24
    25 106650              WCHNGD:  .BLKW   1               ; OLD PORT CONTENTS
    26 106652              BRLEV:   .BLKW   1               ; WORD FOR BRANCH LEVEL STORAGE
    36
    43
```

H3

```
 12                                        .SBTTL   REPORT CODING SECTION
 40
 42                                     ;**
 43                                     ; THE REPORT CODING SECTION CONTAINS THE
 44                                     ; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
 45                                     ;--
 46
 47 106654                             L$RPT::
 48
 49 106654  010046                             MOV     R0,-(SP)          ;;PUSH R0 ON STACK
    106656  010146                             MOV     R1,-(SP)          ;;PUSH R1 ON STACK
    106660  010246                             MOV     R2,-(SP)          ;;PUSH R2 ON STACK
    106662  010346                             MOV     R3,-(SP)          ;;PUSH R3 ON STACK
    106664  010446                             MOV     R4,-(SP)          ;;PUSH R4 ON STACK
    106666  010546                             MOV     R5,-(SP)          ;;PUSH R5 ON STACK
 50 106670  013746  064444                     MOV     TNUM,-(SP)        ;PUSH TNUM ON STACK
    106674  004137  075714                     JSR     R1,LPNTS          ;CALL    LPNTS PRINT ROUTINE
    106700  107262                             .WORD   RPTMSG            ;ADDRESS OF ASCIZ STRING
    106702  000002                             .WORD   ARG.CT            ;ARGUMENT COUNT * 2
 51 106704  004737  106362                     CALL    RNTIME            ;GET RUNTIME PARAMETERS
 52 106710  112700  000015                     MOVB    #CR,R0            ;STORE #CR IN R0 AND
    106714  004737  075506                     JSR     PC,PRINTC         ;PRINT THE CHARACTER.
 53 106720  012701  064632                     MOV     #STIME,R1         ;AT 15 MINUTES FROM NOW
 57 106724  012700  001604                     MOV     #15.*60.,R0       ;SET TIME FOR NEXT REPORT
 58 106730  004737  104530                     CALL    SETTO
 66 106734  022737  000004  064444             CMP     #4,TNUM           ;IF NOT TEST 4
 68 106742  001402                             BEQ     1$                ;BRANCH IF SO, ELSE
 69 106744  000137  107242                     JMP     RPTXX             ;EXIT REPORT SECTION.
 70
 71 106750                             1$:
    106750  004137  075714                     JSR     R1,LPNTS          ;CALL    LPNTS PRINT ROUTINE
    106754  107316                             .WORD   RPTMSH            ;ADDRESS OF ASCIZ STRING
    106756  000000                             .WORD   ARG.CT            ;ARGUMENT COUNT * 2
 72 106760  013705  064424                     MOV     CTABS,R5          ;GET ADDRESS OF 1ST CONTROLLER TABLE
 73
 74 106764  005765  000002     RPTCT:          TST     C.UNIT(R5)        ;SEE IF CONTROLLER AVAILABLE FOR TESTING
 76 106770  100520                             BMI     RPTCTN
 83 106772  010504                             MOV     R5,R4             ;COMPUTE ADDRESS OF DRIVE TABLE POINTERS
 84 106774  062704  000020                     ADD     #C.DR0,R4
 85 107000  012703  000010                     MOV     #8.,R3            ;GET COUNT OF DRIVES
 86 107004  012401     RPTDT:                  MOV     (R4)+,R1          ;LOOK AT POINTER
 87 107006  001511                             BEQ     RPTCTN            ;GO TO NEXT IF NO TABLE
 88 107010  005761  000002                     TST     D.UNIT(R1)        ;SEE IF DRIVE AVAILABLE
 90 107014  100504                             BMI     RPTDTN
 98 107016  010346                             MOV     R3,-(SP)          ;;PUSH R3 ON STACK
    107020  010446                             MOV     R4,-(SP)          ;;PUSH R4 ON STACK
    107022  010546                             MOV     R5,-(SP)          ;;PUSH R5 ON STACK
    107024  010146                             MOV     R1,-(SP)          ;;PUSH R1 ON STACK
 99 107026  012700  064476                     MOV     #TEMP,R0          ;PLACE 18 SPACE CHARACTERS INTO
100 107032  012701  000022                     MOV     #18.,R1           ;  TEMP STORAGE
101 107036  112720  000040     1$:             MOVB    #' ,(R0)+
102 107042  005301                             DEC     R1
103 107044  001374                             BNE     1$
104 107046  005010                             CLR     (R0)              ;THEN A NULL CHARACTER
105 107050  011605                             MOV     (SP),R5           ;GET DRIVE TABLE STORAGE ADDRESS
106 107052  016501  000200                     MOV     D.SERN(R5),R1     ;GET SERIAL NUMBER
107 107056  016502  000202                     MOV     D.SERN+2(R5),R2
```

I3

```
108 107062  016503  000204            MOV    D.SERN+4(R5),R3
109 107066  005004                     CLR    R4
110 107070  004737  102742      2$:    CALL   DIV10                    ;DIVIDE BY 10
111 107074  062705  000060             ADD    #'0,R5                   ;CONVERT TO ASCII CHARACTER
112 107100  110540                      MOVB   R5,-(R0)                 ;PUT DIGIT INTO TEMP STORAGE
113 107102  010146                     MOV    R1,-(SP)
114 107104  050216                     BIS    R2,(SP)                  ;SEE IF QUOTIENT IS ZERO
115 107106  050316                     BIS    R3,(SP)
116 107110  050426                     BIS    R4,(SP)+
117 107112  001366                     BNE    2$                       ;IF NOT, DIVIDE AGAIN
118 107114  012601                     MOV    (SP)+,R1                 ;;POP STACK INTO R1
119 107116  016146  000164             MOV    D.XFRW(R1),-(SP)
    107122  016146  000166             MOV    D.XFRR(R1),-(SP)
    107126  016146  000174             MOV    D.SEEK(R1),-(SP)
    107132  012746  064476             MOV    #TEMP,-(SP)
    107136  011146                     MOV    (R1),-(SP)
    107140  016146  000002             MOV    D.UNIT(R1),-(SP)
    107144  012746  107534             MOV    #RPTMSD,-(SP)
    107150  012746  000007             MOV    #7,-(SP)
    107154  010600                     MOV    SP,R0
    107156  104416                     TRAP   C$PNTS
    107160  062706  000020             ADD    #20,SP
121 107164  016146  000176             MOV    D.ECCC(R1),-(SP)
    107170  016146  000172             MOV    D.SERR(R1),-(SP)
    107174  016146  000170             MOV    D.HERR(R1),-(SP)
    107200  012746  107603             MOV    #RPTMD2,-(SP)
    107204  012746  000004             MOV    #4,-(SP)
    107210  010600                     MOV    SP,R0
    107212  104416                     TRAP   C$PNTS
    107214  062706  000012             ADD    #12,SP
145 107220  012605                     MOV    (SP)+,R5                 ;;POP STACK INTO R5
    107222  012604                     MOV    (SP)+,R4                 ;;POP STACK INTO R4
    107224  012603                     MOV    (SP)+,R3                 ;;POP STACK INTO R3
146 107226  005303             RPTDTN: DEC    R3                       ;COUNT THE DRIVE TABLES
147 107230  003265                     BGT    RPTDT                    ;REPEAT FOR ALL DRIVE TABLES
148 107232  062705  000046     RPTCTN: ADD    #C.SIZE,R5               ;GO TO NEXT CONTROLLER TABLE
149 107236  005715                     TST    (R5)
154 107240  001251                     BNE    RPTCT
156 107242             RPTXX:
    107242  012605                     MOV    (SP)+,R5                 ;;POP STACK INTO R5
    107244  012604                     MOV    (SP)+,R4                 ;;POP STACK INTO R4
    107246  012603                     MOV    (SP)+,R3                 ;;POP STACK INTO R3
    107250  012602                     MOV    (SP)+,R2                 ;;POP STACK INTO R2
    107252  012601                     MOV    (SP)+,R1                 ;;POP STACK INTO R1
    107254  012600                     MOV    (SP)+,R0                 ;;POP STACK INTO R0
168
169 107256  000167                     .WORD  J$JMP
    107260  000344                     .WORD  L10042-2 .
170
174 107262  116  042  124     RPTMSG: .ASCIZ \N"TEST "D3" IN PROGRESS.  "\
175 107316  116  042  125     RPTMSH: .ASCII \N"UNIT DRIVE         SERIAL-NUMBER SEEKS MBYTES  MBYTES  HARD    SOFT     ECC N\
176 107430  042  040  040             .ASCIZ \'                                X1000   READ  WRITTEN ERRORS ERRORS'N\
177 107534  045  123  062     RPTMSD: .ASCIZ \#S2#D2#S3#D3#S1#T#S1#D5#S2#D5#S3#D5#S2\
178 107603  045  104  065     RPTMD2: .ASCIZ \#D5#S2#D5#S1#D5#N\
198                                   .EVEN
199
200 107626             L10042:
```

J3

107626 104425   TRAP   C$RPT

K3

```
 1
 2                              .SBTTL   PROTECTION TABLE
 3
 4                              ;.+
 5                              ; THIS TABLE IS USED BY THE RUNTIME SERVICES
 6                              ; TO PROTECT THE LOAD MEDIA.
 7                              ;.-
 8
 9 107630                      L$PROT::
10
11 107630  177777                           -1              ;OFFSET INTO P-TABLE FOR CSR ADDRESS
12 107632  177777                           -1              ;OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
13 107634  17777                            -1              ;OFFSET INTO P TABLE FOR DRIVE NUMBER
14
```

L3

```
 1                              .SBTTL   INITIALIZE SECTION
 2
 3                              ;++
 4                              ; THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
 5                              ; AT THE BEGINNING OF EACH PASS.
 6                              ;--
 7
 8                              ;;*****************************************************************************
 9                              ;    IF HERE FROM START COMMAND
10                              ;        THEN
11                              ;            SET ISTRT BIT & CLEAR OTHER BITS IN FLAG
12                              ;    ENDIF
13                              ;    IF HERE FROM RESTART COMMAND
14                              ;        THEN
15                              ;            SET IREST BIT IN IFLAGS
16                              ;    ENDIF
17                              ;    IF HERE FROM START OR RESTART COMMAND
18                              ;        THEN
19                              ;            RESET ALL UNITS
20                              ;            ESTABLISH FREE MEMORY
21                              ;            CLEAR TNUM
22                              ;            INITIALIZE CLOCK
23                              ;            BUILD CONTROLLER & DRIVES TABLES IN MEMORY
24                              ;            EXIT INIT SECTION
25                              ;    ENDIF
26                              ;    IF HERE FROM CONTINUE COMMAND
27                              ;        THEN
28                              ;            SET ICONT BIT IN IFLAGS
29                              ;            EXIT INIT SECTION
30                              ;    ENDIF
31                              ;    IF HERE FROM POWER FAIL RESTART
32                              ;        THEN
33                              ;            EXIT INIT SECTION
34                              ;    ENDIF
35                              ;    IF HERE FROM NEW PASS OR SUB-PASS
36                              ;        THEN
37                              ;            LOOK FOR ANY ADDED OR DROPPED UNITS
38                              ;            EXIT INIT SECTION
39                              ;    ENDIF
40                              ;;*****************************************************************************
```

M3

```
  1  107636                              L$INIT::
  2
  3  107636  012700  000040                       MOV    #EF.STA,R0        ;HERE FROM START COMMAND?
     107642  104447                               TRAP   C$REFG
  4                                                                        ;BRANCH TO 1$ IF NOT, ELSE
  5  107644  103004                               BCC    1$
  6  107646  012737  000010  064440               MOV    #ISTRT,IFLAGS     ;SET START BIT IN FLAG.
  7  107654  000531                               BR     INIT1
  8  107656                         1$:                                    ;HERE FROM RESTART COMMAND?
  9  107656  012700  000037                       MOV    #EF.RES,R0
     107662  104447                               TRAP   C$REFG
 10                                                                        ;BRANCH TO 2$ IF NOT, ELSE
 11  107664  103004                               BCC    2$
 12  107666  052737  000004  064440               BIS    #IREST,IFLAGS     ;SET RESTART BIT IN FLAG.
 13  107674  000521                               BR     INIT1
 14  107676                         2$:                                    ;HERE FROM CONTINUE COMMAND?
 15  107676  012700  000036                       MOV    #EF.CON,R0
     107702  104447                               TRAP   C$REFG
 16                                                                        ;BRANCH TO 3$ IF NOT, ELSE
 17  107704  103007                               BCC    3$
 18  107706  042737  000020  064440               BIC    #ISTRTH,IFLAGS    ;CLEAR 1ST TIME THRU TEST 4 FLAG AND
 19  107714  052737  000002  064440               BIS    #ICONT,IFLAGS     ;SET CONTINUE BIT IN FLAG.
 20  107722  000476                               BR     13$
 21  107724                         3$:                                    ;HERE FROM POWER FAIL?
 22  107724  012700  000034                       MOV    #EF.PWR,R0
     107730  104447                               TRAP   C$REFG
 23                                                                        ;BRANCH TO 4$ IF NOT, ELSE
 24  107732  103001                               BCC    4$
 25  107734  000471                               BR     13$
```

```
  1
  2                                    ;MAKE ALL CONTROLLER/DRIVE TABLES NOT AVAILABLE FOR TESTING
  3
  4 107736  013705  064424        4$:   MOV     CTABS,R5              ;GET ADDRESS OF 1ST CONTROLLER TABLE
  5 107742  052765  100000  000002 5$:   BIS     #CT.AVL,C.UNIT(R5)   ;SET CONTROLLER TABLE NOT AVAILABLE
  6 107750  010502                      MOV     R5,R2                ;GET POINTER TO DRIVE TABLES
  7 107752  062702  000020              ADD     #C.DR0,R2
  8 107756  012703  000010              MOV     #8.,R3               ;GET NUMBER OF DRIVES PER CONTROLLER TABLE
  9 107762  012200                6$:   MOV     (R2)+,R0             ;SEE IF THIS DRIVES HAS A TABLE.
 10 107764  001403                      BEQ     7$                   ;BRANCH IF NOT, ELSE
 11 107766  052760  100000  000002      BIS     #DT.AVL,D.UNIT(R0)   ;SET DRIVE TABLE NOT AVAILABLE.
 12 107774  005303                7$:   DEC     R3                   ;LOOK AT NEXT DRIVE IN CONTROLLER TABLE.
 13 107776  001371                      BNE     6$                   ;BRANCH IF NO DRIVES, ELSE
 14 110000  062705  000046              ADD     #C.SIZE,R5           ;LOOK AT NEXT CONTROLLER TABLE.
 15 110004  005715                      TST     (R5)                 ;SEE IF THERE IS ANOTHER CONTROLLER TABLE,
 16 110006  001012                      BNE     9$                   ;BRANCH IF SO, ELSE
 17 110010  062705  000046              ADD     #C.SIZE,R5           ;MOVE TO NEXT CONTROLLER TABLE
 18 110014  005715                      TST     (R5)                 ;IS THERE A NEXT ONE?
 19 110016  001351                      BNE     5$                   ;IF SO, CLEAR THE BITS THERE
 20
 21                                    ;NOW GET EACH P-TABLE AND MAKE THE APPROPRIATE CONTROLLER/DRIVE
 22                                    ;TABLES AVAILABLE FOR TESTING.
 23
 24 110020  005003                      CLR     R3                   ;START WITH LOGICAL UNIT 0
 25 110022                        8$:                                ;GET POINTER TO IT'S P TABLE
 26 110022  010300                      MOV     R3,R0
    110024  104442                      TRAP    C$GPHRD
 27                                                                  ;BRANCH TO 12$ IF NOT AVAILABLE
 28 110026  103030                      BCC     12$
 29 110030  013705  064424              MOV     CTABS,R5             ;GET ADDRESS OF 1ST CONTROLLER TABLE
 30 110034  021015                9$:   CMP     (R0),(R5)            ;SEE IF UDA ADDRESSES ARE THE SAME,
 31 110036  001411                      BEQ     11$                  ;BRANCH IF SO, ELSE
 32 110040  062705  000046              ADD     #C.SIZE,R5           ;LOOK AT NEXT CONTROLLER TABLE.
 33 110044  005715                      TST     (R5)                 ;SEE IF THERE IS ANOTHER CONTROLLER TABLE,
 34 110046  001372                      BNE     9$                   ;BRANCH IF SO, ELSE
 35 110050                       10$:                                ;REPORT TABLE CONSISTANCY ERROR.
 36 110050  104454                      TRAP    C$ERSF
    110052  000006                      .WORD   6
    110054  000000                      .WORD   0
    110056  074400                      .WORD   ERR006
 37                                                                  ;DO CLEAN-UP TRAP
 38 110060  104444                      TRAP    C$DCLN
 39
 40 110062  016001  000010       11$:   MOV     H.DRV(R0),R1         ;GET DRIVE NUMBER FROM P TABLE
 41 110066  004737  102274              CALL    GTDRVT               ;FIND THE DRIVE TABLE ADDRESS
 42 110072  001366                      BNE     10$                  ;BRANCH IF NOT FOUND, ELSE
 43 110074  042765  100000  000002      BIC     #CT.AVL,C.UNIT(R5)   ;CLEAR AVAILABLE BIT IN CONTROLLER AND
 44 110102  042764  100000  000002      BIC     #DT.AVL,D.UNIT(R4)   ;THE DRIVE TABLES.
 45 110110  005203                12$:   INC     R3                   ;INCREMENT TO NEXT UNIT IN P TABLE
 46 110112  020337  002012              CMP     R3,L$UNIT            ;SEE IF ALL P-TABLES CHECKED,
 47 110116  002741                      BLT     8$                   ;BRANCH IF NOT, ELSE
 48 110120  012701  064632       13$:   MOV     #STIME,R1            ;AT 15 MINUTES FROM NOW
 52 110124  012700  001604              MOV     #15.*60.,R0          ;SET TIME FOR NEXT REPORT
 53 110130  004737  104530              CALL    SETTO
 55 110134  000137  111170         :    JMP     INITXX               ;EXIT THE INITIALIZE SECTION.
```

B4

```
   1
   2                                        ;INITIALIZE KW11 CLOCK, FREE MEMORY AND IP ADDRESS TABLE
   3                                        ;DURING START OR RESTART COMMAND ONLY
   4
   5 110140  005037  064626     INIT1:  CLR     KW.EL               ;CLEAR ELAPSED TIME
   6 110144  005037  064630             CLR     KW.EL+2
   7 110150  012700  000114             MOV     #'L,R0
     110154  104462              '      TRAP    C$CLCK
   8 110156  103413                     BCS     1$
   9 110160  012700  000120             MOV     #'P,R0
     110164  104462                     TRAP    C$CLCK
  10 110166  103407                     BCS     1$
  11 110170  005037  064616             CLR     KW.CSR              ;IF NEITHER, CLEAR CSR STORAGE WORD
  12 110174  004137  075664             JSR     R1,LPNTF            ;CALL   LPNTF PRINT ROUTINE
     110200  066466                     .WORD   NOCLOCK             ;ADDRESS OF ASCIZ STRING
     110202  000000                     .WORD   ARG.CT              ;ARGUMENT COUNT = 2
  13 110204  000434                     BR      2$
  14
  15 110206  012037  064616     1$:     MOV     (R0)+,KW.CSR        ;STORE DATA RETURNED
  16 110212  012037  064620             MOV     (R0)+,KW.BRL
  17 110216  012037  064622             MOV     (R0)+,KW.VEC
  18 110222  012037  064624             MOV     (R0)+,KW.HZ
  19                                                                ;SETUP KW11 VECTOR ADDRESS
  20 110226  012746  000340             MOV     #PRIO7,-(SP)
     110232  012746  106230             MOV     #KW11I,-(SP)
     110236  013746  064622             MOV     KW.VEC,-(SP)
     110242  012746  000003             MOV     #3,(SP)
     110246  104437                     TRAP    C$SVEC
     110250  062706  000010             ADD     #10,SP
  21 110254  012777  000105  154334     MOV     #KW.OUT,@KW.CSR     ;START THE CLOCK
  22 110262  012701  064632             MOV     #STIME,R1           ;AT 15 MINUTES FROM NOW
  26 110266  012700  001604             MOV     #15.*60.,R0         ;SET TIME FOR NEXT REPORT
  27 110272  004737  104550             CALL    SETTO
  29 110276  004737  106260     2$:     CALL    RESET               ;RESET ALL UDA S
  30 110302  104431                     TRAP    C$MEM
     110304  010037  064412             MOV     R0,FFREE
  31 110310  017737  154076  064414     MOV     @FFREE,FSIZE        ;RESET SIZE OF FREE MEMORY
  32 110316  005037  064444             CLR     TNUM                ;INITIALIZE TEST NUMBER TO NO TEST RUNNING
  33 110322  005037  064442             CLR     FNUM                ;INITIALIZE FILE NUMBER TO NO FILE IN MEMORY
```

