UDA50 KDA50
RA60 RA80/81/82

BAD BLK REPLACE UTIL
CZUDLA0

AH-FF57A-MC
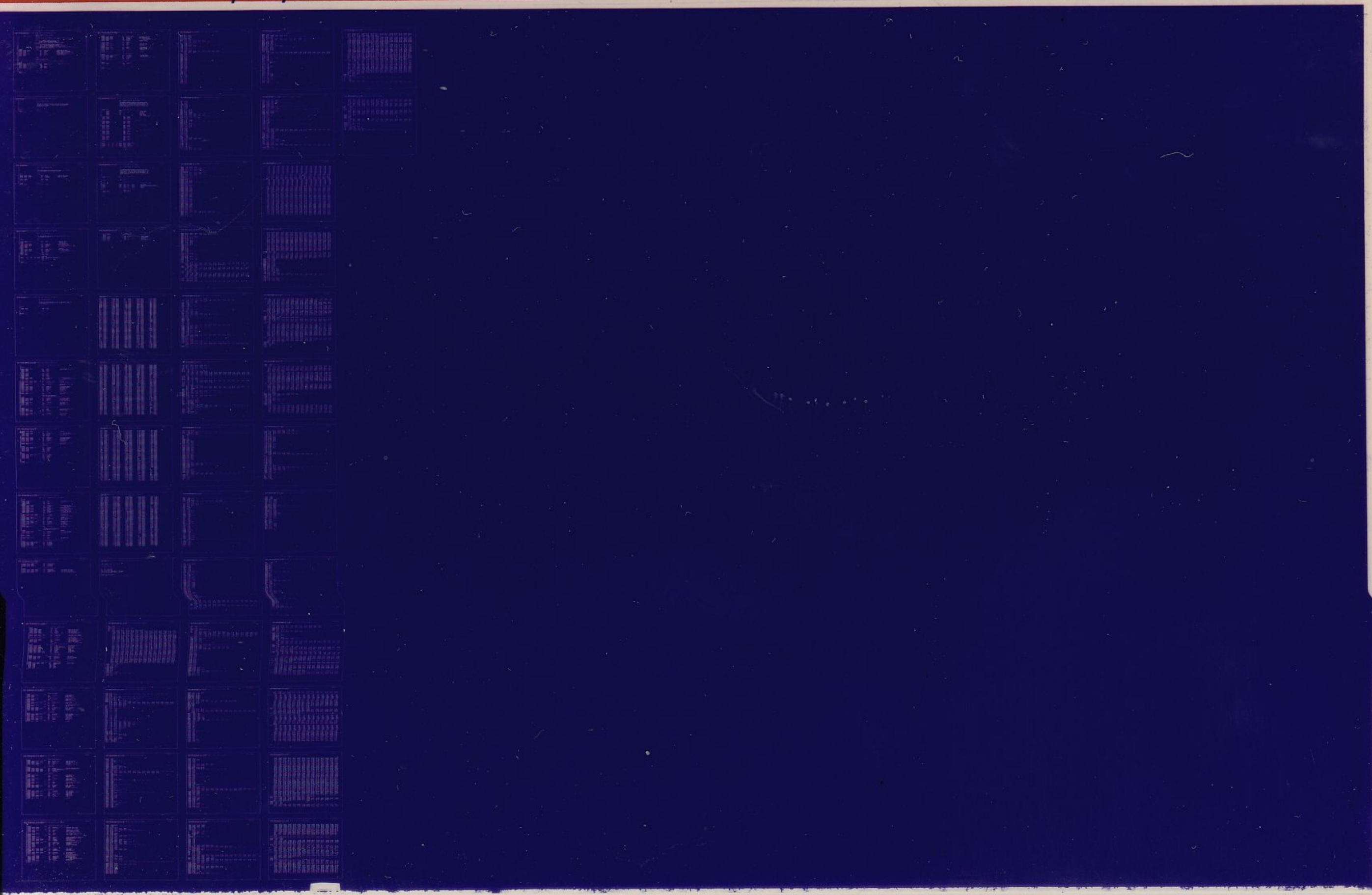1 OF 2   OCT 1985
COPYRIGHT© 1985