```
 1
 2                                    ;ALLOCATE DRIVE TABLES TO MEMORY
 3
 4 110326  013737  064412  064422  INIT2:  MOV    FFREE,DTABS      ;STORE START OF DRIVE TABLES AND
 5 110334  005077  154062                   CLR    @DTABS          ;MARK ZERO END.
 6 110340  013700  002012                   MOV    L#UNIT,R0       ;GET NUMBER OF LOGICAL UNITS TO RUN,
 7 110344  012701  000001                   MOV    #1,R1           ;GET INITIAL SIZE OF DRIVE TABLE AND
 8 110350  062701  000103           1$:     ADD    #<D.SIZE>/2,R1  ;ACCUMULATE DRIVE TABLE SIZE.
 9 110354  005300                           DEC    R0              ;SEE IF ANY MORE LOGICAL UNITS,
10 110356  001374                           BNE    1$              ;BRANCH IF NOT, ELSE
11 110360  004737  075422                   CALL   ALOCM           ;ALLOCATE ALL DRIVE TABLES TO MEMORY.
12                                                                 ; R1 POINTS TO 1ST WORD IN DRIVE TABLE
13
14                                    ;INITIALIZE CONTROLLER TABLE STORAGE WITH A WORD OF ZEROS
15
16 110364  013737  064412  064424  INIT3:  MOV FFREE,CTABS         ;STORE START OF CONTROLLER TABLES AND
17 110372  005077  154026                   CLR    @CTABS          ;MARK ZEROS END.
18 110376  005037  064426                   CLR    CTRLRS          ;CLEAR CONTROLLER COUNT
19 110402  012701  064534                   MOV    #IPADRS,R1      ; R1 -> IP ADDRESS
20 110406  012702  000010                   MOV    #8.,R2          ; R2 IS A COUNTER
21 110412  005021                   1$:     CLR    (R1)+           ; CLEAR ENTRY
22 110414  005302                           DEC    R2              ; DONE?
23 110416  001375                           BNE    1$              ; IF NOT, BRANCH
```

```
 1
 2                                        ;BUILD CONTROLLER TABLES
 3
 4 110420  005005              INIT4:  CLR     R5              ;CLEAR CUSTOMER DATA FLAG
 5 110422  005002                      CLR     R2              ;START WITH LOGICAL UNIT 0
 6 110424                      1$:                             ;GET POINTER TO IT'S P-TABLE
 7 110424  010200                      MOV     R2,R0
   110426  104442                      TRAP    C$GPHRD
 8
                                                               ;BRANCH TO 16$ IF NOT AVAILABLE
 9 110430  103156                      BCC     16$
10 110432  013703  064424             MOV     CTABS,R3        ;GET ADDRESS OF 1ST CONTROLLER TABLE
11 110436  005713              2$:     TST     (R3)            ;CHECK IF ANY MORE TABLES
12 110440  001435                      BEQ     6$              ;BUILD NEW TABLE IF FOUND ZERO WORD
13 110442  021013                      CMP     (R0),(R3)       ;CHECK IF SAME UNIBUS ADDRESS.
16 110444  001017                      BNE     4$              ;BRANCH IF NOT, ELSE
17                                                             ;CHECK THAT OTHER PARAMETERS MATCH.
18 110446  016004  000004             MOV     H.BRL(R0),R4    ;GET BR LEVEL FROM P TABLE
19 110452  000304                      SWAB    R4              ;SWAP TO HIGH BYTE
20 110454  006104                      ROL     R4              ;SHIFT ONE MORE TO LEFT
21 110456  056004  000002             BIS     H.VEC(R0),R4    ;ADD VECTOR ADDRESS
22 110462  020463  000004             CMP     R4,C.VEC(R3)    ;COMPARE VECTOR AND BR LEVELS.
23 110466  001004                      BNE     3$              ;BRANCH IF DIFFERENT. ELSE
24 110470  026063  000006  000006     CMP     H.BST(R0),C.BST(R3)  ;COMPARE BURST RATES.
25 110476  001461                      BEQ     11$             ;BRANCH IF SAME. ELSE
26 110500  000137  111212     3$:     JMP     CTABER          ;FOUND SAME UDA WITH DIFFERENT
27                                                             ;BR LEVEL, VECTOR ADDRS OR BURST RATE.
28 110504  016304  000004     4$:     MOV     C.VEC(R3),R4    ;GET VECTOR FROM CONTROLLER TABLE
29 110510  042704  177000             BIC     #^C<CT.VEC>,R4  ;AND
30 110514  026004  000002             CMP     H.VEC(R0),R4    ;COMPARE VECTOR ADDRESSES.
31 110520  001002                      BNE     5$              ;BRANCH IF DIFFERENT. ELSE
32 110522  000137  111262             JMP     SAMVEC          ;FOUND TWO UDA S WITH SAME VECTOR ADDRESS.
33
34 110526  062703  000046     5$:     ADD     #C.SIZE,R3      ;POINT TO BEGINNING OF NEXT CONTROLLER
35 110532  000741                      BR      2$              ;TABLE IN MEMORY.
```

F4

```
  1
  2                                   ;BUILD NEW CONTROLLER TABLE
  3
  4 110534  012704  064534    6$:    MOV    #IPADRS,R4              ;GET BEGINNING OF IP ADDRESS TABLE
  5 110540  020427  064554    7$:    CMP    R4,#IPADRS+16.         ;SEE IF END OF IP ADDRESS TABLE.
  6 110544  101004                   BHI    9$                     ;BRANCH IF SO. ELSE
  7 110546  005724                   TST    (R4)+                  ;DID WE FIND AN OPEN ENTRY ?
  8 110550  001401                   BEQ    8$                     ;BRANCH IF SO, ELSE
  9 110552  000772                   BR     7$                     ;LOOK AGAIN.
 10
 11 110554  011044            8$:    MOV    (R0),-(R4)             ;TAKE UNIBUS ADDRESS FROM P-TABLE
 12                                                                ;AND STORE IT IN THE IP ADDRESS TABLE.
 13 110556  012701  000023    9$:    MOV    #<C.SIZE>/2,R1         ;GET # OF ENTRIES IN CONTROLLER TABLE
 14 110562  004737  075422           CALL   ALOCM                  ;AND ALLOCATE A TABLE TO MEMORY.
 15                                                                ; R0 POINTS TO 1ST WORD P-TABLE
 16                                                                ; R1 POINTS TO 1ST WORD IN CONTROLLER TABLE
 17 110566  011021                   MOV    (R0),(R1)+             ;STORE UDA IP ADDRESS AND
 18 110570  010221                   MOV    R2,(R1)+              ;LOGICAL UNIT NUMBER IN THE CONTROLLER TABLE.
 19 110572  016004  000004           MOV    H.BRL(R0),R4          ;GET THE BR LEVEL.
 20 110576  000304                   SWAB   R4                     ;SWAP TO HIGH BYTE.
 21 110600  006104                   ROL    R4                     ;SHIFT ONE MORE TO LEFT.
 22 110602  056004  000002           BIS    H.VEC(R0),R4          ;ADD VECTOR ADDRESS AND
 23 110606  010421                   MOV    R4,(R1)+             ;STORE IT IN THE CONTROLLER TABLE.
 24 110610  016021  000006           MOV    H.BST(R0),(R1)+      ;STORE THE BURST RATE.
 25 110614  012721  004037           MOV    #4037,(R1)+          ;THE 'JSR R0' INSTRUCION AND
 26 110620  012721  104520           MOV    #UDASRV,(R1)+        ;THE ADDRESS OF THE INTERRUPT SERVICE
 27                                                                ;ROUTINE IN THE CONTROLLER TABLE.
 28 110624  012704  000015           MOV    #<C.SIZE-C.FLG>/2,R4 ;GET # OF ENTRIES TO END OF TABLE.
 29 110630  005021           10$:    CLR    (R1)+                ;CLEAR REST OF TABLE AND
 30 110632  005304                   DEC    R4                    ;ADD ZERO WORD AT END.
 31 110634  002375                   BGE    10$                   ;LOOP TIL ALL CLEARED
 32 110636  005237  064426           INC    CTRLRS                ;KEEP TRACK OF CONTROLLER COUNT
```

41

F.4

```
 1
 2                                      ;BUILD DRIVE TABLES
 3
 4 110642  013701  064422        11$:   MOV    DTABS,R1              ;GET ADDRESS OF CURRENT DRIVE TABLE
 5 110646  062703  000020               ADD    #C.DR0,R3            ;INDEX TO 1ST DRIVE IN CONTROLLER TABLE
 6 110652  012704  000010               MOV    #8.,R4               ;GET MAXIMUM # OF DRIVES PER CONTROLLER
 7 110656  005713                12$:   TST    (R3)                 ;ANY ENTRY TO DRIVE TABLE,
 8 110660  001411                       BEQ    14$                  ;BRANCH IF NOT, ELSE
 9 110662  026033  000010               CMP    H.DRV(R0),@(R3)+     ;COMPARE DRIVE NUMBER IN DRIVE TABLE,
10 110666  001002                       BNE    13$                  ;BRANCH IF DIFFERENT, ELSE
11 110670  000137  111226               JMP    MLDRER               ;FOUND TWO P TABLES WITH SAME DRIVE.
12
13 110674  005304                13$:   DEC    R4                   ;COUNT DRIVES
14 110676  001367                       BNE    12$                  ;IF EIGHT DRIVE TABLES EXIST,
15 110700  000137  111244               JMP    TOOMER               ;   THEN REPORT ERROR
16
17 110704  010113                14$:   MOV    R1,(R3)              ;STORE ADDRESS OF DRIVE TABLE IN
18                                                                   ;CONTROLLER TABLE.
19 110706  016021  000010               MOV    H.DRV(R0),(R1)+      ;STORE DRIVE NUMBER AND
20 110712  010221                       MOV    R2,(R1)+             ;LOGICAL UNIT NUMBER IN DRIVE TABLE.
21 110714  016011  000012               MOV    H.PRM(R0),(R1)       ;GET TEST AREA BIT
22 110720  051105                       BIS    (R1),R5              ;SAVE "OR" OF BIT FROM ALL DRIVES
23 110722  005111                       COM    (R1)                 ;COMPLIMENT IT
24 110724  042711  157777               BIC    #^C<HM.CYL>,(R1)
25 110730  052721  011012               BIS    #ODEF,(R1)+          ;LOAD DEFAULT PARAMETER BITS
26 110734  012704  000100               MOV    #<D.SIZE/2>-3,R4     ;CLEAR REST OF TABLE
27 110740  005021                15$:   CLR    (R1)+
28 110742  005304                       DEC    R4
29 110744  003375                       BGT    15$
30 110746  012761  177777  177754       MOV    #-1,<D.ECYL+2-D.SIZE>(R1) ;MARK CYLINDERS AT TEST ALL
31
32 110754  062737  000206  064422       ADD    #D.SIZE,DTABS        ;NEXT DRIVE TABLE ADDRESS AND
33 110762  005077  153434               CLR    @DTABS               ;MARK ZERO END.
34 110766  005202                16$:   INC    R2                   ;INCREMENT LOGICAL UNIT NUMBER
35 110770  020237  002012               CMP    P2,L$UNIT            ;CHECK IF GOT ALL TABLES
36 110774  002613                       BLT    1$                   ;IF NOT, GO BACK FOR NEXT, ELSE
37 110776  012701  000001               MOV    #1,R1                ;GET 1 WORD TO TERMINATE ALL CONTROLLER
38 111002  004737  075422               CALL   ALOCM                ;TABLES AND ALLOCATE IT TO MEMORY.
```

```
  1
  2                                        ;CHECK FOR CUSTOMER WARNING MESSAGE
  3
  4 111006  032705  020000      INIT5:  BIT     #HM.CYL.,R5          ;CHECK IF BIT EVER SET
  5 111012  001460                      BEQ     5$                  ;BYPASS IF NOT
  6 111014  004137  075664              JSR     R1,LPNTF            ;CALL    LPNTF PRINT ROUTINE
    111020  065454                      .WORD   INITWA              ;ADDRESS OF ASCIZ STRING
    111022  000000                      .WORD   ARG.CT              ;ARGUMENT COUNT * 2
  7 111024  013705  064424              MOV     CTABS,R5            ;GET ADDRESS 1ST CONTROLLER TABLE
  8 111030  010504              1$:     MOV     R5,R4               ;GET ADDRESS OF POINTER TO DRIVE TABLE
  9 111032  062704  000020              ADD     #C.DR0,R4
 10 111036  012701  000010              MOV     #8.,R1              ;GET COUNT OF DRIVE TABLES
 11 111042  012403              2$:     MOV     (R4)+,R3            ;GET ADDRESS OF DRIVE TABLE
 12 111044  001422                      BEQ     4$
 13 111046  032763  020000  000004      BIT     #D.DCY,D.PRM(R3)    ;CHECK IF CUSTOMER DATA SELECTED
 14 111054  001014                      BNE     3$
 15 111056  011346                      MOV     (R3),-(SP)
    111060  011546                      MOV     (R5),-(SP)
    111062  016346  000002              MOV     D.UNIT(R3),-(SP)
    111066  012746  065560              MOV     #INITWB,-(SP)
    111072  012746  000004              MOV     #4,-(SP)
    111076  010600                      MOV     SP,R0
    111100  104417                      TRAP    C$PNTF
    111102  062706  000012              ADD     #12.SP
 16 111106  005301              3$:     DEC     R1                  ;COUNT THE DRIVE TABLES
 17 111110  001354                      BNE     2$                  ;LOOK AT ALL OF THEM
 18 111112  062705  000046      4$:     ADD     #C.SIZE,R5          ;MOVE TO NEXT CONTROLLER TABLE
 19 111116  005715                      TST     (R5)                ;SEE IF ANOTHER TABLE AND
 20 111120  001343                      BNE     1$                  ; LOOK AT IT
 21
 23                                      ;GET CONFIRMATION TO PROCEED
 24
 25 111122  104450                      TRAP    C$MANI
 26 111124  103013                      BCC     5$
 27 111126  104443                      TRAP    C$GMAN
    111130  000404                      BR      10000$
    111132  064476                      .WORD   TEMP
    111134  000120                      .WORD   T$CODE
    111136  064763                      .WORD   INITWC
    111140  000001                      .WORD   1
    111142                      10000$:
 28 111142  032737  000001  064476      BIT     #1,TEMP             ;LOOK AT RESPONSE
 29 111150  001001                      BNE     5$                  ;BRANCH IF YES WAS ANSWER
 30                                                                 ;DO CLEAN-UP TRAP
 31 111152  104444                      TRAP    C$DCLN
 33
 34                                      ;SAVE CURRENT PARAMETERS TO FREE MEMORY SO EACH TEST CAN USE ALL OF IT
 35
 36 111154  013737  064412  064416  5$: MOV     FFREE,FMEM          ;SAVE START ADDRESS
 37 111162  013737  064414  064420      MOV     FSIZE,FMEMS         ;SAVE SIZE
```

H4

```
   1
   2                                      ;EXIT INITIALIZE SECTION
   3
   4 111170                      INITXX:
     111170  012700  000000              MOV     #PRIOO,RO
     111174  104441                      TRAP    C$SPRI
   5 111176  005037  064656              CLR     DLL             ;ERASE DOWNLINE LOAD DATA
   6 111202  004737  105612              CALL    CLOSEF          ;MAKE SURE DATA FILE IS CLOSED
  30
  31 111206  104432                      TRAP    C$EXIT
     111210  000066                      .WORD   L10044-.
```

I4

```
    1                                    .SBTTL   INITIALIZE ERRORS
    2
    3                                             ;DIFFERENT VECTORS, BR LEVELS OR BURST RATES FOR ONE CONTROLLER
    4
    5 111212  010305               CTABER:  MOV     R3,R5              ;GET CONTROLLER ADDRESS
    6 111214  104454                        TRAP    C$ERSF
      111216  000001                        .WORD   1
      111220  000000                        .WORD   0
      111222  074252                        .WORD   ERR001
    7                                                                   ;DO CLEAN UP TRAP
    8 111224  104444                        TRAP    C$DCLN
    9
   10                                            ;TWO P-TABLES FOR SAME DRIVE
   11
   12 111226  013705  064476       MLDRER:  MOV     TEMP,R5            ;GET CONTROLLER ADDRESS
   13 111232  104454                        TRAP    C$ERSF
      111234  000002                        .WORD   2
      111236  000000                        .WORD   0
      111240  074270                        .WORD   ERR002
   14                                                                   ;DO CLEAN-UP TRAP
   15 111242  104444                        TRAP    C$DCLN
   16
   17                                            ;MORE THAN EIGHT DRIVES SELECTED ON ONE CONTROLLER
   18
   19 111244  013705  064476       TOOMER:  MOV     TEMP,R5            ;GET CONTROLLER ADDRESS
   20 111250  104454                        TRAP    C$FRSF
      111252  000003                        .WORD   3
      111254  000000                        .WORD   0
      111256  074306                        .WORD   FRR003
   21                                                                   ;DO CLEAN UP TRAP
   22 111260  104444                        TRAP    C$DCLN
   23
   24                                            ;TWO UDA'S USE THE SAME VECTOR
   25
   26 111262  010305               SAMVEC:  MOV     R3,R5              ;GET CONTROLLER ADDRESS
   2` 111264  104454                        TRAP    C$ERSF
      111266  000010                        .WORD   8
      111270  000000                        .WORD   0
      111272  074412                        .WORD   ERR008
   28                                                                   ;DO CLEAN UP TRAP
   29 111274  104444                        TRAP    C$DCLN
   41                                            .EVEN
   42
   43 111276                       L10044:
      111276  104411                        TRAP    C$INIT
```

J4

```
 1                                    .SBTTL   AUTODROP SECTION
 2
 3                               ;**
 4                               ; THIS CODE IS EXECUTED IMMEDIATELY AFTER THE INITIALIZE CODE IF
 5                               ; THE "ADR" FLAG WAS SET.  THE UNIT(S) UNDER TEST ARE CHECKED TO
 6                               ; SEE IF THEY WILL RESPOND.  THOSE THAT DON'T ARE IMMEDIATELY
 7                               ; DROPPED FROM TESTING.
 8                               ; --
 9
10 111300                        L$AUTO::
11
18 111300                        L10045:
   111300  104461                         TRAP     C$AUTO
```

K4

```
 1                                      .SBTTL   CLEANUP CODING SECTION
 2
 3                                 ;++
 4                                 ; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
 5                                 ; AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.
 6                                 ;-
 7
 8 111302                         L$CLEAN::
 9
10 111302  004737  105612                 CALL     CLOSEF          ;CLOSE DATA FILE
12 111306  022737  000004  064444          CMP      #4,TNUM         ;ARE WE DOING TEST 4 ?
13 111314  001402                          BEQ      1$              ;BRANCH IF SO, DON'T RESET BUS
16 111316  004737  106260                  CALL     RESET           ;RESET ALL UDA'S
17 111322                         1$:
27
28 111322  104432                          TRAP     C$EXIT
   111324  000002                          .WORD    L10046-.
29
41                                         .EVEN
42
43 111326                         L10046:
   111326  104412                          TRAP     C$CLEAN
```

L4

```
     1                              .SBTTL   DROP UNIT SECTION
     2
     3                              ;**
     4                              ; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
     5                              ; TO NO LONGER BE TESTED.
     6                              ;--
     7
     8  111330                      L$DU::
     9
    18
    19  111330  000167                      .WORD    J$JMP
        111332  000000                      .WORD    L10047-2 .
    20
    32                                      .EVEN
    33
    34  111334                      L10047:
        111334  104453                      TRAP     C$DU
```

```
      1                              .SBTTL  ADD     UNIT SECTION
      2
      3                         ; **
      4                         ; THE ADD-UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
      5                         ;  TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
      6                         ;  TO THE TEST CYCLE.
      7                         ; -
      8
      9 111336            .     L$AU::
     10
     19
     20 111336  000167                  .WORD   J$JMP
        111340  000000                  .WORD   L10050-2-.
     21
     33                                  .EVEN
     34
     35 111342           L10050:
        111342  104452                   TRAP   C$AU
```

```
   2                                       .SBTTL   HARDWARE TESTS
  14                                       .SBTTL   TEST 1: UNIBUS ADDRESSING TEST
  51
  54 111344                       T1::
  59 111344  012701  000001                MOV      #1,R1              ;INITIALIZE TEST PARAMETERS
  60 111350  004737  076336                CALL     TINIT
  61 111354  013737  064424  064430        MOV      CTABS,TSTTAB       ;GET ADDRESS OF 1ST CONTROLLER TABLE
  62 111362  013705  064430      T1NEXT:   MOV      TSTTAB,R5          ; GET CONTROLLER TABLE ADDRESS
  63 111366  116537  000002  002074        MOVB     C.UNIT(R5),L$LUN   ; CHECK IF UNIT AVAILABLE FOR TESTING
  64 111374  005765  000002             TST      C.UNIT(R5)
  65 111400  100010                       BPL      T1NOW              ; TEST IF AVAILABLE
  67 111402  062737  000046  064430  T1SKIP: ADD    #C.SIZE,TSTTAB     ; MOVE TO NEXT CONTROLLER
  68 111410  005777  153014                TST      @TSTTAB            ; CHECK IF ANOTHER CONTROLLFR TABLE
  69 111414  001362                        BNE      T1NEXT
  70 111416  104432                        TRAP     C$EXIT
     111420  000776                        .WORD    L10051-.
  71
  72 111422  004737  106260      T1NOW:    CALL     RESET              ;RESET ALL UDA'S
```

```
     1
     2 111426                          T1.1.
       111426  104402                          TRAP     C$BSUB
     3 111430  005037  064636                  CLR      NXMAD          ;CLEAR MEMORY ERROR FLAG
     4                                                                 ;SETUP TIMEOUT ERROR VECTOR
     5 111434  012746  000340                  MOV      @PRIO7, (SP)
       111440  012746  104510                  MOV      @NXMI, (SP)
       111444  012746  000004                  MOV      @ERRVEC, (SP)
       111450  012746  000003                  MOV      #3, (SP)
       111454  104437                          TRAP     C$SVEC
       111456  062706  000010                  ADD      #10,SP
     6 111462  011504                          MOV      (R5),R4        ;GET ADDRESS OF UDAIP REGISTER
     7 111464  005714                          TST      (R4)           ;READ UDAIP
     8 111466  005764  000002                  TST      2(R4)          ;READ UDASA
     9                                                                 ;RETURN TIMEOUT ERROR VECTOR
    10 111472  012700  000004                  MOV      @ERRVEC,R0
       111476  104436                          TRAP     C$CVEC
    11 111500  005737  064636                  TST      NXMAD          ;CHECK FLAG
    12 111504  001406                          BEQ      T1GOOD
    13 111506  104455                          TRAP     C$ERDF
       111510  000046                          .WORD    38
       111512  000000                          .WORD    0
       111514  075044                          .WORD    ERRO38
    14 111516  104406                          TRAP     C$CLP1
    15 111520  000730                          BR       T1SKIP         ;END TEST NOW
    16 111522                          T1GOOD:
    17 111522                          L10052:
       111522  104403                          TRAP     C$ESUB
```

```
        1
        2                                  ;
        3                                  ;       MAKE SURE UDA PASSES INTERNAL DIAGNOSTIC
        4                                  ;       MAKE SURE UDA CAN SENSE STEP 1 AND 2
        5                                  ;
        6
        7 111524                          T1.2:
          111524  104402                          TRAP    C$BSUB
        8 111526  005014                          CLR     (R4)                ; INIT UDA
        9 111530  012737  004000  105546          MOV     #SA.S1,UDARSD       ; STEP 1 ASSERTED?
       10 111536  004737  105410                  CALL    UDARSP              ; WAIT FOR RESPONSE
       11 111542  103410                          BCS     1$                  ; IF FAIL, EXIT
       12 111544  012764  100000  000002          MOV     #SA.STP,2(R4)       ; SEND STEP 1
       13 111552  012737  010000  105546          MOV     #SA.S2,UDARSD       ; STEP 2 ASSERTED?
       14 111560  004737  105410                  CALL    UDARSP
       15 111564                          1$:
       16 111564                          L10053:
          111564  104403                          TRAP    C$ESUB
```

```
     1
     2                                     ;
     3                                     ;       TEST THE DIAGNOSTIC LOOP MODE OF ALL UDA S ON THE SYSTEM
     4                                     ;
     5
     6 111566                     T1.3:
       111566 104402                        TRAP     C$BSUB
     7 111570 011504                        MOV      (R5),R4                   ; R4 POINTS TO UDAIP REGISTER
     9 111572 005014                        CLR      (R4)                      ; INITIALIZE THE UDA
    10 111574 012737 004000 105546          MOV      #SA.S1,UDARSD             ; LOOK FOR STEP 1
    11 111602 004737 105410                 CALL     UDARSP                    ; WAIT FOR RESPONSE
    12 111606 103444                        BCS      5$                        ; IF ERROR, BRANCH
    13 111610 016437 000002 106650          MOV      2(R4),WCHNGD              ; MOVE OLD PORT CONTENTS TO STORAGE
    14 111616 012764 140000 000002          MOV      #<SA.STP+SA.WRP>,2(R4)    ; INITIALIZE FOR PORT WRAP
    15 111624 004737 106554                 CALL     WCHNG                     ; WAIT FOR THE PORT TO CHANGE
    16 111630 001433                        BEQ      5$                        ; IF ERROR, BRANCH
    17 111632 022764 140000 000002          CMP      #<SA.STP+SA.WRP>,2(R4)    ; COMPARE WITH DATA WRITTEN
    18 111640 001017                        BNE      3$
    19 111642 012702 000001           1$:   MOV      #1,R2                     ; SET UP FOR SHIFTING '1'
    20 111646 012703 000020                 MOV      #16.,R3                   ; SET UP LOOP COUNT
    21 111652 016437 000002 106650    2$:   MOV      2(R4),WCHNGD              ; SAVE OLD PORT CONTENTS
    22 111660 010264 000002                 MOV      R2,2(R4)                  ; WRITE PATTERN TO UDASA FOR LOOP
    23 111664 004737 106554                 CALL     WCHNG                     ; WAIT FOR UDASA TO CHANGE
    24 111670 001413                        BEQ      5$                        ; IF ERROR, BRANCH
    25 111672 020264 000002                 CMP      R2,2(R4)                  ; COMPARE RO WITH WHAT WAS ECHOED
    26 111676 001405                        BEQ      4$                        ; IF MATCH, BRANCH
    27 111700                          3$:
       111700 104455                        TRAP     C$ERDF
       111702 000032                        .WORD    26
       111704 000000                        .WORD    0
       111706 074634                        .WORD    ERRO26
    28 111710 000403                        BR       5$                        ; BRANCH
    29 111712 006302                   4$:   ASL      R2                        ; MOVE THE SHIFTING ONE LEFT BY 1
    30 111714 005303                        DEC      R3                        ; DECREMENT COUNT
    31 111716 001355                        BNE      2$                        ; IF LOOP INCOMPLETE, BRANCH
    32 111720                          5$:
    33 111720                     L10054:
       111720 .403                         TRAP     C$ESJB
```

E5

```
   1
   2                                      ;
   3                                      ;      TEST THE INTERRUPTS VECTOR AND BR LEVEL
   4                                      ;
   5
   6 111722                       T1.4:
     111722  104402                        TRAP    C$BSUB
   7 111724  011504                        MOV     (R5),R4          ; R4 POINTS TO UDAIP REGISTER
   9 111726  016503  000004                MOV     C.VEC(R5),R3     ; GET VECTOR AND BRANCH LEVEL
  10 111732  010302                        MOV     R3,R2            ; COPY TO R2 FOR BR LEVEL
  11 111734  042703  177000                BIC     #^CCT.VEC,R3     ; CLEAR UNUSED VECTOR BITS
  12 111740  042702  170777                BIC     #^CCT.BRL,R2     ; CLEAR UNUSED BRANCH LEVEL BITS
  13 111744  012701  000011                MOV     #9.,R1           ; SET UP TO SHIFT BR LEVEL
  14 111750  006202               1$:      ASR     R2               ; SHIFT BY ONE BIT
  15 111752  005301                        DEC     R1               ; COUNT SHIFTS
  16 111754  001375                        BNE     1$               ; IF INCOMPLETE, BRANCH
  17 111756  010237  106652                MOV     R2,BRLEV         ; SAVE THE BRANCH LEVEL
  18 111762  010346                        MOV     R3,-(SP)         ;PUSH R3 ON STACK
     111764  011546                        MOV     (R5),-(SP)       ;PUSH (R5) ON STACK
     111766  004137  075704                JSR     R1,LPNTX         ;CALL    LPNTX PRINT ROUTINE
     111772  065342                        .WORD   INTSTO           ;ADDRESS OF ASCIZ STRING
     111774  000004                        .WORD   ARG.CT           ;ARGUMENT COUNT + 2
  20                                                                ; SETUP INTERRUPT VECTOR ADDRESS
  21 111776  012746  000000                MOV     #PRIOO,-(SP)
     112002  012746  106252                MOV     #INTSRV,-(SP)
     112006  010346                        MOV     R3,-(SP)
     112010  012746  000003                MOV     #3,(SP)
     112014  104437                        TRAP    C$SVEC
     112016  062706  000010                ADD     #10,SP
  22 112022  012700  000000                MOV     #PRIOO,R0
     112026  104441                        TRAP    C$SPRI
  23 112030  006203                        ASR     R3               ; DIVIDE VECTOR BY 4 FOR UDA INITIALIZATION
  24 112032  006203                        ASR     R3               ; DIVIDE VECTOR BY 4 FOR UDA INITIALIZATION
  25 112034  052703  100200                BIS     #<SA STP+SA.INT>,R3 ; SET OTHER BITS FOR UDA INITIALIZATION
  26 112040  005037  064454                CLR     INTR(V           ; FLAG AS NO INTERRUPTS RECEIVED
  27 112044  005014                        CLR     (R4)             ; INIT UDA
  28 112046  012737  004000  105546        MOV     #SA.S1,UDARSD    ; LOOK FOR STEP 1 COMPLETION
  29 112054  004737  105410                CALL    UDARSP           ; WAIT FOR COMPLETION
  30 112060  010364  000002                MOV     R3,2(R4)         ; MOVE STEP 1 DATA TO UDA
  31 112064  012700  000012                MOV     #10.,R0          ;SET UP TIMEOUT OF 10 SECONDS
  32 112070  010501                        MOV     R5,R1
  33 112072  062701  000040                ADD     #C.TO,R1         ;POINT TO CONTROLLER TABLE
  34 112076  004737  104530                CALL    SETTO
  35 112102  005737  064454      2$:       TST     INTRCV           ; SEE IF INTERRUPTED
  36 112106  001016                        BNE     3$               ; IF SO, EVERYTHING'S OK, SO BRANCH
  37                                                                ;>>>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<<
  38 112110  104422                        TRAP    C$BRK
  39
  40 112112  005737  064616                TST     KW.CSR           ;SEE IF CLOCK ON SYSTEM
  41 112116  001771                        BEQ     2$
  42 112120  023765  064630  000042        CMP     KW.EL+2,C.TOH(R5) ;SEE IF TIME ELAPSED
  43 112126  101041                        BHI  7$
  44 112130  001364                        BNE     2$
  45 112132  023765  064626  000040        CMP     KW.EL,C.TO(R5)
  46 112140  103760                        BLO 2$
  47 112142  000433                        BR      7$               ; BRANCH
  48 112144  005037  064454      3$:       CLR     INTRCV           ; FLAG AS NO INTERRUPTS RECEIVED
```

F5

```
49 112150   012700  000340              MOV     #PRIO7,RO
   112154   104441                       TRAP    C$SPRI
50 112156   005064  000002              CLR     2(R4)           ; WRITE SECOND STEP TO UDA
51 112162   012702  000144              MOV     #100.,R2        ; SET UP DELAY SO WE KNOW WE'RE INTERRUPTED
52 112166   005302            4$:       DEC     R2              ; DECREMENT COUNT
53 112170   001376                       BNE     4$              ; IF INCOMPLETE, BRANCH
54 112172   012701  000007              MOV     #7.,R1          ; R1 IS PROCESS PRIORITY LEVEL
55 112176                     5$:
   112176   010146                       MOV     R1,-(SP)        ;;PUSH R1 ON STACK
56 112200   012702  000005              MOV     #5.,R2          ; SET UP FOR SHIFTING PRIORITY
57 112204   006301            6$:       ASL     R1              ; SHIFT PRIORITY
58 112206   005302                       DEC     R2              ; DECREMENT SHIFT COUNT
59 112210   001375                       BNE     6$              ; IF INCOMPLETE, BRANCH
60 112212   010100                       MOV     R1,RO
   112214   104441                       TRAP    C$SPRI
61 112216   012601                       MOV     (SP)+,R1        ;;POP STACK INTO R1
62 112220   005737  064454              TST     INTRCV          ; SEE IF INTERRUPT RECEIVED
63 112224   001007                       BNE     8$              ; IF SO, BRANCH
64 112226   005301                       DEC     R1              ; DECREMENT PRIORITY LEVEL
65 112230   100362                       BPL     5$              ; IF ALL LEVELS UNTESTED, BRANCH
66 112232                     7$:
   112232   104455                       TRAP    C$ERDF
   112234   000034                       .WORD   28
   112236   000000                       .WORD   0
   112240   074672                       .WORD   ERRO28
67 112242   000420                       BR      10$             ; BRANCH
68
69 112244                     8$:
   112244   012700  000000              MOV     #PRIO0,RO
   112250   104441                       TRAP    C$SPRI
70 112252   005201                       INC     R1              ; SO PRIORITY = BR LEVEL
71 112254   023701  106652              CMP     BRLEV,R1        ; SEE IF BR LEVEL MATCHES PRIORITY
72 112260   001405                       BEQ     9$              ; IF SO, BRANCH
73 112262   104455                       TRAP    C$ERDF
   112264   000035                       .WORD   29
   112266   000000                       .WORD   0
   112270   074704                       .WORD   ERRO29
74 112272   000404                       BR      10$             ; BRANCH
75 112274                     9$:
   112274   004137  075704              JSR     R1,LPNTX        ;CALL   LPNTX PRINT ROUTINE
   112300   065437                       .WORD   INTST1          ;ADDRESS OF ASCIZ STRING
   112302   000000                       .WORD   ARG.CT          ;ARGUMENT COUNT = 2
76 112304   016503  000004    10$:      MOV     C.VEC(R5),R3    ; GET VECTOR ADDRESS
77 112310   042703  177000              BIC     #^CCT.VEC,R3    ; CLEAR UNUSED BITS
78 112314   010300                       MOV     R3,RO
   112316   104436                       TRAP    C$CVEC
79 112320                     L10055:
   112320   104403                       TRAP    C$ESUB
```

G5

```
    1
    2 112322                          T1.5:
      112322  104402                          TRAP     C$BSUB
    3 112324  005004                          CLR      R4              ; INITIALIZE UDA WITH SMALLEST
    4 112326  004737  104612                  CALL     UDAINT          ; RING BUFFER AND INTERRUPTS DISABLED
    5 112332                          L10056:
      112332  104403                          TRAP     C$ESUB
```

H5

```
  1
  2 112334                          T1.6:          .
    112334  104402                          TRAP    C$BSUB
  3 112336  012704  126400                  MOV     #<SA.STP+<5*SA.MS1>+<5*SA.CM1>>,R4  ;INITIALIZE UDA WITH RING BUFFER
  4 112342  004737  104612                  CALL    UDAINT                             ;LARGE ENOUGH TO COVER NORMAL
  5                                                                                    ;HOST COMM AREA PACKET AND BUFFER
  6                                                                                    ;SPACE (A 5 IN MESSAGE LENGTH AND
  7                                                                                    ;A 5 IN COMMAND LENGTH)
  8 112346                          L10057:
    112346  104403                          TRAP    C$ESUB
```

```
   1
   2 112350                        T1.7:
     112350   104402                        TRAP      C$BSUB
   3 112352   013746   064412               MOV       FFREE,(SP)        ;;PUSH FFREE ON STACK
     112356   013746   064414               MOV       FSIZE,-(SP)       ;;PUSH FSIZE ON STACK
   4 112362   012701   000001               MOV       #1,R1             ; RUN DM PROGRAM IN
   5 112366   004737   076532               CALL      RUNDM             ; ONE CONTROLLER ONLY
   6 112372   001402                         BEQ       1$
   7 112374   004737   076626               CALL      RESPDM
   8 112400                        1$:
     112400   012637   064414               MOV       (SP)+,FSIZE       ;;POP STACK INTO FSIZE
     112404   012637   064412               MOV       (SP)+,FFREE       ;;POP STACK INTO FFREE
   9 112410                        L10060:
     112410   104403                        TRAP      C$ESUB
  10 112412   000137   111402               JMP       T1SKIP
  27                                         .EVEN
  28
  29 112416                        L10051:
     112416   104401                        TRAP      C$ETST
  30
```

J5

CZUOCEO UDA & DISK DRV DIAG MACRO V05.00 Wednesday 04 Jan 84 16:12 Page 211   SFQ 0268
TEST 2: DISK RESIDENT DIAGNOSTIC TEST

```
         1                                     .SBTTL   TEST 2: DISK RESIDENT DIAGNOSTIC TEST
         2
         3 112420                     T2::
         8 112420  012701  000002               MOV      #2,R1               ;INITIALIZE TEST PARAMETERS
         9 112424  004737  076336               CALL     TINIT
        10 112430  013737  064424  064430       MOV      CTABS,TSTTAB        ;GET ADDRESS TO 1ST CONTROLLER TABLE
        11
        12 112436  004737  106260      1$:      CALL     RESET               ;RESET ALL UDA'S
        13 112442  013746  064412               MOV      FFREE,-(SP)         ;;PUSH FFREE ON STACK
           112446  013746  064414               MOV      FSIZE,-(SP)         ;;PUSH FSIZE ON STACK
        14 112452  012701  000001               MOV      #1,R1               ;RUN DM PROGRAM IN
        15 112456  004737  076532               CALL     RUNDM               ; ONE CONTROLLER ONLY
        16 112462  001402                        BEQ      2$
        17 112464  004737  076626               CALL     RESPDM
        18 112470                     2$:
           112470  012637  064414               MOV      (SP)+,FSIZE         ;;POP STACK INTO FSIZE
           112474  012637  064412               MOV      (SP)+,FFREE         ;;POP STACK INTO FFREE
        19 112500  062737  000046  064430       ADD      #C.SIZE,TSTTAB      ;MOVE TO NEXT CONTROLLER
        20 112506  005777  151716               TST      @TSTTAB             ;CHECK IF ANY MORE CONTROLLER TABLES
        21 112512  001351                        BNE      1$
        38                                       .EVEN
        39
        40 112514                     L10061:
           112514  104401                        TRAP     C$ETST
        41
```

K5

```
   1                                      .SBTTL   TEST 3: DISK FUNCTION TEST
   2
   3 112516                           T3::
   8 112516   012701   000003            MOV      #3,R1                     ;INITIALIZE TEST PARAMETERS
   9 112522   004737   076336            CALL     TINIT
  10 112526   013737   064424   064430   MOV      CTABS,TSTTAB              ;GET ADDRESS OF 1ST TABLE ADDRESS
  11 112534   013701   064426            MOV      CTRLRS,R1                 ;RUN DM PROGRAM ON ALL CONTROLLERS
  12 112540   004737   076532            CALL     RUNDM                     ;  AT ONCE
  13 112544   001402                     BEQ      1$
  14 112546   004737   076626            CALL     RESPDM
  15 112552                           1$:
  32                                      .EVEN
  33
  34 112552                           L10062:
     112552   104401                     TRAP     C$ETST
  35
```