digital
MADE IN USA

UDA50 KDA50
RA60 RA80/81/82

BAD BLK REPLACE UTIL
CZUDLA0

AH-FF57A-MC
2 OF 2   OCT 1985
COPYRIGHT© 1985

digital
MADE IN USA

IDENTIFICATION

PRODUCT CODE: AC-FF56A-YC

PRODUCT NAME: CZUDLAO BAD BLOCK REPLACEMENT UTILITY

PRODUCT DATE: 19-APR-1985

MAINTAINER: H. PAUL HOLSINGER

AUTHOR: H. PAUL HOLSINGER

.REM 8

This software is furnished for the purpose of maintaining Digital Equipment
Corporation RA series drives. It should only be used by persons authorized to
perform service on these products. This software or any other copies thereof
should not be provided or otherwise made available to any unauthorized person.

Even though this program is distributed along with diagnostics it is NOT
(repeat not) intended for use as an RA drive diagnostic. This program assumes
that the RA drive subsystem is not defective and that the user only intends to
use this program for its intended purpose of finding and replacing bad blocks.
Using this program on defective hardware could produce indeterminate results
detrimental to the customer and Digital.

INDEX

1.0 INTRODUCTION

This program is a highly specialized utility for use in those cases where an
RA drive has been diagnosed to have bad blocks on the media. These bad blocks
may be causing ECC errors that can be isolated to a specific Logical Block
Number(s) (LBN). Using the Digital Storage Architecture (DSA) Bad Block
Replacement (BBR) algorithm, this program will replace those blocks that it
either defines as bad (Automatic Replacement Mode) or will take an LBN address
that the user supplies and replace only that block (Manual Mode). In either
case, bad block(s) will be replaced, with no apparent loss of storage capacity
to the customer.

WARNING   WARNING   WARNING   WARNING   WARNING   WARNING   WARNING
+---+

```
)           IN ORDER TO PROPERLY EXECUTE THIS PROGRAM IN AUTOMATIC OR MANUAL    )
)           MODE, THE FOLLOWING STEPS MUST BE FOLLOWED WITHOUT EXCEPTION.       )
)           UNPREDICTABLE RESULTS COULD RESULT IF NOT FOLLOWED.                 )
)                                                                              )
)                                                                              )
)1. CUSTOMER MUST BACKUP THE DATA FROM THE DRIVE(S) TO BE USED AND VERIFY       )
)   ITS CORRECTNESS.                                                           )
)                                                                              )
)2. EXECUTE THIS PROGRAM ON THE DRIVE(S). MULTIPLE PASSES ON EACH DRIVE, WHEN )
)   IN AUTOMATIC MODE, IS RECOMMENDED ---- DR>Start/pass:(Number of passes)     )
)                                                                              )
)3. CUSTOMER MUST RESTORE THE BACKED UP DATA TO THOSE DRIVES FROM WHICH IT      )
)   WAS BACKED UP AND AGAIN VERIFY ITS CORRECTNESS.                            )
)                                                                              )
+------------------------------------------------------------------------------+
```

1.1 Hardware Requirements

This program requires the following minimum hardware configuration:

* PDP-11 Unibus or Q-bus processor
* 28K words of memory (minimum)
* Console terminal
* XXDP+ load media containing this program
* One or more UDA50-A or KDA50-Q subsystems.
* One or more DSA disk drives.

* A system clock - either type L or P - will be used to
  report runtime, and to detect controller MSCP response
  timeouts.

1.2 Software Requirements

This program is designed to run using XXDP+ V2.0 or greater. It has
been assembled using the SVC35R MACRO library.

1.3 Memory Requirements

The utility occupies approximately 20k bytes of memory. Each drive
configured for test will require about 1k bytes additional memory.

2.0 MODES OF OPERATION

This program will run in one of three modes.