```
   3                                    .SBTTL   TEST 4: DISK EXERCISER
   4
   5 112554                   T4::
  13 112554  022737  000004  064444       CMP      #4,TNUM              ;CHECK IF TEST 4 WAS IN PROGRESS
  15 112562  001053                       BNE      T4STRT               ;BRANCH IF NOT
  16 112564  022737  000002  064440       CMP      #ICONT,IFLAGS        ;CHECK IF HERE BY CONTINUE COMMAND
  17 112572  001047                       BNE      T4STRT               ;BRANCH IF NOT
  18 112574  005037  064440               CLR      IFLAGS               ;CLEAR FLAGS FOR NEXT TIME HERE
  19 112600  013704  064650               MOV      LBUFS,R4             ;GET LOG BUFFER POINTER
  20 112604  001423                       BEQ      LOGCHK               ;  IF ZERO, NONE EXISTS
  21 112606  004137  075664               JSR      R1,LPNTF             ;CALL   LPNTF PRINT ROUTINE
     112612  066543                       .WORD    LOGM1                ;ADDRESS OF ASCIZ STRING
     112614  000000                       .WORD    ARG.CT               ;ARGUMENT COUNT * 2
  22 112616  005037  064650               CLR      LBUFS                ;CLEAR START ADDRESS TO ERASE BUFFER
  23 112622  012405           LOGOUT:     MOV      (R4)+,R5             ;GET CONTROLLER TABLE ADDRESS
  24 112624  004137  103250               CALL     PNTERR               ;PRINT ERROR REPORT
  25 112630  062704  000104               ADD      #<HC.BSZ-2>,R4       ;BUMP POINTER TO NEXT ENTRY
  26 112634  020437  064652               CMP      R4,LBUFN             ;CHECK IF AT END
  27 112640  103770                       BLO      LOGOUT               ;PRINT ALL ENTRIES
  28 112642  004137  075664               JSR      R1,LPNTF             ;CALL   LPNTF PRINT ROUTINE
     112646  066575                       .WORD    LOGM2                ;ADDRESS OF ASCIZ STRING
     112650  000000                       .WORD    ARG.CT               ;ARGUMENT COUNT * 2
  29 112652  000410                       BR       T4CON
  30
  31 112654  032737  001000  064400 LOGCHK: BIT    #SM.LOG,SFPTBL+SO.BIT ;CHECK IF LOG ENABLED
  32 112662  001404                       BEQ      T4CON
  33 112664  004137  075664               JSR      R1,LPNTF             ;CALL   LPNTF PRINT ROUTINE
     112670  066622                       .WORD    LOGM3                ;ADDRESS OF ASCIZ STRING
     112672  000000                       .WORD    ARG.CT               ;ARGUMENT COUNT * 2
  34 112674  005737  064450   T4CON:      TST      URNING               ;CHECK IF ANY CONTROLLERS STILL RUNNING
  35 112700  001404                       BEQ      T4STRT               ;RESTART IF NOT
  36 112702  004737  076626               CALL     RESPDM               ;CONTINUE BY RESPONDING IO REQUESTS
  37 112706  000137  113240               JMP      T4WAIT               ;END OF TEST WHEN DONE
```

```
  1
  2                                    ;      START TEST
  3
 10 112712  012701  000004           T4STRT: MOV    #4,R1                 ;INITIALIZE TEST PARAMETERS
 12 112716  004737  076336                   CALL   TINIT
 13 112722  032737  000014  064440           BIT    #ISTRT!IREST,IFLAGS   ;HERE FROM OPERATOR COMMAND?
 14 112730  001521                            BEQ    T4RUN                 ;RUN WITH PREVIOUS PARAMETERS IF NEW PASS
 15 112732  032737  000200  064400           BIT    #SM.MAN,SFPTBL+SO.BIT ;MANUAL INTERVENTION MODE?
 16 112740  001463                            BEQ    T4DEF                 ;IF NOT, SET UP DEFAULT PARAMETERS
 17 112742  104450                            TRAP   C$MANI
 18 112744  103055                            BCC    T4DEFW
 20 112746  012701  000004                    MOV    #4,R1                 ; R1 = T4QUEST FILE NUMBER
 21 112752  020137  064442                    CMP    R1,FNUM               ; IS IT ALREADY LOADED?
 22 112756  001406                            BEQ    1$                    ; IF SO, BRANCH
 23 112760  005005                            CLR    R5                    ; ELSE R5 = ADJUSTED ADDRESS
 24 112762  004737  105630                    CALL   RDREC                 ; READ IN FILE
 25 112766  103002                            BCC    1$                    ; IF OK, BRANCH
 26 112770  000137  076424                    JMP    TINITE                ; ELSE, ERROR
 28
 29                                   ;INPUT PARAMETERS
 30
 31 112774  005037  064452           1$:     CLR    UCNT                  ;CLEAR COUNT OF UNITS USING PATTERN 16
 32 113000  013705  064424                   MOV    CTABS,R5              ;GET ADDRESS OF 1ST CONTROLLER TABLE
 33 113004  012702  000010           T4PRM1: MOV    #8.,R2                ;GET COUNT OF DRIVE TABLES
 34 113010  010504                           MOV    R5,R4                 ;GET FIRST DRIVE TABLE POINTER
 35 113012  062704  000020                   ADD    #C.DR0,R4
 36 113016  012403                   T4PRM2: MOV    (R4)+,R3              ;GET DRIVE TABLE ADDRESS
 37 113020  001416                           BEQ    T4PRM4                ;GO TO NEXT CONTROLLER IF NONE
 38 113022  032763  100000  000002           BIT    #DT.AVL,D.UNIT(R3)    ;SEE IF TO BE TESTED
 39 113030  001010                           BNE    T4PRM3
 41 113032  004737  002122                   CALL   STORAG                ;ASK QUESTIONS
 45 113036  022763  000020  000006           CMP    #16.,D.PAT(R3)
 46 113044  001002                           BNE    T4PRM3
 47 113046  005237  064452                   INC    UCNT
 48 113052  005302                   T4PRM3: DEC    R2                    ;COUNT DRIVE TABLES
 49 113054  001360                           BNE    T4PRM2                ;GO LOOK AT NEXT
 50 113056  062705  000046           T4PRM4: ADD    #C.SIZE,R5            ;GO TO NEXT CONTROLLER
 51 113062  005715                           TST    (R5)                  ;  IF THERE IS ONE
 52 113064  001347                           BNE    T4PRM1
 53 113066  012701  064554                   MOV    #PAT16C,R1            ; R1  > PAT16C FOR INPUT
 55 113072  004737  002124                   CALL   STORAG+2             ; ASK LAST QUESTIONS
 60 113076  000436                           BR     T4RUN
```

```
     1
     2                                 ;      NOW GET DATA PATTERN 16 IF SELECTED BY ANY DRIVE
     3                                 ;      GIVE WARNING MANUAL INTERVENTION NOT ALLOWED
     4
     5 113100                  T4DEFW:
       113100  004137  075664           JSR     R1.LPNTF            ;CALL    LPNTF PRINT ROUTINE
       113104  065605                   .WORD   T4WARN             ;ADDRESS OF ASCIZ STRING
       113106  000000                   .WORD   ARG.CT             ;ARGUMENT COUNT * 2
```

```
   1
   2                                    !        SET UP DEFAULT PARAMETERS
   3
   4  113110   013705   064424          T4DEF:   MOV     CTABS,R5            ;GET ADDRESS OF 1ST CONTROLLER TABLE
   5  113114   012702   000010          T4DEFA:  MOV     #8.,R2             ;GET COUNT OF DRIVE TABLES
   6  113120   010504                            MOV     R5,R4              ;GET FIRST DRIVE TABLE POINTER
   7  113122   062704   000020                   ADD     #C.DRO,R4          ;GET FIRST DRIVE TABLE POINTER
   8  113126   012403                   T4DEFB:  MOV     (R4)+,R3           ;GET DRIVE TABLE ADDRESS
   9  113130   001415                            BEQ     T4DEFE             ;GO TO NEXT CONTROLLER IF NONE
  10  113132   062703   000004                   ADD     #D.PRM,R3
  11  113136   042713   157777                   BIC     #^C<D.DCY>,(R3)
  12  113142   052723   011012                   BIS     #DDEF,(R3)+
  13  113146   012700   000067                   MOV     #55.,R0
  14  113152   005023                   T4DEFC:  CLR     (R3)+
  15  113154   005300                            DEC     R0
  16  113156   001375                            BNE     T4DEFC
  17  113160   005302                   T4DEFD:  DEC     R2                 ;COUNT DRIVE TABLES
  18  113162   001361                            BNE     T4DEFB             ;GO LOOK AT NEXT
  19  113164   062705   000046          T4DEFE:  ADD     #C.SIZE,R5         ;GO TO NEXT CONTROLLER
  20  113170   005715                            TST     (R5)               ;   IF THERE IS ONE
  21  113172   001350                            BNE     T4DEFA
  22
  23                                    ;START TEST
  24
  25  113174   006137   064440          T4RUN:   ROL     IFLAGS             ;CLEAR FLAGS FOR NEXT TIME HERE
  26  113200   042737   177757   064440          BIC     #^C<ISTRTM>,IFLAGS
  36  113206   013737   064424   064430          MOV     CTABS,TSTTAB       ;GET ADDRESS OF 1ST CONTROLLER TABLE
  37  113214   013701   064426                   MOV     CTRLRS,R1          ;RUN DM PROGRAM ON ALL CONTROLLERS
  39  113220   004737   076436                   CALL    RNT4DM             ;   AT ONCE
  40  113224   001405                            BEQ     T4WAIT
  41  113226   013737   064424   064430          MOV     CTABS,TSTTAB       ; MAKE SURE TSTTAB HAS CONTROLLER INFO
  46  113234   004737   076626                   CALL    RESPDM
  47  113240   032737   001000   064400 T4WAIT:  BIT     #SM.LOG,SFPTBL+SO.BIT  ;CHECK IF LOG IS ENABLED
  48  113246   001402                            BEQ     T4EXIT             ;EXIT IF NOT
  49                                                                        ;>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<
  50  113250   104422                            TRAP    C$BRK
  51  113252   000772                            BR      T4WAIT             ;WAIT TILL STOPPED BY CONTROL C
  52  113254                            T4EXIT:                             ;PRINT A STATISTICAL REPORT
  53  113254   104424                            TRAP    C$DRPT
  54  113256   104432                            TRAP    C$EXIT
      113260   000002                            .WORD   L10063 .
  71                                             .EVEN
  72
  73  113262                            L10063:
      113262   104401                            TRAP    C$ETST
  74
```

C6

```
  2                                             .TITLE PARAMETER CODING
 13
 14                                             .SBTTL   HARDWARE PARAMETER CODING SECTION
 42
 44                                          ;**
 45                                          ; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
 46                                          ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
 47                                          ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
 48                                          ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
 49                                          ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
 50                                          ; WITH THE OPERATOR.
 51                                          ;
 52
 53 113264   000032                                          .WORD L10064 L$HARD/2
    113266                                   L$HARD::
 55           000000                         H.UBA   = 0                          ;UNIBUS ADDRESS
 56           000002                         H.VEC   = 2                          ;UDA VECTOR
 57           000004                         H.BRL   = 4                          ;BR LEVEL
 58           000006                         H.BST   = 6                          ;BURST RATE
 59           000010                         H.DRV   = 10                         ;DRIVE NUMBER
 60           000012                         H.PRM   = 12                         ;PROGRAM PARAMETERS
 61
 62           020000                         HM.CYL  == BIT13                     ;TEST CUSTOMER DATA AREA
 64                                                                               ;PRINT 'UNIBUS ADDRESS OF UDA?'
 65 113266   000031                                  .WORD   T$CODE
    113270   113352                                  .WORD   MSGUBA
    113272   160000                                  .WORD   T$LOLIM
    113274   177774                                  .WORD   T$HILIM
 66
 67 113276   001031                                                              ;PRINT 'VECTOR?'
    113300   113400                                  .WORD   T$CODE
    113302   000004                                  .WORD   MSGVEC
    113304   000774                                  .WORD   T$LOLIM
                                                     .WORD   T$HILIM
 68                                                                              ;PRINT  BR LEVEL?'
 69 113306   002052                                  .WORD   T$CODE
    113310   113407                                  .WORD   MSGBRL
    113312   177777                                  .WORD   -1
    113314   000004                                  .WORD   T$LOLIM
    113316   000007                                  .WORD   T$HILIM
 70                                                                              ;PRINT 'UNIBUS BURST RATE?'
 71 113320   003052                                  .WORD   T$CODE
    113322   113420                                  .WORD   MSGBST
    113324   177777                                  .WORD   -1
    113326   000000                                  .WORD   T$LOLIM
    113330   000077                                  .WORD   T$HILIM
 72                                                                              ;PRINT 'DRIVE 0?'
 73 113332   004052                                  .WORD   T$CODE
    113334   113442                                  .WORD   MSGLDR
    113336   177777                                  .WORD   -1
    113340   000000                                  .WORD   T$LOLIM
    113342   000377                                  .WORD   T$HILIM
 75                                                                              ;PRINT  EXERCISE ON CUSTOMER DATA AREA
 76                                                                              ;        IN TEST 4?
 77 113344   005130                                  .WORD   T$CODE
    113346   113452                                  .WORD   MSGCST
    113350   020000                                  .WORD   HM.CYL
 79                                                  .EVEN
```

D6

```
        80  113352                        L10064:
        84  113352    125    116    111   MSGUBA: .ASCIZ  \UNIBUS ADDRESS OF UDA\
        85  113400    126    105    103   MSGVEC: .ASCIZ  \VECTOR\
        86  113407    102    122    040   MSGBRL: .ASCIZ  \BR LEVEL\
        87  113420    125    116    111   MSGBST: .ASCIZ  \UNIBUS BURST RATE\
        88  113442    104    122    111   MSGLDR: .ASCIZ  \DRIVE 4\
        90  113452    105    130    105   MSGCST: .ASCIZ  \EXERCISE ON CUSTOMER DATA AREA IN TEST 4\
        92                                        .EVEN
        96
       106
```

6

```
        1                                    .SBTTL   SOFTWARE PARAMETER CODING SECTION
        2
        3                             ;**
        4                             ; THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
        5                             ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.   THE
        6                             ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
        7                             ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.   THE
        8                             ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
        9                             ; WITH THE OPERATOR.
       10                             ; -
       11
       12 113524   000030                     .WORD L10065 L$SOFT/2
          113526                      L$SOFT::
       14          000000            SO.EL   = 0                       ;ERROR LIMIT
       15          000002            SO.XL   = 2                       ;DATA TRANSFER LIMIT (MEGABITS)
       16          000004            SO.BIT  = 4                       ;SINGLE BIT ANSWERS
       25
       26          000200            SM.MAN  == BIT07                  ;  MANUAL INTERVENTION MODE
       27          000400            SM.SSF  == BIT08                  ;  SUPPRESS SOFT ERRORS
       28          001000            SM.LOG  == BIT09                  ;  ERROR LOG ENABLED
       29          040000            SM.IW   == BIT14                  ;  INITIAL WRITE
```

F6

```
    1
    2                                                          ;PRINT 'ENTER MANUAL INTERVENTION MODE
    3                                                          ;       FOR SPECIAL DIAGNOSIS?'
    4  113526  002130                         .WORD  T$CODE
       113530  113606                         .WORD  S.MAN
       113532  000200                         .WORD  SM.MAN
    7        .                                                 ;PRINT  REMAINING SOFTWARE QUESTIONS
    8                                                          ;       APPLY TO TEST 4 ONLY'
    9  113534  000003                         .WORD  T$CODE
       113536  113673                         .WORD  S.MES
   11                                                         ;PRINT 'ERROR LIMIT?'
   12  113540  000052                         .WORD  T$CODE
       113542  113756                         .WORD  S.EL
       113544  177777                         .WORD  1
       113546  000001                         .WORD  T$LOLIM
       113550  177777                         .WORD  T$HILIM
   13                                                         ;PRINT 'READ TRANSFER LIMIT IN MEGABYTES
   14                                                         ;       - O FOR NO LIMIT?'
   15  113552  001052                         .WORD  T$CODE
       113554  113772                         .WORD  S.XL
       113556  177777                         .WORD  -1
       113560  000000                         .WORD  T$LOLIM
       113562  177777                         .WORD  T$HILIM
   16                                                         ;PRINT 'SUPPRESS PRINTING SOFT ERRORS?'
   17  113564  002130                         .WORD  T$CODE
       113566  114054                         .WORD  S.SSF
       113570  000400                         .WORD  SM.SSF
   18                                                         ;PRINT 'DO INITIAL WRITE ON START?'
   19  113572  002130                         .WORD  T$CODE
       113574  114112                         .WORD  S.IW
       113576  040000                         .WORD  SM.IW
   20                                                         ;PRINT 'ENABLE ERROR LOG?'
   21  113600  002130                         .WORD  T$CODE
       113602  114144                         .WORD  S.LOG
       113604  001000                         .WORD  SM.LOG
   32                                         .EVEN
       113606                        L10065:
   33
   37  113606   105  116  124  S.MAN:  .ASCIZ  \ENTER MANUAL INTERVENTION MODE FOR SPECIAL DIAGNOSIS\
   40  113673   122  105  115  S.MES:  .ASCIZ  \REMAINING SOFTWARE QUESTIONS APPLY TO TEST 4 ONLY\
   42  113755   000                          .BYTE 0
   43  113756   105  122  122  S.EL:   .ASCIZ  \ERROR LIMIT\
   44  113772   122  105  101  S.XL:   .ASCIZ  \READ TRANSFER LIMIT IN MEGABYTES - O FOR NO LIMIT\
   45  114054   123  125  120  S.SSF:  .ASCIZ  \SUPPRESS PRINTING SOFT ERRORS\
   46  114112   104  117  040  S.IW:   .ASCIZ  \DO INITIAL WRITE ON START\
   47  114144   105  116  101  S.LOG:  .ASCIZ  \ENABLE ERROR LOG\
   56                                          .EVEN
   60
   69
   84  114166                       $PATCH::
   85           000050                          .REPT 40.
  100                                           .EVEN
       114306   114332                          .WORD T$FREE
       114310   000010                          .WORD T$SIZE
       114312                       L$LAST::
```

```
     1
    14
    16 114312  000000                        .WORD     0
       114314  000006                        .WORD     L10070 ./2-1
       114316                     L10066:
    17 114316  172150                        .WORD     172150      ; UNIBUS ADDRESS
    18 114320  000154                        .WORD     154         ; VECTOR ADDRESS
    19 114322  000005                        .WORD     5.          ; BR LEVEL
    20 114324  000077                        .WORD     63.         ; UNIBUS BURST RATE
    21 114326  000000                        .WORD     0.          ; DRIVE NUMBER
    22 114330  000000                        .WORD     0.          ; COSTUMER DATA AREA
    23 114332                     L10070:
    25         000001            .END
```

Symbol	Value		Symbol	Value		Symbol	Value		Symbol	Value		Symbol	Value	
ADR	= 000020	G	CALR5	102142		C$DU	= 000053		C.UNIT	= 000002		D.DC	= 000002	
ALOCM	075422		CALR6	102220		C$EDIT	= 000003		C.VEC	= 000004		D.DCA	= 000001	
APRINT	104472		CALR7	102234		C$ERDF	= 000055		DDEF	= 011012		D.DCY	= 020000	
APRIZ	104506		CALR8	102252		C$ERHR	= 000056		DFPTBL	064356	G	D.DRV	= 000000	
ARG.CT	= 000000		CALR9	102270		C$ERRO	= 000060		DIAG	= 1777'7		D.ECC	= 010000	
ASS	= 100000		CF.ATN	= 000200		C$ERSF	= 000054		DIAGMC	= 000000		D.ECCC	= 000176	
ASSEMB	= 000010		CF.MSC	= 000100		C$ERSO	= 000057		DIVIDE	102704		D.ECYL	= 000160	
A1	= 000001		CF.OTH	= 000040		C$ESCA	= 000010		DIV10	102742		D.END1	= 000120	
A2	= 000002		CF.SHD	= 000002		C$ESEG	= 000005		DLL	064656		D.END2	= 000130	
A3	= 000020		CF.THS	= 000020		C$ESUB	= 000003		DLLADR	064666		D.END3	= 000140	
A4	= 000040		CF.576	= 000001		C$ETST	= 000001		DLLDR	064660		D.END4	= 000150	
BAS	067047		CLOG	105550		C$EXIT	= 000032		DLLNAM	064674		D.HERR	= 000170	
BASLN	067050		CLOSEF	105612		C$GETB	= 000026		DLLR	064664		D.IW	= 040000	
BASL1	066775		CLRBFL	104304		C$GETW	= 000027		DLLSIZ	064672		D.PAT	= 000006	
BASL2	067013		CLRBUF	104256		C$GMAN	= 000043		DLLV	064662		D.PRM	= 000004	
BASL3	067032		CON.A	076062		C$GPHR	= 000042		DMFRST	= 001000		D.RET	= 001000	
BASNO	066651		CON.A1	076066		C$GPLO	= 000030		DMMAIN	= 000040		D.RO	= 004000	
BASN1	066670		CON.A2	076110		C$GPRI	= 000040		DMOVRL	= 000004		D.SEEK	= 000174	
BASN2	066714		CON.D	076112		C$INI1	= 000011		DMPROG	064432		D.SEQ	= 000100	
BASN3	066734		CON.H	076124		C$INLP	= 000017		DMTRLN	= 000000		D.SERN	= 000200	
BASN4	066754		CON.N	076150		C$MANI	= 000050		DONE	101722		D.SERR	= 000172	
BELL	= 000007	G	CON.N1	076154		C$MEM	= 000031		DSPSIZ	= 000017		D.SIZE	= 000206	
BIT0	= 000001	G	CON.O	076136		C$MSG	= 000023		DTABS	064422	G	D.TR	= 000020	
BIT00	= 000001	G	CON.QU	076042		C$OPEN	= 000034		DT.AVL	= 100000		D.UNIT	= 000002	
BIT01	= 000002	G	CON.QX	076060		C$PNTB	= 000014		DT.UNT	= 000077		D.WC	= 000010	
BIT02	= 000004	G	CON.R	076172		C$PNTF	= 000017		DUP	= 001000		D.WCA	= 000004	
BIT03	= 000010	G	CON.R1	076214		C$PNTS	= 000016		DU.DFL	= 020000		D.WO	= 002000	
BIT04	= 000020	G	CON.S	076230		C$PNTX	= 000015		DU.FTL	= 050000		D.XFRR	= 000166	
BIT05	= 000040	G	CON.S1	076234		C$QIO	= 000377		DU.INF	= 030000		D.XFRW	= 000164	
BIT06	= 000100	G	CR	= 000015	G	C$RDBU	= 000007		DU.QUE	= 010000		D.ZERO	= 140200	
BIT07	= 000200	G	CRLF	065042		C$REFG	= 000047		DU.SPC	= 060000		EF.BBR	= 000200	
BIT08	= 000400	G	CTABER	111212		C$RESE	= 000033		DU.TER	= 040000		EF.BBU	= 000100	
BIT09	= 001000	G	CTABS	064424	G	C$REVI	= 000003		D.BB	= 000010		EF.CON	= 000036	G
BIT1	= 000002	G	CTRLRS	064426		C$RFLA	= 000021		D.BB01	= 000012		EF.LOG	= 000040	
BIT10	= 002000	G	CT.AVL	= 100000		C$RPT	= 000025		D.BB02	= 000016		EF.NEW	= 000035	G
BIT11	= 004000	G	CT.BRL	= 007000		C$SEFG	= 000046		D.BB03	= 000022		EF.PWR	= 000034	G
BIT12	= 010000	G	CT.CMD	= 000004		C$SPRI	= 000041		D.BB04	= 000026		EF.RES	= 000037	G
BIT13	= 020000	G	CT.MSG	= 000010		C$SVEC	= 000037		D.BB05	= 000032		EF.SEX	= 000020	
BIT14	= 040000	G	CT.REQ	= 000020		C$TPRI	= 000013		D.BB06	= 000036		EF.STA	= 000040	G
BIT15	= 100000	G	CT.RN	= 000002		C.BST	= 000006		D.BB07	= 000042		EN	= 040000	
BIT2	= 000004	G	CT.UNT	= 000077		C.DRO	= 000020		D.BB08	= 000046		EO	= 140000	
BIT3	= 000010	G	CT.U50	= 000040		C.DR1	= 000022		D.BB09	= 000052		ERRBLK	064410	G
BIT4	= 000020	G	CT.VEC	= 000777		C.DR2	= 000024		D.BB10	= 000056		ERRC	076304	
BIT5	= 000040	G	C$AU	= 000052		C.DR3	= 000026		D.BB11	= 000062		ERRD	076316	
BIT6	= 000100	G	C$AUTO	= 000061		C.DR4	= 000030		D.BB12	= 000066		ERRLIM	065245	
BIT7	= 000200	G	C$BRK	= 000022		C.DR5	= 000032		D.BB13	= 000072		ERRMC	101470	
BIT8	= 000400	G	C$BSEG	= 000004		C.DR6	= 000034		D.BB14	= 000076		ERRMES	101250	
BIT9	= 001000	G	C$BSUB	= 000002		C.DR7	= 000036		D.BB15	= 000102		ERRME1	065101	
BLDCMD	104124		C$CEFG	= 000045		C.FLG	= 000014		D.BB16	= 000106		ERRMSG	064406	G
BLDCO	104164		C$CLCK	= 000062		C.HCOM	= 000016		D.BCYL	= 000154		ERRMSL	101356	
BLDC1	104172		C$CLEA	= 000012		C.JAD	= 000012		D.BE	= 000040		ERRMSX	101464	
BOE	= 000400	G	C$CLOS	= 000035		C.JSR	= 000010		D.BEC	= 000112		ERRNBR	064404	G
BRLEV	106652		C$CLP1	= 000006		C.REF	= 000044		D.BGN1	= 000114		ERRTYP	064402	G
CALR1	101726		C$CVEC	= 000036		C.SIZE	= 000046		D.BGN2	= 000124		ERRVEC	= 000004	G
CALR2	101754		C$DCLN	= 000044		C.TO	= 000040		D.BGN3	= 000134		ERR.SZ	= 000011	
CALR3	102052		C$DODU	= 000051		C.TOH	= 000042		D.BGN4	= 000144		ERR.TB	076252	
CALR4	102066		C$DRPT	= 000024		C.UADR	= 000000		D.CYL	= 000400		ERR.TN	075250	G

```
ERR001 074252 G    F$PROT= 000021     H.DRV = 000010     KW11I  106230 G    L$HPCP 002016 G
ERR002 074270 G    F$PWR = 000017     H.PRM = 000012     LBUFE  064654      L$HPTP 002022 G
ERR003 074306 G    F$RPT = 000012     H.UBA = 000000     LBUFN  064652      L$HW   064356 G
ERR004 074324 G    F$SEG = 000003     H.VEC = 000002     LBUFS  064650      L$ICP  002104 G
ERR005 074336 G    F$SOFT= 000005     IBE   = 010000 G   LDDM   076554      L$INIT 107636 G
ERR006 074400 G    F$SRV = 000010     ICONT = 000002 G   LDNEXT 076606      L$LADP 002026 G
ERR007 074350 G    F$SUB = 000002     IDU   = 000040 G   LF    = 000012 G   L$LAST 114312 G
ERR008 074412 G    F$SW  = 000014     IER   = 020000 G   LOAD   104006      L$LOAD 002100 G
ERR014 074362 G    F$TEST= 000001     IFLAGS 064440 G    LOADB  103600      L$LUN  002074 G
ERR021 074430 G    GETCDN 102430      INITBL 105276      LOADDM 103410      L$MREV 002050 G
ERR022 074470 G    GETCNT 102364      INITWA 065454      LOADER 104120      L$NAME 002000 G
ERR023 074516 G    GETCNX 102370      INITWB 065560      LOADE1 104110      L$PRIO 002042 G
ERR024 074602 G    GETCXX 102436      INITWC 064763      LOADT1 103674      L$PROT 107630 G
ERR025 074616 G    GTDRVT 102274      INITXX 111170      LOE   = 040000 G   L$PRT  002112 G
ERR026 074634 G    GTT452 076500      INIT1  110140      LOG   = 000001     L$REPP 002062 G
ERR027 074654 G    G$CNTO= 000200     INIT2  110326      LOGCHK 112654      L$REV  002010 G
ERR028 074672 G    G$DELM= 000372     INIT3  110364      LOGM1  066543      L$RPT  106654 G
ERR029 074704 G    G$DISP= 000003     INIT4  110420      LOGM2  066575      L$SOFT 113526 G
ERR030 074720 G    G$EXCP= 000400     INIT5  111006      LOGM3  066622      L$SPC  002056 G
ERR031 074734 G    G$HILI= 000002     INTRCV 064454      LOGOUT 112622      L$SPCP 002020 G
ERR032 074746 G    G$LOLI= 000001     INTSRV 106252 G    LOT   = 000010 G   L$SPTP 002024 G
ERR033 074764 G    G$NO  = 000000     INTSTO 065342      LPNT   075722      L$STA  002030 G
ERR034 074772 G    G$OFFS= 000400     INTST1 065437      LPNTB  075674      L$SW   064374 G
ERR035 075000 G    G$OFSI= 000376     IPADRS 064534      LPNTF  075664      L$TEST 002114 G
ERR036 075016 G    G$PRMA= 000001     IREST = 000004 G   LPNTS  075714      L$TIML 002014 G
ERR037 075030 G    G$PRMD= 000002     ISR   = 000100 G   LPNTX  075704      L$UNIT 002012 G
ERR038 075044 G    G$PRML= 000000     ISTRT = 000010 G   LT1L1  103716      L10000 064372
ERR23A 074536      G$RADA= 000140     ISTRTH= 000020 G   LT1L2  103744      L10001 064402
ERR23B 074562      G$RADB= 000000     IXE   = 004000 G   LT11   103730      L10002 074266
ERR23C 074570      G$RADD= 000040     I$AU  = 000041     L$ACP  002110 G    L10003 074304
EVL   = 000004 G   G$RADL= 000120     I$AUTO= 000041     L$APT  002036 G    L10004 074322
E$END = 002100     G$RADO= 000020     I$CLN = 000041     L$AU   111336 G    L10005 074334
E$LOAD= 000035     G$XFER= 000004     I$DU  = 000041     L$AUT  002070 G    L10006 074346
FDATA  064472      G$YES = 000010     I$HRD = 000041     L$AUTO 111300 G    L10007 074360
FFREE  064412 G    HCOMM  075446      I$INIT= 000041     L$CCP  002106 G    L10010 074376
FILOPN 064474      HC.BF1= 000100     I$MOD = 000041     L$CLEA 111302 G    L10011 074410
FMEM   064416      HC.BF2= 000206     I$MSG = 000041     L$CO   002032 G    L10012 074426
FMEMS  064420      HC.BSZ= 000106     I$PROT= 000040     L$DEPG 002011 G    L10013 074466
FMERR  075410      HC.CCT= 000012     I$PTAB= 000041     L$DESC 064724 G    L10014 074514
FNAME  064456      HC.CEV= 000014     I$PWR = 000041     L$DESP 002076 G    L10015 074600
FNUM   064442      HC.CMD= 000010     I$RPT = 000041     L$DEVP 002060 G    L10016 074614
FRMTT  065037      HC.CPK= 000020     I$SEG = 000041     L$DTSP 064344 G    L10017 074632
FS    = 100000     HC.ESZ= 000004     I$SETU= 000041     L$DLY  002116 G    L10020 074652
FSIZE  064414 G    HC.INT= 000000     I$SFT = 000041     L$DTP  002040 G    L10021 074670
FWORD  106150      HC.ISZ= 000004     I$SRV = 000041     L$DTYP 002034 G    L10022 074702
F$AU  = 000015     HC.MCT= 000006     I$SUB = 000041     L$DU   111330 G    L10023 074716
F$AUTO= 000020     HC.MEV= 000014     I$TST = 000041     L$DUT  002072 G    L10024 074732
F$BGN = 000040     HC.MPK= 000020     J$JMP = 000167     L$DVTY 064700 G    L10025 074744
F$CLEA= 000007     HC.MSG= 000004     KTBASA 064434      L$EF   002052 G    L10026 074762
F$DU  = 000016     HC.PSZ= 000060     KTBASO 064436      L$ENVI 002044 G    L10027 074770
F$END = 000041     HC.RSZ= 000004     KTMEM  064640      L$ERRT 064402 G    L10030 074776
F$HARD= 000004     HC.SIZ= 000314     KW.BRL 064620      L$ETP  002102 G    L10031 075014
F$HW  = 000013     HELP  = 000000     KW.CSR 064616      L$EXP1 002046 G    L10032 075026
F$INIT= 000006     HM.CYL= 020000 G   KW.EL  064626      L$EXP4 002064 G    L10033 075042
F$JMP = 000050     HOE   = 100000 G   KW.HZ  064624      L$EXP5 002066 G    L10034 075054
F$MOD = 000000     H.BRL = 000004     KW.OUT= 000105 G   L$HARD 113266 G    L10035 075406
F$MSG = 000011     H.BST = 000006     KW.VEC 064622      L$HIME 002120 G    L10036 104516
```

Symbol	Value	Symbol	Value	Symbol	Value	Symbol	Value	Symbol	Value
L10037	104526	MSGPKT	075150	PAT16C	064554	P.SHST=	000042	RSP.CK	105366
L10040	106250	MSGUBA	113352	PAT16W	064556	P.SHUN=	000040	RSP.S1	105314
L10041	106256	MSGVEC	113400	PB	075566	P.STS =	000012	RSP.S2	105322
L10042	107626	MX =	000252	PF	075542	P.TIME=	000024	RSP.S3	105342
L10044	111276	MXFERE	101142	PNT =	001000 G	P.TRKS=	000044	RSP.S4	105360
L10045	111300	MXFERP	065170	PNTERR	103250	P.UADR=	000020	RUNDM	076532
L10046	111326	MXFERX	101140	PNTNUM	102442	P.UNFL=	000016	RWORDT	106110
L10047	111334	M1 =	000004	PNTNUS	102450	P.UNIT=	000004	RWRDE1	106216
L10050	111342	M2 =	000010	PNTPKL	075106	P.UNSZ=	000044	SAMVEC	111262
L10051	112416	M3 =	000100	PNTPKT	075056	P.UNTI=	000024	SA.A2 =	001000
L10052	111522	M4 =	000200	PRI =	002000 G	P.USEF=	000022	SA.BST=	000374
L10053	111564	NCON	075374	PRINTC	075506	P.VRSN=	000014	SA.CMD=	034000
L10054	111720	NCONF	076024	PRI00 =	000000 G	P.VSER=	000050	SA.CME=	000070
L10055	112320	NCONS	076002	PRI01 =	000040 G	RDDAT	106004	SA.CM1=	004000
L10056	112332	NOCLOC	066466	PRI02 =	000100 G	RDDLL	105570	SA.CNT=	000360
L10057	112346	NXMAD	064636	PRI03 =	000140 G	RDERR	106172	SA.CTP=	003400
L10060	112410	NXMI	104310 G	PRI04 =	000200 G	RDREC	105630	SA.DI =	000400
L10061	112514	ONEFIL=	000061	PRI05 =	000240 G	RDST	105672	SA.ERC=	003777
L10062	112552	OP.ABO=	000001	PRI06 =	000300 G	RDSTS	105670	SA.ERR=	100000
L10063	113262	OP.ACC=	000020	PRI07 =	000340 G	RESET	106260	SA.GO =	000001
L10064	113352	OP.AVA=	000100	PS	075636	RESPCT	076640	SA.INE=	000200
L10065	113606	OP.AVL=	000010	PTYPE	075770	RESPDM	076626	SA.INT=	000200
L10066	114316	OP.CCD=	000021	PX	075612	RG.FLG=	040000	SA.LFC=	000002
L10070	114332	OP.CMP=	000040	P.BCNT=	000014	RG.OWN=	100000	SA.MCV=	000017
MC =	000314	OP.DUP=	000101	P.BUFF=	000020	RNTIM	065045	SA.MSE=	000007
MD =	000125	OP.ELP=	000003	P.CMST=	000020	RNTIME	106362	SA.MSG=	003400
MD.CMP=	040000	OP.END=	000200	P.CNCL=	000022	RNTIMX	106542	SA.MS1=	000400
MD.CWB=	000010	OP.ERS=	000022	P.CNTF=	000016	RNTIM1	065070	SA.NV =	002000
MD.ERR=	010000	OP.ESP=	000002	P.CNTI=	000024	RNTIM2	065076	SA.NVE=	000400
MD.EXP=	100000	OP.FLU=	000023	P.CPSP=	000042	RNT4DM	076436	SA.PRG=	000001
MD.FEU=	000001	OP.GCS=	000002	P.CRF =	000000	RPTCT	106764	SA.STE=	000200
MD.IMF=	000002	OP.GSS=	000001	P.CTMO=	000020	RPTCTN	107232	SA.STP=	100000
MD.NXU=	000001	OP.GUS=	000003	P.CYLS=	000050	RPTDT	107004	SA.S1 =	004000
MD.PRI=	000001	OP.MRD=	000030	P.DEXT=	000014	RPTDTN	107226	SA.S2 =	010000
MD.RIP=	000001	OP.MWR=	000031	P.DFLG=	000017	RPTMD2	107603	SA.S3 =	020000
MD.SCH=	004000	OP.ONL=	000011	P.DMDT=	000024	RPTMSD	107534	SA.S4 =	040000
MD.SCL=	002000	OP.RD =	000041	P.DPRG=	000020	RPTMSG	107262	SA.TST=	100000
MD.SEC=	000100	OP.RLC=	000103	P.DTMO=	000024	RPTMSH	107316	SA.VCE=	000177
MD.SEQ=	000020	OP.RPL=	000024	P.ELGF=	000034	RPTXX	107242	SA.VEC=	000177
MD.SER=	000400	OP.RSD=	000005	P.FBBK=	000034	RSPDRP	077036	SA.WRP=	040000
MD.SPD=	000001	OP.SCC=	000004	P.FLGS=	000011	RSPDSP	077360	SEKERE	101000
MD.SSH=	000200	OP.SEX=	000007	P.GRPS=	000046	RSPERR	077122	SETO0	104542
MD.SWP=	000004	OP.SHC=	000102	P.HSTI=	000020	RSPIN	077054	SETO1	104550
MD.VOL=	000002	OP.SSD=	000004	P.HTMO=	000020	RSPMWR	077074	SETO2	104556
MD.WBN=	000100	OP.SUC=	000012	P.LBN =	000034	RSPNRP	077022	SETTO	104530
MD.WBV=	000400	OP.WR =	000042	P.MEDI=	000034	RSPNTO	076766	SFPTBL	064374 G
MEMFIL	077426	OSTRE	076040	P.MLUN=	000014	RSPNXT	076770	SM.IW =	040000 G
MESSAG	101610	OSTRNG	075772	P.MOD =	000012	RSPOU	077142	SM.LOG=	001000 G
MESSG	065713	O$APTS=	000000	P.OPCD=	000010	RSPOU2	077324	SM.MAN=	000200 G
MLDRER	111226	O$AU =	000000	P.OTRF=	000014	RSPOU3	077332	SM.SSF=	000400 G
MM =	000377	O$BGNR=	000001	P.OVRL=	000034	RSPPTW	077134	SNDCMD	104210
MSCP =	000000	O$BGNS=	000001	P.RBN =	000014	RSPPT2	077144	SND.S1	105300
MSGBRL	113407	O$DU =	000000	P.RBNS=	000056	RSPPT3	077214	SND.S2	105304
MSGBST	113420	O$ERRT=	000001	P.RCTC=	000057	RSPRPT	077020	SND.S3	105310
MSGCST	113452	O$GNSW=	000001	P.RCTS=	000054	RSPTM	076724	SO.BIT=	000004
MSGLDR	113442	O$POIN=	000001	P.RGID=	000034	RSPTMO	076754	SO.EL =	000000
MSGPKL	075174	O$SETU=	000001	P.RGOF=	000040			SO.XL =	000002

PARAMETER CODING MACRO V05.00 Wednesday 04 Jan 84 16:12 Page 221 4
Symbol table

SFQ 0282

SSTEP4	105406	T$CODE=	002130	T1GOOD	111522	T4BB2	100704	UTOT2	101214
STIME	064632	T$ERRN=	000035	T1MSIZ	077416	T4BB2E	100732	UTOT3	101236
STLDDM	076542	T$EXCP=	000000	T1NEXT	111362	T4CON	112674	UTOT4	101244
STORAG	002122	T$FLAG=	000040	T1NOW	111422	T4DEF	113110	U52EXT	064526
STOSIZ=	062220	T$FREE=	114332	T1SKIP	111402	T4DEFA	113114	WAITMS	104320
ST.ABO=	000002	T$GMAN=	000000	T1.1	111426	T4DEFB	113126	WCHNG	106554
ST.AOL=	000400	T$HILI=	177777	T1.2	111524	T4DEFC	113152	WCHNGD	106650
ST.AVL=	000004	T$LAST=	000001	T1.3	111566	T4DEFD	113160	XFRU	074215
ST.CMO=	000001	T$LOLI=	000000	T1.4	111722	T4DEFE	113164	XMSG1	073543
ST.CMP=	000007	T$LSYM=	010000	T1.5	112322	T4DEFW	113100	XMSG2	073577
ST.CNT=	000012	T$LTNO=	000004	T1.6	112334	T4EXIT	113254	XPKT1	073644
ST.DAT=	000010	T$NEST=	177777	T1.7	112350	T4MPRM	100352	XPKT2	074135
ST.DIA=	000037	T$NS0=	000000	T2	112420 G	T4MXFR	101002	XSA	074164
ST.DRV=	000013	T$NS1 =	000005	T2CMD	077702	T4OPT7	064760	X$ALWA=	000000
ST.HST=	000011	T$NS2 =	000002	T2CMDE	100340	T4PRM1	113004	X$FALS=	000040
ST.MFE=	000005	T$PCNT=	000000	T2CMDM	077716	T4PRM2	113016	X$OFFS=	000400
ST.MSK=	000037	T$PTAB=	010067	T2CMDN	100262	T4PRM3	113052	X$TRUE=	000020
ST.OFL=	000003	T$PTHV=	000001	T2CMDQ	100074	T4PRM4	113056	X1	067160
ST.SUB=	000040	T$PTNU=	000001	T2CMDR	100236	T4RUN	113174	X1A	067061
ST.SUC=	000000	T$SAVL=	177777	T2CMDV	100162	T4SEEK	100762	X14	070061
ST.WPR=	000006	T$SEGL=	177777	T2CMDW	100312	T4SOFT	100734	X2	067254
SVCGBL=	000000	T$SIZE=	000010	T2CMDX	100334	T4STRT	112712	X2A	067061
SVCINS=	000000	T$SUBN=	000000	T2CMDO	077736	T4UPRM	100374	X21	070650
SVCSUB=	000000	T$TAGL=	177777	T2CMD2	100010	T4UPRX	100652	X22	070765
SVCTAG=	000000	T$TAGN=	010071	T2CMD3	100172	T4WAIT	113240	X23A	071123
SVCTST=	000000	T$TEMP=	000000	T2CMD9	077722	T4WARN	065605	X23B	071431
S$LSYM=	010000	T$TEST=	000004	T2CMS1	066056	UAM =	000200 G	X24	071445
S.EL	113756	T$TSTM=	177777	T2CMS5	066445	UCNT	064452	X25	071635
S.IW	114112	T$TSTS=	000001	T2DLL	077536	UDAINT	104612	X26	072006
S.LOG	114144	T$$AU =	010050	T2DR	064646	UDAIST	105070	X27	072174
S.MAN	113606	T$$AUT=	010045	T2GND1	103142	UDARSD	105546	X28	072321
S.MES	113673	T$$CLE=	010046	T2GND2	103144	UDARSP	105410	X29	072371
S.SSF	114054	T$$DAT=	010070	T2GND3	103204	UDASRV	104520 G	X3	067323
S.XL	113772	T$$DU =	010047	T2GNE	103240	UF.CMR=	000001	X3A	067061
TEMP	064476	T$$HAR=	010064	T2GNUM	103122	UF.CMW=	000002	X30	072662
TIEXIT	076422	T$$HW =	010000	T2GNX	103234	UF.INA=	040000	X31	072775
TINDEX=	000006	T$$INI=	010044	T2PNT	103050	UF.RPL=	100000	X32	073117
TINIT	076336	T$$MSG=	010035	T2PNTB	103040	UF.SCH=	004000	X35	073230
TINITE	076424	T$$PC =	000001	T2PNTD	103112	UF.SCL=	002000	X36	073310
TNAMES	076274	T$$PRO=	010043	T2PNTO	103070	UF.WBN=	000100	X37	073426
TNUM	064444	T$$PTA=	010067	T2PNTW	103010	UF.WPH=	020000	X38	070460
TOOMER	111244	T$$RPT=	010042	T2WARN	065757	UF.WPS=	001000	X4	067406
TSTTAB	064430	T$$SOF=	010065	T2WRO	064644	UF.576=	000004	X5	067725
TTYOUT	075662	T$$SRV=	010041	T2WRR	064642	URNING	064450	X6	067573
TYPCNT	064532	T$$SUB=	010060	T3	112516 G	URUN	064446	X7	067775
TY.U50=	000002	T$$SW =	010001	T4	112554 G	UTOTST	101144	X8	067660
TY.U52=	000001	T$$TES=	010063	T4BB1	100654	UTOT1	101156	X8A	067061
T$ARGC=	000004	T1	111344 G	T4BB1E	100702	UTOT1A	101210	$PATCH	074166 G

```
. ABS.   114332    000   (RW,I,GBL,ABS,OVR)
         000000    001   (RW,I,LCL,REL,CON)
Errors detected:  0
```

*** Assembler statistics

L6

SFQ 0283

Work file reads: 396
Work file writes: 367
Size of work file: 29429 Words (115 Pages)
Size of core pool: 17152 Words (67 Pages)
Operating system: RT 11 (Under RSTS/E)

Elapsed time: 00:04:35.08
ZUDCEO.ZUDCEO/C-SVC34R.MLB/P:1,ZUDCEO.DOC,ZUDCEO

```
$PATCH  220 84◊
A1        1 23◊      1 34      1 44     1-203    83-359   83-428   103-26   200-15
A2        1 24◊      1 35      1-45     1-285    83-360   83-429   103-27   104-12   104-52   106-42   106-113  216-38
A3        1 27◊      1 34      1-46     1-564    83-361   83-430   103-30   200-15
A4        1 28◊      1-35      1 47     1-646    83-362   83-431   103-31   104-12   104-52   106-42   106-113  216-38
ADR      88 57◊
ALOCM   108 17◊    109-16    192-11    194-14   195-38
APRINT  170 15◊
APRIZ   170 18     170-20◊
ARG.CT  106-27    106 27    106-27◊   106-27◊  106-31   106-31   106-31◊  106-31◊  106-35   106-35   106-35◊  106-35◊  106-39   106-39◊
        106-44    106-44◊   106-48    106-48◊  106-53   106-53   106-53   106-53◊  106-53◊  106-53◊  106-57   106-57◊  106-61   106-61
        106-61◊   106-61◊   106-83    106-83   106-83   106-83◊  106-83◊  106-83◊  106-88   106-88   106-88   106-88◊  106-88◊  106-88◊
        106-92    106-92    106-92    106-92◊  106-92◊  106-92◊  106-96   106-96   106-96   106-96◊  106-96◊  106-96◊  106-102  106-102◊
        106-106   106 106   106-106◊  106-106◊ 106-110  106-110  106-110  106-110◊ 106-110◊ 106-110◊ 106-115  106-115  106-115  106-115◊
        106 115◊  106-115◊  106-119   106-119  106-119◊ 106-119◊ 106-123  106-123◊ 106-127  106-127  106-127◊ 106-127◊ 106-132  106-132
        106 132◊  106-132◊  106-136   106-136◊ 106-140  106-140◊ 106-153  106-153◊ 106-158  106-158◊ 106-162  106-162  106-162◊ 106-162◊
        106 166   106-166◊  106-169   106-169◊ 106-176  106-176  106-176  106-176  106-176  106-176◊ 106-176◊ 106-176◊ 106-176◊ 106-176◊
        106 183   106-183◊  106-187   106-187  106-187  106-187  106-187  106-187  106-187  106-187  106-187◊ 106-187◊ 106-187◊ 106-187◊
        106-187◊  106-187◊  106 187◊  106-187◊ 106-200  106-200  106-200  106-200  106-200  106 200  106-200  106-200  106-200◊ 106-200◊
        106-200◊  106-200◊  106-200◊  106-200◊ 106-200◊ 106-200◊ 112-47   112-47◊  114-26   114-26◊  130-36   130-36◊  130-45   130-45
        130-45    130-45    130-45◊   130-45◊  130-45◊  130-45◊  130-111  130-111◊ 138-55   138-55   138-55   138-55   138-55◊  138-55◊
        138-55◊   138-55◊   138-57    138-57◊  142-55   142-55   142-55◊  142-55◊  143-26   143-26   143-26   143-26   143-26◊  143-26◊
        143-26◊   143-26◊   148-7     148-7    148-7    148-7    148-7    148 7◊   148-7◊   148-7◊   148-7◊   148-7◊   149-7    149 7
        149-7     149-7     149-7     149-7    149-7◊   149-7◊   149-7◊   149-7◊   149-7◊   149-7◊   183 25   183-25   183-25◊  183-25◊
        183-29    183-29    183-29◊   183-29◊  183-34   183-34   183-34◊  183-34◊  186-50   186-50   186-50◊  186-50◊  186-71   186-71◊
        191-12    191-12◊   196-6     196-6◊   207-18   207-18   207-18   207-18◊  207-18◊  207-18◊  207-75   207-75◊  213-21   213-21◊
        213-28    213-28◊   213-33    213-33◊  215-5    215-5◊
ASS       1-40◊     1-42      1-43     1-44     1-45     1-46     1-47     1-48     1-49     1-50     1-51     1-63     1-117    1-203
          1-285     1-366     1-450    1-564    1-646    1-727    1-811    83-357   83-358   83-359   83-360   83-361   83-362   83-363
         83-364    83-365    83-366   83-426   83-427   83-428   83-429   83-430   83-431   83-432   83-433   83-434   83-435   83-447
         85-8      85-9      85-10    87-14    97-15    98-27    99-20    101-6    101-7    103-24   103-25   103-26   103-27   103-28
        103-29    103-30    103-31   103-32   103-33   103-42   103-85   103-90   104-12   104-20   104-52   105-9    106-42   106-64
        106-113   115-17    115-22   117-19   117-42   118-1    120-24   121-30   128-28   129-39   130-30   133-9    136-12   136-23
        138-30    138-60    140-26   140-41   140-61   140-69   140-74   142-48   142-57   159-1    161-27   162-16   162-63   162-74
        162-88    162-92    169-22   173-7    173-18   173-32   173-64   173-73   174-92   178-12   179-11   179-20   183-38   186-54
        186 63    186-75    186-89   186-122  186-150  186-179  190-49   191-23   195-39   196-22   200-11   200-15   203-13   203-53
        213-2     213-10    214-7    214-19   214-40   214-54   216-27   216 28   216-38   218-74   218-89   219-17   220-5    220-6
        220-23    220-38    220-39   220-49   220-76   220 83
ASSEMB   83-374    83-374
BAS     103-96◊   106-199   148-7    148-7    148-7    149-7    149 7
BASL1   103 93◊   106-200
BASL2   103-94◊   106-200   149-7
BASL3   103-95◊   106 196
BASLN   103-98◊   106-200   148-7    149-7
BASNO   103-84◊   148-7     149-7
BASN1   103-86◊   115-18
BASN2   103-87◊   115-19
BASN3   103 88◊   115-20
BASN4   103-91◊   115-23
BELL     88 60
BITO     88 57◊    98-26    101-23
BIT00    88-57     88-57◊
BIT01    88-57     88 57◊
BIT02    88-57     88-57◊
BIT03    88 57     88-57◊
BIT04    88-57     88-57◊
```

```
BIT05  88-57   88-57@
BIT06  88-57   88-57@
BIT07  88-57   88-57@  219-26
BIT08  88-57   88-57@  219-27
BIT09  88-57   88-57@  219-28
BIT1   88-57@  97-23   98-25   100-25  101-22
BIT10  88-57@  98 17
BIT11  88-57@  98-16
BIT12  88-57@  98-15
BIT13  88-57@  98 14   218-62
BIT14  88-57@  98-13   219 29
BIT15  88-57@  97-14   98-12   98-32   120-32  138-25  138-26  138-53  139 26  139 35  140-52  142-56  156-30  161-26
       162-56  186-77  186-91  203-66
BIT2   88 57@  97 24   98-24   100-26
BIT3   88-57@  97-21   98-23   100-27
BIT4   88-57@  97-19   98-22   100-28
BIT5   88-57@  97-18   98-21
BIT6   88 57@  98-20
BIT7   88-57@  98-32   147-11
BIT8   88-57@  98-19
BIT9   88-57@  98-18
BLDC0  166-23  166-25@
BLDC1  166-27@ 166-29
BLDCMD 124 16  124-35  162-69  164-18  166-16@
BOE    88-57@
BRLEV  184-26@ 207-17* 207-71
C$AU   83-374@ 202-35
C$AUTO 83-374@ 199-18
C$BRK  83-374@ 121-50  169-33  175-15  176-29  180-29  184-13  207-38  216-50
C$BSEG 83 374@
C$BSUB 83-374@ 204-2   205-7   206-6   207-6   208-2   209-2   210-2
C$CEFG 83-374@
C$CLCK 83-374@ 191-7   191-9
C$CLEA 83-374@ 200-43
C$CLOS 83-374@ 179-14
C$CLP1 83 374@ 204-14
C$CVEC 83-374@ 175-32  204-10  207-78
C$DCLN 83 374@ 107-9   117-32  180-120 190-38  196-31  198-8   198-15  198-22  198-29
C$DODU 83 374@
C$DRPT 83 374@ 122-12  216-53
C$DU   83-374@ 201-34
C$EDIT 83-374@ 83-427
C$ERDF 83 374@ 121-24  121-46  123-22  124-12  154-25  162 57  162-61  165 4   169 41  169 50  174-65  174 86  175 35
       175-105 176-38  176 45  184-21  204-13  206-27  207-66  207-73
C$ERHR 83-374@
C$ERRO 83-374@ 161-45
C$ERSF 83-374@ 107-7   117-30  180-118 190-36  198-6   198-13  198-20  198-27
C$ERSO 83 374@
C$ESCA 83-374@
C$ESEG 83 374@
C$ESUB 83 374@ 204-17  205-16  206-33  207 79  208 5   209-8   210 9
C$ETST 83-374@ 210-29  211-40  212 34  216 73
C$EXIT 83-374@ 197-31  200-28  203 70  216 54
C$GETB 83-374@ 180 31  180-37  180 48  180 77  180 85  180-92  180-104 180-108
C$GETW 83 374@
C$GMAN 83 374@ 130-64  196-27
```

B7

```
C$GPHR   83-3740 190 26    193 7
C$GPLO   83 3740
C$GPRI   83-3740
C$INIT   83-3740 198-43
C$INLP   83-3740
C$MANI   83-3740 130 34    196 25    214 17
C$MEM    83-3740 191-30
C$MSG    83-3740 106 28    106 32    106-36   106-40   106 45   106 49   106-54   106 58   106 62   106-84   106-89   106 103  106 107
         106 111 106 116   106 120   106-124  106-128  106-133  106 137  106 142  106 146  106 150  106 155  106-159  106 163  106 167
         106-214
C$OPEN   83 3740 180 25
C$PNTB   83-3740 110-20
C$PNTF   83-3740 110-18    196 15
C$PNTS   83-3740 110-24    186-119  186 121
C$PNTX   83 3740 110-22
C$QIO    83-3740
C$RDBU   83 3740
C$REFG   83-3740 189-3     189-9     189 15   189 22
C$RESE   83 3740 83-3740
C$REVI   83-3740 83-427
C$RFLA   83-3740 138-50    140-38    142 45
C$RPT    83-3740 186-200
C$SEFG   83-3740
C$SPRI   83-3740 197-4     207-22    207-49   207-60   207-69
C$SVEC   83-3740 162-36    175-28    182-15   191-20   204-5    207-21
C$TPRI   83-3740
C.BST    97-300  174-100   193 24
C.DR0    97-350  139-20    154-16    186 84   190-7    195-5    196 9    214 35   216-7
C.DR1    97-360
C.DR2    97-370
C.DR3    97-380
C.DR4    97 390
C.DR5    97 400
C.DR6    97 410
C.DR7    97 420
C.FLG    97 330  120-?7*   121-10    121-13   121-15   122-23*  123 7    123-27   124 30*  124 31   124-38*  124-41*  138-54*  140 60*
         143 19* 162-86*   162-87*   164 28*  167-21*  169-18   169 46*  194 28
C.MCOM   97 340  106-184   109-18*   121-9    166-17   167-15
C.JAD    97 320
C.JSR    97-310  162-35
C.REF    97-450  123-20    167-16*   167-17
C.SIZE   97-470  120-36    122-16    186-148  190-14   190-17   190 32   193-34   194-13   194 28   196-18   203-67   211-19   214 50
         216-19
C.TO     97-430  121-44    124-46    169-15   169-39   176-23   176-35   184-8    184-19   207-33   207-45
C.TOH    97-440  121-41    169-36    176-32   184-16   207 42
C.UADR   97 270  106-201   175-25    193-14   206-8    207-8    207-19
C.UNIT   97-280  120-29    120-30    121-12   161-18   162-56*  186 74   190-5*   190-43*  203 63   203-64
C.VEC    97-290  162-32    193 22    193-28   207 9    207-76
CALR1    115-4   145-70    147-12
CALR2    115-5   146 60
CALR3    115 6   147-110
CALR4    115-7   148-70
CALR5    115-8   149 70
CALR6    115 9   150-60
CALR7    115 10  151 60
CALR8    115 11  152 60
```

```
CALRQ   115 12    153 60
CF.576   93 420
CF.ATN   93 370
CF.MSC   93 380
CF.OTH   93 390
CF.SMD   93 410
CF.TMS   93 400
CLOG    174 30    174-38   177-110
CLOSEF  178 19    179 120  180-113  197 6    200 10
CLRBFL  168 230   168 25
CLRBUF  124-18    124 37   168 170
CON.A   113-150   116-31
CON.A1  113-160   113 18   146-21
CON.A2  113 20    113 220
CON.D   113 260   116-31
CON.H   113-320   116-31
CON.N   114-100   116-31
CON.N1  114-110   114-13
CON.O   114-40    116-31
CON.QU  113-50    113-9    116-31
CON.QX  113-7     113-100
CON.R   114-190   116-31
CON.R1  114-21    114-260
CON.S   114-320   116-31
CON.S1  114-330   114-35
CR       88-58    106-203  110 11   114-11   130-60   146-14   146-22   148-9    149 9    186-52
CRLF    103-550   110 13
CT.AVL   97-140   120-32   186-77   190-5    190-43   203-66
CT.BRL   97-120   207-12
CT.CMD   97-240   121 15   162-86   164-28   167-21
CT.MSG   97-210   121 13   122-23   124-30   138-54   140-60   143-19   169-18   169-46   172-20
CT.REQ   97-190   123-7    123-27   124-31   124-38   124-41   162-86
CT.RN    97-230   121-10   122-23   162 87
CT.USO   97-180
CT.UNT   97-100
CT.VEC   97 110   162-33   193-29   207-11   207-77
CTABER  193 26    198 50
CTABS   100-150   180-72   190-4    190-29   192-160  192-170  193-10   196-7    203 61   211 10   212-10   214 32   216 4    216 36
        216 41
CTRLRS  100-160   192-180  194 320  212-11   216-37
D.BB     98-390   134-26
D.BB01   98-400
D.BB02   98-410
D.BB03   98 420
D.BB04   98 430
D.BB05   98 440
D.BB06   98 450
D.BB07   98-460
D.BB08   98-470
D.BB09   98 480
D.BB10   98 490
D.BB11   98 500
D.BB12   98-510
D.BB13   98-520
D.BB14   98 530
D.BB15   98 540   135-19
```

```
D.BB16   98 55●
D.BCYL   99 11●   133-45    133-63
D.BE     98-21●   133-23    133-59    133-71
D.BEC    99-2●    133-30    133-41    133 51
D.BGN1   99 3●
D.BGN2   99-5●
D.BGN3   99-7●
D.BGN4   99-9●
D.CYL    98-19●   133-36
D.DC     98 25●   98-30
D.DCA    98-26●
D.DCY    98-14●   196-13    216-11
D.DRV    98-35●   186-120
D.ECC    98 15●   98-30
D.ECCC   99 18●   136-22●   186 121
D.ECYL   99-12●   195-30●
D.END1   99-4●
D.END2   99-6●
D.END3   99-8●
D.END4   99-10●
D.HERR   99 15●   142-42●   142-43    186-121
D.IW     98-13●   98-32     133-20
D.PAT    98 38●   133-22    214-45
D.PRM    98-37●   133-14    133-36    196-13    216-10
D.RET    98-18●   98-30
D.RO     98-16●
D.SEEK   99-17●   137-16●   186-119
D.SEQ    98-20●
D.SERN   99 19●   133-6●    133 7●    133-8●    186-106   186-107   186-108
D.SERR   99-16●   136-21●   186-121
D.SIZE   99-25●   139-29    192-8     195-26    195-30●   195-32
D.TR     98-22●
D.UNIT   98-36●   130-45    138-23    138-55    139 24    142-55   143 24   143 26   154 31   161 24   186 88   186 119   190-11●   190-44●
         196 15   214-38
D.WC     98-23●   98-30
D.WCA    98-24●
D.WO     98 17●
D.XFRR   99-14●   138-44●   138-48    186-119
D.XFRW   99 13●   138-45●   139-28    139-29    186 119
D.ZERO   98-32●   133-15
DDEF     98 30●   195-25    216-12
DFPTBL   86 10●
DIAG     90 13●   166-24
DIAGMC   83 374    83-374
DIV10    158 20●   186-110
DIVIDE   156 40    156-53    157-17●   183-20    183 22    183 24
DLL      102-27●   129-40●   129-55●   129-56●   139-36    197 5●
DLLADR   102 32●   129-50●   129-51●   129-59    129-60
DLLDR    102 29●   129-41●   139-38    139-39
DLLNAM   102 34●   129 44●   129-45●   139 39    180-54    180 56
DLLR     102 31●   129-43●
DLLSIZ   102-33●   129-52●   129-57    129-58●   129 59●
DLLV     102 30●   129 42●
DMFRST   99 32●   117 24    119 7
DMMAIN   99 31●   163 6     163 8     163 31
DMOVRL   99 30●
```

```
DMPROG 100-180  106-207  106-208  117 25*  118 19*  141 4    141-8    143-31   143-32   145-10   145-11   162-67   162-70   162 72
       162-73   163-5    163-7
DMTRLN 99-290
DONE   125 18   144-140
DSPSIZ 124 10   125-?00
DT.AVL 98-120   138 25   139-26   186-91   190-11   190-44   214 38
DT.UNT 98-100
DTABS  100 140  192-4*   192-5*   195-4    195-32*  195-33*
DU.DFL 96-230
DU.FTL 96 260
DU.INF 96-240
DU.QUE 96-220
DU.SPC 96-270   124-6
DU.TER 96-250
DUP    90 140   166-21
E$END  83 3740
E$LOAD 83 3740  83-427
EF.BBR 93-290
EF.BBU 93-300
EF.CON 88-570   189-15
EF.LOG 93-310
EF.NEW 88-570
EF.PWR 88-570   189-22
EF.RES 88-570   189-9
EF.SEX 93 320
EF.STA 88-570   189-3
EN     1-220    1-38     1-42     1-117    83-357   83-426   83-447   87-14    101-6    103-24   117-19   118-1    121-30   162-16
       162-74   169 22   178-12   179-11   179-20   200-11   200 15   214-19   214-40   214-54   219-17   220-23   220-49
EO     1-380    85-8     103-85   103 90   104-20   106-64   115-17   115-22   140-61   162-63   162 88   162-92   183-38   203-13
       203-53   213-2
EPR.SZ 114-20   115-140
ERR.TB 114-23   115-40   115-14
ERR.TN 106-1930 161-44
ERR001 106 260  198-6
ERR002 106-300  198-13
ERR003 106 340  198-20
ERR004 106-380  107-7
ERR005 106-430  180-118
ERR006 106-560  190-36
ERR007 106-470  117-30
ERR008 106-600  198-27
ERR014 106-520  162-57   162-61
ERR021 106-750  176-45
ERR022 106-860  176-38
ERR023 106-910  174-86
ERR024 106-1050 174-63
ERR025 106-1090 175-105
ERR026 106-1140 206-27
ERR027 106-1180 184-21
ERR028 106-1220 207-66
ERR029 106 1260 207-73
ERR030 106-1310 121-24
ERR031 106-1350 121-46
ERR032 106-1390 124-12
ERR033 106 1440 123-22
ERR034 106 1480 165 4
```

F 7

```
ERRO35  106-152♦ 154-25
ERRO36  106-157♦ 169-41
ERRO37  106-161♦ 169-50
ERRO38  106-165♦ 175-35   204-13
ERR23A  106-94♦  106-101
ERR23B  106-95   106-99♦
ERR23C  106-98   106-102♦
ERRBLK  100-8♦
ERRC    112-42   112-49   116-16♦
ERRD    112-51   116 31♦
ERRLIM  103-61♦  142-55
ERRMC   125-16   142-30♦
ERRME1  103-59♦  112 47   114-26
ERRMES  125-15   140-25♦  142-31
ERRMSG  100-8♦
ERRMSL  140-66   141-2♦
ERRMSX  140 58   140-68   141-21   141-23♦
ERRNBR  100-8♦
ERRTYP  100-8♦   161-35
ERRVEC  88-61    175-28   175-32   182-15   204-5    204-10
EVL     88-57♦
F$AU    83-374♦ 202-9    202-35
F$AUTO  83-374♦ 199-10   199-18
F$BGN   83-374♦  83-401   87-31    88-51    106-26   106-30   106-34   106-38   106-43   106-47   106-52   106-56   106-60   106 75
        106-86   106-91   106-105  106-109  106-114  106-118  106-122  106-126  106-131  106-135  106-139  106-144  106-148  106 152
        106-157  106-161  106-165  106-193  171-11   172-19   181-6    181-12   185-1    186-41   186-47   187-9    189-1    197-31
        199-10   200-8    200-28   201-8    202-9    202-36   203-52   203-54   203-70   204-2    204-2    204-17   205-7    205 7
        205-16   206-6    206-6    206-33   207-6    207-6    207-79   208-2    208-2    208-5    209-2    209-2    209-8    210 2
        210-2    210-9    210-29   211-3    211-40   212-3    212-34   213-5    216-54   216-73   217-2    218-43   218-53   219-12
        220-9    220 9    220-101  221-15   221-16   221-16   221-23   221-24
F$CLEA  83 374♦ 200-8    200-43
F$DU    83 374♦ 201-8    201-34
F$END   83 374    83-374   83-374   83-374   83-374   83-374   83-374   83-374   83-374   83-374   83-374   83-374   83-374   83-374
        83-374    83-374   83-374♦  83-401   87-31    88-51    106-28   106-32   106-36   106-40   106-45   106-49   106-54   106-58
        106 62    106-84   106-89   106-103  106-107  106-111  106-116  106-120  106-124  106 128  106-133  106-137  106-142  106 146
        106 150   106-155  106-159  106-163  106-167  106-214  171-13   172-22   181-10   181-14   185-1    186-47   186-169  186-200
        197-31    198 43   199-18   200-28   200-43   201-19   201-34   202-20   202-35   202-36   203-52   203 54   203-54   203 54
        203 70    204-2    204-2    204-17   204-17   205-7    205 7    205-16   205-16   206-6    206-6    206-33   206-33   207 6
        207-6     207-79   207-79   208-2    208-2    208-5    208-5    209-2    209-2    209-8    209-8    210-2    210-2    210-9
        210-9     210-29   210-29   211-3    211-3    211-3    211-40   211 40   212-3    212-3    212-3    212 34   212 34   213-5
        213-5     213-5    216-54   216-73   216-73   217-2    218-43   218-79   220-32   220-101  221-15   221-16   221 23   221-24
F$HARD  83-374♦ 218-53   218-79
F$HW    83-374♦ 86-10    86-27
F$INIT  83-374♦ 189-1    198-43
F$JMP   83-374♦ 186-169  186-169  197-31   200-28   201-19   201-19   202-20   202-20   203-70   216-54
F$MOD   83 374♦ 83-401   87-31    88-51    185-1    186-41   202-36   203-52   217-2    218-43   220-101
F$MSG   83 374♦ 106-26   106-28   106-30   106-32   106-34   106-36   106-38   106-40   106-43   106-45   106-47   106-49   106-52
        106 54    106-56   106-58   106-60   106-62   106-75   106-84   106-86   106-89   106-91   106 103  106-105  106-107  106-109
        106-111   106-114  106-116  106-118  106 120  106-122  106-124  106-126  106-128  106 131  106-133  106-135  106-137  106-139
        106 142   106-144  106-146  106 148  106-150  106-152  106-155  106-157  106 159  106 161  106 163  106 165  106 167  106 193
        106 214
F$PROT  83-374♦ 187-9    187-15
F$PWR   83-374♦
F$RPT   83-374♦ 186-47   186-200
F$SEG   83 374♦
F$SOFT  83-374♦ 219 12   220 32
```

PARAMETER CODING MACRO V05.00 Wednesday 04-Jan 84 16:12 Page S 8
Cross reference table (CREF V05.00)

SFQ 0291

```
F$SRV    83-374♦ 171-11   171 13   172 19   172-22   181-6    181-10   181-12   181-14
F$SUB    83 374♦ 204-2    204-17   205-7    205-16   206-6    206-33   207-6    207 79   208-2    208-5    209-2    209-8    210-2
         210 9
F$SW     83-374♦ 87-10    87-30
F$TEST   83 374♦ 203-54   210-29   211-3    211-40   212-3    212-34   213-5    216-73
FDATA    101 10♦ 180-45   180-56   180-62   180-68   180-106♦ 180-110♦
FFREE    100-10♦ 106-92   108-17   108-21♦  117-17♦  128 29   128-41   128-45   129 50   129-58   129 60♦  163-16   174 48   174-78
         175-19  191-30♦  191-31   192-4    192-16   196-36   210-3    210-8♦   211-13   211-18♦
FILOPN   101-11♦ 179-12   179-15♦  180-23   180-26♦
FMEM     100-12♦ 117-17   196-36♦
FMEMS    100 13♦ 117-18   196 37♦
FMERR    107-7♦  108 19   174-44
FNAME    101-7♦  180-25
FNUM     100 30♦ 117 22   119-8    141-11♦  180-22♦  180-99♦  191-33♦  214-21
FRMTT    103 54♦ 110-10
FS       1-21♦   1-38     1-40     1-43     1-63     83-358   83-427   83-447   101-7    103 25   117-19   118-1    178-12   179-11
         179-20  200-11   200-15   214-19   214-40   214-54
FSIZE    100-11♦ 108-18♦  117-18♦  128-30   128-43   129-52   129-57♦  174-42   191-31♦  196-37   210-3    210 8♦   211-13   211-18♦
FWORD    180-44  180-46   180-61   180-63   180-103♦
G$CNTO   83-374♦
G$DELM   83-374♦
G$DISP   83-374♦ 220-9
G$EXCP   83 374♦
G$HILI   83 374♦
G$LOLI   83-374♦
G$NO     83-374♦ 130-64   196-27
G$OFFS   83-374♦ 130-64   196-27   218-65   218-67   218-69   218-71   218-73   218-77   220-4    220 12   220-15   220-17   220-19
         220-21
G$OFSI   83-374♦ 130-64   196-27   218 65   218-67   218-69   218-71   218 73   218-77   220-4    220-12   220-15   220-17   220-19
         220-21
G$PRMA   83 374♦ 218 65   218-67
G$PRMD   83-374♦ 130-64   218-69   218-71   218-73   220-12   220-15
G$PRML   83-374♦ 196-27   218-77   220-4    220-17   220-19   220-21
G$RADA   83-374♦ 130-64
G$RADB   83 374♦
G$RADD   83-374♦ 218-69   218-71   218-73   220-12   220-15
G$RADL   83-374♦ 196-27   218-77   220-4    220-17   220-19   220-21
G$RADO   83-374♦ 218 65   218-67
G$XFER   83-374♦
G$YES    83-374♦ 218-65   218-67   218-69   218-71   218-73   218 77   220 4    220-12   220-15   220-17   220-19   220-21
GETCDN   155-17  155-19   155-29♦
GETCNT   113-15  114-10   114-19   114-32   155-14♦  156-16
GETCNX   155-16♦ 155-28
GETCXX   155-30  155-32♦
GTDRVT   130-41  133-4    134 24   135-17   136-19   137-14   138 21   142-37   143-22   154-14♦  161-22   190-41
GTT452   118-37  119-6♦
H.BPL    193-18  194-19   218-57♦  218-69   218-69   218 69
H.BST    193 24  194-24   218-58♦  218-71   218-71   218-71
H.DRV    190-40  195-9    195-19   218-59♦  218 73   218-73   218-73
H.PRM    195-21  218-60♦  218-77   218 77   218-77
H.UBA    193 15  218-55♦  218-65   218-65   218 65
H.VEC    193-21  193 30   194-22   218 56♦  218-67   218-67   218 67
HC.BF1   90-44♦  90-45    124 17
HC.BF2   90-45♦  90-47    106 185  124 4    124 8♦   124 9    124 36
HC.BSZ   90 24♦  90 45    90 47    124 20   141 13   141 17   163 11   163 17   168 20   168 22   213 25
HC.CCT   90-32♦  167 19♦
```

```
HC.CEV   90 374   166-19
HC.CMD   90 314   90-32    90-34    109-23*
HC.CPK   90-384   90-44    106-171  109-22   162-70*  162-71*  162-72*  162-73*  164-19*  164-20*  164-21*  164-22*  166-30*  167-17*
         168 19*  168-20*
HC.ESZ   90-224   90-35
HC.INT   90-264   90-28
HC.ISZ   90-204   90 28    174-41
HC.MCT   90-294   167-18*
HC.MEV   90 344   90-35    90-37
HC.MPK   90-354   90 38    106-173  109-20   109-22   123-10   123-15   123-20   162-84   164-26
HC.MSG   90-284   90-29    90-31    109-21*  175-20
HC.PSZ   90-234   90-44    166-20   166-26
HC.RSZ   90 214   90-31    90-34
HC.SIZ   90-474   109-15
HCOMM    109 154  162-62
HELP     83 3494  83-369   83-391   83-409   83-437   85-12    86-18    87-22    88-44    98-41    103-16   103-35   184-28   184-37
         184 44   186-44   186-157  186-187  197-7    198-30   199-12   200-19   200-30   201-10   201-21   202-11   202 22   203-5*
         210 11   210-16   210-31   211-22   211-27   211 42   212-16   212-21   212-36   216-55   216-60   216 75   218-5*   218-97
         218-107  220-61   220-70   220-94   221-2
HM.CYL   195-24   196-4    218-624  218-77
HOE      88-574
I$AU     83-3744  202-94           55*
I$AUTO   83-3744  199-104  199-184
I$CLN    83-3744  200-84   200-28   200-43*
I$DU     83-3744  201-84   201-344
I$HRD    218-534  218-794  220-9
I$INIT   83-3744  189-14   197-31   198-43*
I$MOD    83-3744  83-401   83-4014  87-31    87-314   88-51    88-514   185-1    185-14   186-41   186-414  202-36   202-364  203-52
         203-524  217-2    217-24   218-43   218-434  220-101  220-1014
I$MSG    83-3744  106-264  106-284  106-304  106-324  106-344  106-364  106-384  106-404  106-434  106-454  106-474  106-494  106-524
         106-544  106-564  106-584  106-604  106-624  106-754  106-844  106-864  106-894  106-914  106-1034 106-1054 106-1074 106-1094
         106-1114 106-1144 106-1164 106-1184 106-1204 106-1224 106-1244 106 1264 106-1284 106-1314 106-1334 106-1354 106-1374 106-1394
         106-1424 106-1444 106-1464 106-1484 106-1504 106-1524 106-1554 106-1574 106-1594 106-1614 106-1634 106-1654 106-1674 106-1934
         106-2144
I$PROT   83-3744  187-94
I$PTAB   83-3744  221-16   221-164  221-23   221-234
I$PWR    83 3744
I$RPT    83-3744  186-474  186-2004
I$SEG    83-3744  203-54   204-2    205-7    206-6    207-6    208-2    209-2    210-2    211-3    212-3    213-5
I$SETU   83-3744  221-15   221-154  221-16   221-24   221-244
I$SFT    219-124  220-9    220-324
I$SRV    83-3744  171-114  171-134  172-194  172-224  181-64   181-104  181-124  181-144
I$SUB    83-3744  203-54   204-2    204-24   204-17   204-174  204-174  205-7    205-74   205-16   205-164  205-164  206 6    206-64
         206 33   206-334  206-334  207-6    207-64   207-79   207 794  207-794  208-2    208-24   208-5    208-54   208 54   209-2
         209-24   209-8    209-84   209-84   210-2    210-24   210-9    210-94   210-94   211-3    212-3    213-5
I$TST    83-3744  203-54   203-544  203-70   204-2    205 7    206-6    207-6    208-2    209-2    210-2    210 29   210 294  210-294
         211-3    211-34   211-40   211-404  211-404  212-3    212-34   212-344  212-34   212-344  213-5    213-54   216-54   216-73
         216-734  216-734
IBE      88-574
ICONT    100-254  189-19   213-16
IDU      88 574   138-51   140-39   142-46
IER      88 574
IFLAGS   100-234  133 16   189-6*   189 12*  189 18*  189 19*  213 16   213 18*  214 13   216-25*  216-26*
INIT1    189-7    189-13   191-5*
INIT2    192 44
INIT3    192-164
```

```
INIT4    193-4@
INIT5    196-4@
INITBL   175-42   175-62@
INITWA   103-64@  196-6
INITWB   103-65@  196-15
INITWC   103-48@  196-27
INITXX   190-55   197-4@
INTRCV   100-35@  181-13+  207-26+  207-35   207-48+  207-62
INTSRV   181-12@  207-21
INTSTO   103-62@  207-18
INTST1   103-63@  207-75
IPADRS   101-25@  182-17   192-19   194-4    194-5
IREST    100-26@  189-12   214-13
ISR       88-57@
ISTRT    100 27@  189-6    214 13
ISTRTH   100-28@  133-16   189-18   216-26
IXE       88 57@
J$JMP     83-374@ 186-169  201-19   202-20
KTBASA   100-20@
KTBASO   100 21@
KTMEM    102-12@
KW.BRL   102-5@   191-16+
KW.CSR   102-4@   121-39   122-4    169-34   176-30   181-9+   182-23   182-25+  183-14   184-14   191-11+  191-15+  191-21+  207 40
KW.EL    102-8@   121-41   121-44   122-6    122-9    169-36   169-39   173-55   173-56   173-57   176-32   176-35   181-7+   181-8+
         183 17   183-18   184-16   184-19   191-5+   191-6+   207-42   207 45
KW.HZ    102-7@   173-31   183-19   191-18+
KW.OUT    88-62   181-9    182-25   191-21
KW.VEC   102-6@   191-17+  191-20
KW11I    181-6@   191-20
L$ACP     83-427@
L$APT     83-427@
L$AU     202 9@
L$AUT     83-427@
L$AUTO    83-427   199 10@
L$CCP     83-427@
L$CLEA    83-427   200-8@
L$CO      83-427@
L$DEPO    83-427@
L$DESC    83-427   103-25@
L$DESP    83-427@
L$DEVP    83-427@
L$DISP    83-427    85-8@
L$DLY     83-427@
L$DTP     83-427@
L$DTYP    83-427@
L$DU     201-8@
L$DUT     83-427@
L$DVTY    83-427   103-14@
L$EF      83-427@
L$ENVI    83 427@
L$ERRT    83 427   100-8@
L$ETP     83-427@
L$EXP1    83-427@
L$EXP4    83-427@
L$EXP5    83-427@
L$HARD    83-427   218 53   218-53@
```

J7

```
L$HIME  83 4270 128-54
L$HPCP  83-4270
L$HPTP  83-4270
L$HW    83-427   86-10    86-100
L$ICP   83 4270
L$INIT  83-427  189 10
L$LADP  83 4270
L$LAST  83-427  220-1000 221-24
L$LOAD  83-4270
L$LUN   83-4270 120 29*  121-12*  154-31*  161-18*  203-63*
L$MREV  83-4270
L$NAME  83-4270
L$PRIO  83-4270
L$PROT  83-427  187-90
L$PRT   83 4270
L$REPP  83-4270
L$REV   83-4270
L$RPT   83-427  186-470
L$SOFT  83-427  219-12   219-120
L$SPC   83-4270
L$SPCP  83 4270
L$SPTP  83-4270
L$STA   83-4270
L$SW    83-427   87-10    87-100
L$TEST  83-4270
L$TIML  83-4270
L$UNIT  83-4270 190-46   192-6    195-35
L10000  86 10    86 270
L10001  87-10    87-300
L10002 106-280
L10003 106 320
L10004 106-360
L10005 106-400
L10006 106 450
L10007 106 490
L10010 106 540
L10011 106-580
L10012 106 620
L10013 106-840
L10014 106-890
L10015 106 1030
L10016 106-1070
L10017 106-1110
L10020 106-1160
L10021 106 1200
L10022 106-1240
L10023 106-1280
L10024 106 1330
L10025 106-1370
L10026 106-1420
L10027 106 1460
L10030 106 1500
L10031 106-1550
L10032 106 1590
L10033 106 1630
L10034 106 1670
```

K7

```
L10035 106-214*
L10036 171-13*
L10037 172-22*
L10040 181-10*
L10041 181-14*
L10042 186-169   186-200*
L10044 197-31    198-43*
L10045 199-18*
L10046 200-28    200-43*
L10047 201-19    201 34*
L10050 202 20    202 35*
L10051 203-70    210-29*
L10052 204-17*
L10053 205-16*
L10054 206 33*
L10055 207-79*
L10056 208-5*
L10057 209-8*
L10060 210-9*
L10061 211-40*
L10062 212-34*
L10063 216-54    216-73*
L10064 218-53    218-79*
L10065 219-12    220-32*
L10066 221-16*
L10070 221-16    221-23*
LBUFE  102-23*   141-10*   141-14
LBUFN  102-22*   141-7*    141-12    141-13*   141-14    141-22*  213-26
LBUFS  102-21*   117-16*   141-2     141-6*    213-19    213-22*
LDDM   120-23*   120-38
LDNEXT 120-31    120-34    120-36*
LF     88-59
LOAD   163-24    164-16*
LOADB  162-68*   163-32
LOADDM 120-33    162-15*
LOADE1 162-85    164-27    165-4*
LOADER 162-42    162-58    163-25    164-25    165-5*
LOADT1 162-65    163-5*
LOE    88-57*
LOG    90-12*
LOGCHK 213-20    213-31*
LOGM1  103-80*   213-21
LOGM2  103-81*   213-28
LOGM3  103-82*   213-33
LOGOUT 213-23*   213-27
LOT    88 57*
LPNT   111-7     111-10    111-13    111-17*  183-25   183-29   183-34
LPNTB  106 27    106-31    106-35    106-39   106 44   106-48   106 53   106-57   106-61   106 83   106 88   106 92   106 96   106 102
       106-106   106-110   106-115   106-119  106 123  106 127  106-132  106-136  106-140  106 153  106-158  106-162  106-166  106 169
       106 176   106-183   106-187   106-200  111-9*   148 7    149 7
LPNTF  111 6*    112-47    114-26    130-36   130 45   130-111  191 12   196 6    213-21   213-28   213 33   215-5
LPNTS  111-15*   186-50    186-71
LPNTX  111-12*   138-55    138-57    142 55   143 26   207-18   207 75
LT11   163 13    163-15*
LT1L1  163 11*   163-30
LT1L2  163-19*   163 21
```

L7

```
M1       1-25@    1-34     1 37     1 48     1 366    83-363   83-432   103-28   200-15   218-74   21o-89   220-5    220-38
M2       1-26@    1-35     1-37     1-49     1-450    83-364   83-433   98-27    99-20    103-29   103-42   104-12   104-52   106-42
         106-113  128-28   129-39   130 3C   133-9    136-12   136-23   138-30   138-60   140-26   140-41   140-69   140-74   142 48
         142-57   159-1    161-27   173 7    173-18   173-32   173-64   173-73   186-54   186-75   186-89   186-122  186-150  186-179
         190 49   191-23   195-39   196-22   216 27   216-38   220-6    220-39   220-83
M3       : 29@    1-34     1-37     1-50     1-727    83-365   83-434   103-32   200-15   218 74   218-89   220-5    220-38
M4       1-30@    1-35     1-37     1-51     1-811    83-366   83-435   98-27    99 20    103-33   103 42   104-12   104 52   106-42
         106 113  128-28   129-39   130-30   133-9    136-12   136-23   138-30   138 60   140-26   140-41   140-69   140-74   142-48
         142-57   159-1    161-27   173-7    173-18   173-32   173-64   173-73   186 54   186-75   186-89   186-122  186-150  186-179
         190-49   191 23   195-39   196-22   216-27   216-38   220-6    220 39   220-83
MC       1-37@    97-15    105 9    120-24   174-92
MO       1-34@    1-36     85-9     103-85   115-17   117-42   140-61   162-63   162 92   183-38   203-13   203-53
MO.CMP   93-5@
MO.CWB   93-23@
MO.ERR   93-7@
MO.EXP   93-6@
MO.FEU   93-17@
MO.IMF   93-21@
MO.NXU   93 19@
MO.PRI   93-24@
MO.RIP   93-20@
MO.SCH   93-8@
MO.SCL   93-9@
MO.SEC   93-10@
MO.SEQ   93-15@
MO.SER   93-11@
MO.SPD   93-16@
MO.SSH   93 12@
MO.SWP   93-22@
MO.VOL   93-18@
MO.WBN   93 13@
MO.WBV   93-14@
MEMFIL   128-34@   128 37
MESSAG   125 17    143-19@
MESSG    103-67@   138-55    143 26
MLDRER   195 11    198-12@
MM       1-36@     220 76
MSCP     90-11@
MSGBRL   218-69    218-86@
MSGBST   218-71    218-87@
MSGCST   218-77    218-90@
MSGLDR   218-73    218-88@
MSGPKL   106-187@  106 190
MSGPKT   106-141   106-154   106-183@
MSGUBA   218-65    218-84@
MSGVEC   218-67    218-85@
MX       1 35@     1-36     85-10    103-90   115-22   117 42   186 63   213-2    213-10   214 7    216 28
MXFERE   138-22    138-59@
MXFERP   103-60@   138-57
MXFERX   138 27    138-47    138-49   138 52   138-58@
NCON     106 ¿10   106-212@
NCONF    112-44    112-49@
NCONS    112-43@   112-46
NOCLOC   103-79@   191-12
NODOCU   1 926     81-7
NXMAD    102-11@   171-12*   175-26*  175-33   182-12*  204 3*   204-11
```

```
NXMI    171-110   175-28    182 15    204 5
O$APTS  83-3740   83-427
O$AU    83-3740   83-427
O$BGNR  83-3740   83-4070   83-427
O$BGNS  83-3740   83-4070   83-427
O$DU    83-3740   83-427
O$ERRT  83-3740   83-4070   83-427
O$GNSW  83-3740   83-4070   83-427
O$POIN  83-3740   83-407    83-4070   83-4070   83-4070   83-4070   83 4070   83-427
O$SETU  83-3740   83 4070   83-427    220 100
ONEFIL  1-50      1-9       81 12     82-1      83-3530   83-395    87-32     88 1      88 80    88-13    185 2    186-1    186-80   186-13
        202 37    203-1     203-90    203 16    217-3     218 1     218-90    218-15
OP.ABO  92-30
OP.ACC  92-40
OP.AVA  92 220
OP.AVL  92 50
OP.CCD  92-60
OP.CMP  92-70
OP.DUP  92-230
OP.ELP  92-290
OP.END  92 200    123 6     123-9
OP.ERS  92 80
OP.ESP  92 280    162-68
OP.FLU  92-90
OP.GCS  92 100
OP.GSS  92-270
OP.GUS  92-110
OP.MRD  92-180
OP.MWR  92-190    164-17    166-22
OP.ONL  92-120
OP.RD   92-130
OP.RLC  92-250
OP.RPL  92-140
OP.RSD  92 310    123 9     124 34
OP.SCC  92-150
OP.SEX  92 210
OP.SHC  92-240
OP.SSD  92-300    123-6     124-15
OP.SUC  92 160
OP.WR   92-170
OSTRE   112-41    112 48    112-530
OSTRNG  106-213   111-22    112-400   112-52    143-36    145 12    150-8    151-8    152-8
P.BCNT  94-100    95-110    162-710   164-200   168-200
P.BUFF  94-110
P.CMST  95-180
P.CNCL  95-570
P.CNTF  94-380    95-550
P.CNTI  95-580
P.CPSP  94-270
P.CRF   94-60     95-60     123-20    167 170
P.CTMO  95 560
P.CYLS  95-320
P.DEXT  95-630
P.DFLG  95-640
P.DMDT  94 520
P.DPRG  95-650
```

```
P.DTMO  95-66#
P.ELGF  94-25#
P.FBBK  95-12#
P.FLGS  95-9#
P.GRPS  95-31#
P.HSTI  94-24#   95-25#    95-43#
P.HTMO  94 39#
P.LBN   94-13#
P.MEDI  95-27#   95-45#
P.MLUN  95-23#   95-41#
P MOD   94-9#
P.OPCD  94-8#    95-8#     123-10    166-30*
P.OTRF  94-18#   95 17#
P.OVRL  94-53#   162-72*   162-73*
P.RBN   94-32#
P.RBNS  95-34#
P.RCTC  95-35#
P.RCTS  95-33#
P.RGID  94-46#   164-22*
P.RGOF  94 47#   164-21*
P.SHST  95-29#   95-47#
P.SHUN  94-26#   95-28#    95-4o#
P.STS   95-10#   123-15    162-84    164-26
P.TIME  94-41#
P.TRKS  95-30#
P.UADR  94-12#   162-70*   164-19*   168-19*
P.UNFL  94-23#   95-24#    95-42#
P.UNIT  94-7#    95-7#
P.UNSZ  95-48#
P.UNTI  95-26#   95-44#
P.USEF  94-40#
P.VRSN  94-37#   95-54#
P.VSER  95-49#
PAT16C 101 34#   131-20    214-53    214-59
PAT16W 101-35#   214-59
PB     110-20#   111-9
PF     110-18#   111-6     111-27
PNT     88-57#
PNTERR 140-67    161-15#   213-24
PNTNUM 113-27    113-33    114-5     156-15#
PNTNUS 146-13    156-17#
PNTPKL 106-176#  106-180
PNTPKT 106 145   106-149   106-169#
PRI     88-57#
PRIOO   88-57#   197-4     207-21    207-22    207-69
PRIO1   88-57#
PRIO2   88-57#
PRIO3   88-57#
PRIO4   88-57#
PRIO5   88 57#
PRIO6   88 57#
PRIO7   83-427   88-57#    162-36    175-28    182-15   191-20   204-5   207 49
PRINTC 106-203   110-8#    113-8     113-16    114-11   114-33   130 52  130-57  130 60  146-14  146-22  148 9  149-9  15o 59
       156-67    159-4     159-12    159-28    183-28   183-33   183-36  186 52
PS     110 24#   111-15
PTYPE  106-212*  110-15    111-6*    111-9*    111-12*  111-15*  111-27#  148 10* 149 10*
```

```
PX       106 212   110 220   111 12    148 10    149 10
RDDAT    180 40    180 42    180 61#
RDDLL    129 53    178 16#
RDERR    180-43    180 71    180 74    180 113#
RDREC    117 26    119-10    178 17    180 21#    214 24
RDST     180 28#   180 33    180 55    180 57    180-59    180 88    180 97
RDSTS    180 24    180 27#
RESET    117-15    182 12#   191 29    200 16    203-72    211 12
RESPCT   121 9#    122 18
RESPDM   121 7#    122 19    129 65    210 7     211-17    212 14   213-36   216 46
RG.FLG   90 6#     167 18
RG.OWN   90-5#     167 18    167 19
RNT4DM   118 16#   216 39
RNTIM    103-56#   183 25
RNTIM1   103 57#   183 29
RNTIM2   103-58#   183-34
RNTIME   106 202   138 56    143 27    148 8     149 8     183 14#   186 51
RNTIMX   183-15    183-36#
RPTCT    186-74#   186-154
RPTCTN   186-76    186-87    186 148#
RPTDT    186 86#   186-147
RPTDTN   186-90    186-146#
RPTMD2   186-121   186 178#
RPTMSD   186 119   186 177#
RPTMSG   186-50    186-174#
RPTMSH   186-71    186-175#
RPTXX    186-69    186-156#
RSP.CK   175-74    175-83    175-91    175-101#
RSP.S1   175-62    175 73#
RSP.S2   175-64    175-79#
RSP.S3   175-66    175-88#
RSP.S4   175-68    175-96#
RSPDRP   121-25    121-47    122-23#   123 23    124-13    124-25
RSPDSP   124 24    125-4#    125-20
RSPERR   123-11    123-16    123-22#
RSPIN    121-14    123-6#
RSPMWR   123-8     123-10#
RSPNRP   122-5     122-8     122-10    122-16#
RSPNTO   121-40    121-43    121-45    121-48#
RSPNXT   121-11    122-4#    122 25    124 48
RSPOU    121-16    123-28#
RSPOU2   124-32    124-41#
RSPOU3   124 39    124-42#
RSPOUT   123-28    124-30#
RSPPT2   124-4#
RSPPT3   124-11    124-15#
RSPPTW   123-21    123-27#
RSPRPT   122-7     122-11#
RSPTM    121-22    121-29#
RSPTMO   121-42    121-46#
RUNDM    120 16#   129-63    210-5     211-15    212 12
RWORDT   180-65    180-91#
RWRDE1   180-35    180-51    180-89    180-95    180-118#
S$LSYM   83-374#   86-27#    87-30#    106-28#   106 32#   106-36#   106-40#   106-45#   106-49#   106-54#   106-58#   106 62#   106-84#   106 89#
         106 103#  106-107#  106-111#  106-116#  106-120#  106 124#  106 128#  106-133#  106 137#  106-142#  106-146#  106-150#  106 155#  106-159#
         106-163#  106 167#  106 214#  130-64    130-64    130 64    130 64#   171-13#   172-22#   181-10#   181 14#   186 200#  196 27   196 2*
```

	196 27	196-270	198 430	199 180	200 430	201 340	202-350	204-170	205-160	206 330	207 790	208-50	209 80	210 90
	210-290	211-400	212 340	216 730	218 790	220-320								
S.EL	220 12	220-430												
S.IW	220-19	220-460												
S.LOG	220 21	220-470												
S.MAN	220-4	220 370												
S.MES	220-9	220 400												
S.SSF	220-17	220-450												
S.XL	220 15	220 440												
SA.A2	89-340													
SA.BST	89-720													
SA.CM1	89-280	209-3												
SA.CMO	89-230													
SA.CME	89-480													
SA.CNT	89 780	162-49	162 54											
SA.CTP	89-510													
SA.DI	89-350	175 73												
SA.ERC	89-150													
SA.ERR	89-60	106 87	176 20											
SA.GO	89-700	174 104												
SA.INE	89 630													
SA.INT	89-210	207-25												
SA.LFC	89-710													
SA.MCV	89-770	162-48												
SA.MS1	89-270	209-3												
SA.MSE	89-470													
SA.MSG	89 220													
SA.NV	89-330													
SA.NVE	89 640													
SA.PRG	89-410													
SA.S1	89-100	175 41	175-73	205-9	206-10	207 28								
SA.S2	89 90	175-82	205 13											
SA.S3	89-80	175-90												
SA.S4	89 70	175-51												
SA.STE	89 500													
SA.STP	89-250	175-17	205 12	206 14	206 17	207 25	209 3							
SA.TST	89-570	175-21												
SA.VCE	89-620													
SA.VEC	89-200													
SA.WRP	89 240	206-14	206-17											
SAMVEC	193-32	198-260												
SEKERE	137-15	137-180												
SETOO	173-460	173-51												
SETO1	173-47	173-490												
SETO2	173-550	173-58												
SETTO	124-47	169-16	173-280	176-24	184-9	186 58	190 53	191 27	207 34					
SFPTBL	87-100	130-31	133-18	138-46	138-48	140 57	140 65	142 43	213-31	214 15	216-47			
SM.IW	133-18	219-290	220 19											
SM.LOG	140-65	213-31	216-47	219 280	220 21									
SM.MAN	130-31	214-15	219-260	220 4										
SM.SSF	140 57	219-270	220-17											
SND.S1	175-180	175-630	175-79	175 88										
SND.S2	175-190	175 200	175-650											
SND.S3	175 210	175-670												
SNDCMD	124 43	162-82	164-23	167-150										
SO.BIT	130-31	133 18	140 57	140-65	213-?	214-15	216 47	219 160	220 4	220-4	220-4	220-17	220-17	220-1?

```
          220 19   220-19   220-19   220 21   220-21   220 21
SO.EL     142 43   219-14•  220-12   220-12   220-12
SO.XL     138 46   138-48   219-15•  220-15   220-15   220-15
SSTEP4    162 46   175-97•  175-109•
ST.ABO    96 7•
ST.AOL    96 18•
ST.AVL    96 9•
ST.CMD    96 6•
ST.CMP    96 12•
ST.CNT    96-15•
ST.DAT    96 13•
ST.DIA    96-17•
ST.DRV    96 16•
ST.HST    96 14•
ST.MFE    96 10•
ST.MSK    96-3•   123 15   162-84   164-26
ST.OFL    96 8•
ST.SUB    96 4•
ST.SUC    96-5•
ST.WPR    96-11•
STIME     102-9•  122-6    122-9    186-53   19C 48   191-22
STLDOM    118-38  120-21•
STORAG    84 8•   117-24   117-25   118-19   119-7    180-70   180 73   214 41   214-55
STOSIZ    84-5•   84-8     180-73
SVCGBL    83-374• 83-383•  83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83 427   83-4'7   83-427   83-427
          83-427  83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83 427   83-427   83 427
          83-427  83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427
          83-427  83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83 427   83-427   83-427   83-427
          83-427  83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83 427   83-427   83-427   83-427
          83-427  83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83 427   83-427   83-427
          83-427  83-427   83-427   83 427   83-427   83-427   83 427   83-427   83 427   83-427   83-427   83-427   83-427   83-427
          83-427  83-427   83-427   83-427   85 8     85-8     85-8     86-10    86 10    86-10    86-10    86-10    86-10    87-10
          87 10   87-10    87-10    87-10    87-10    100-8    100-8    100-8    103-14   103-14   103-14   103-25   103-25   103 25
          106 16• 106-26   106-26   106 26   106-30   106-30   106 30   106 34   106-34   106-34   106-38   106-38   106-43
          106-43  106-43   106-47   106-47   106-47   106 52   106 52   106 52   106-56   106 56   106-56   106-60   106-60   106-60
          106-75  106-75   106-75   106-86   106-86   106-86   106 91   106-91   106-91   106 105  106-105  106-105  106-109  106-109
          106-109 106-114  106-114  106-114  106-118  106 118  106 118  106 122  106-122  106-122  106-126  106 126  106-126  106-131
          106-131 106-131  106-135  106-135  106 135  106-139  106 139  106 139  106-144  106-144  106-144  106-148  105 148  106 148
          106-152 106-152  106-152  106-157  106 157  106 157  106 161  106 161  106 161  106-165  106-165  106 165  106-193  106-193
          106-193 106-224• 171-11   171-11   171-11   172-19   172 19   172 19   181-6    181 6    181 6    181 12   181-12   181-12
          186-47  186-47   186-47   187-9    187-9    187 9    189 1    189-1    189-1    199 10   199-10   199-10   200-8    200-8
          200-8   201-8    201-8    201-8    202-9    202-9    202-9    218 53   218-53   218 53   219-12   219-12   219 12   220-100
          220-100 220-100  220-100•
SVCINS    83-374• 83-380•  83-427   83-427   83-427   83 427   83-427   83-427   83-427   83-427   83-427   83-427   83-427   83-427
          83-427  83-427   83-427   83-427   83-427   83 427   83-427   83-427   83-427   83 427   83-427   83-427   83 427   83-427
          83 427  83-427   83-427   83-427   83-427   83-427   83 427   83-427   83-427   83 427   83-427   83-427   83-427   83-427
          83-427  83-427   83-427   83-427   83-427   83-427   83 427   83-427   83 427   83-427   83 427   83-427   83-427   83-427
          83-427  83-427   83-427   83-427   83-427   83-427   83-427   83-427   83 427   83-427   83 427   83-427   83-427   83-427
          83-427  83-427   83-427   83-427   83-427   83-427   83-427   83-427   83 427   83-427   83 427   83-427   A3-427   83-427
          83 427  83-427   83-427   83-427   83-427   83 427   83-427   83-42?   83 427   83-427   83 427   83-427   85-8     85-8
          85-8    85-8     85-8     85-8     85-8     85 8     85-8     85-8     86 10    86 10    87-10    87 10    103-14   103 14
          103-14  103-14   103-25   103-25   103 25   103-25   106-130  106 28   106 28   106 32   106-32   106 36   106-36   106-40
          106-40  106-45   106-45   106 49   106 49   106 54   106-54   106 58   106 58   106-62   106-62   106 84   106-84   106-89
          106 89  106-103  106-103  106-107  106 107  106-111  106 111  106-116  106 116  106 120  106-120  106 124  106-124  106-128
          106-128 106-133  106 133  106-137  106-137  106 142  106-142  106 146  106 146  106 150  106 150  106 155  106 155  106-159
```