2.1 Verify Mode
This allows the user to "scan" the media without the program taking
any action on what it finds. In this way, you can get a "picture"
of any problem found, without the program writing in any way on the
media. This is extremely valuable, since the drive can be write
protected and thus the lengthy Backup and Restore operations
recommended need not be done. By getting a "look" at the condition
of the media, you can then decide whether to commit to either

Automatic or Manual mode and do the backup and restore of the customer
data. This mode can also be used to get the replacement descriptors
displayed (By answering the question "Display Replacement Descriptors"
with a yes) without the Backup and Restore operations. This way you
can get a "look" at the total number of replacements quickly.
Execution in this mode is entered by:
1) WRITE PROTECT THE SUBJECT DRIVE --- AFTER THE DRIVE HAS BECOME
   READY. SPINNING UP AN RA80 WITH IT WRITE PROTECTED WILL
   RESULT IN A FAULT, BECAUSE OF A WRITE FAIL DURING THE SPIN-UP
   WRITE TESTS.
2) BRING UP THE PROGRAM AND SPECIFY AUTOMATIC MODE
3) ANSWER THE QUESTION "ENABLE REPLACEMENTS" WITH A "N" (NO)

Verify Mode can also be entered by specifying Manual Mode with
replacements disabled. In this way, you can "look" at the "status"
of a particular LBN.

2.2 Automatic Mode
This is the mode used when you do not know what the bad block LBN(s)
are, or do not have the bad block LBN address(es) available in decimal.
Or, possibly you just want the program to run unattended, and take
action on any blocks it finds bad. Normally you should
use Manual Mode and enter the bad blocks, given that you know (from the
error logger or other sources of information) what blocks are bad.
Please make note of the Warning in Section 4.0, when running with large
pass counts. Large pass counts in Automatic Mode are recommended,
however please understand and follow that warning. Execution in this
mode is entered by:
1) Insure the Customer has backed-up and verified the data from the
   subject drive
2) Specify Automatic replacements
3) Answer "Enable Replacements" with a Y (yes)
4) After the completion of the program, the customer can restore
   and verify the backed-up data.

2.3 Manual Mode
This is the preferred method for using the program and the most
efficient. If you can obtain a decimal address of the bad block(s) you
want replaced, this mode will take the information (LBN address) and
without hesitation replace it. Operating System error loggers, should
provide the LBN address of those blocks that generate ECC and Header
errors. Sometimes you may have to "convert" the LBN address from
Hexadecimal or Octal, as provided in the error log-report, to decimal
so it can be used by this program. Providing the wrong decimal address
of a bad block is not good. The program will replace any LBN you
specify and if you specify the wrong one, you will have one additional
replaced block on the media.

If you have no idea what the LBN addresses of the bad block(s) are, you
can possibly run this program in Verify Mode and if Verify finds any
bad blocks, it will report them to you in decimal. Taking these
addresses and using them in Manual Mode, will result in the quickest
method for replacing known bad blocks. However, the customers operating
system has much more time to find bad blocks than does Verify Mode.
Thus, you should use the error log for bad block determination whenever
possible. Execution in this mode is entered by:
1) Insure the Customer has backed-up and verified the data from the

subject drive
2) Specify Manual replacements
3) Answer "Enable Replacements" with a Y (yes)
4) Provide the LBN address of the bad block in decimal, when
   asked for. A Carriage Return at this question will exit Manual
   mode.
4) After the completion of the program, the customer can restore
   the backed-up data and verify it.

3.0 BACKGROUND

This program was primarily developed for use in those cases where a customer
has an operating system that does not have the DSA Bad Block Replacement
capability already in it. Several operating systems already have a method of
doing DSA Bad Block Replacement (BBR) and would only rarely need the services
of this program (VMS is one of many). However, their are cases where even with
an operating system that has the BBR capability, the need to replace a specific
LBN, or allow this program to search for bad blocks, may be appropriate.

In those cases where a customer is running an operating system that does
not have the BBR capability, bad blocks may develop and show up as ECC errors
in the system error log. When this occurs, this program can be used to
replace those bad blocks and thus eliminate the logging of ECC errors related
to specific bad block(s). It's a known fact that bad blocks can become worse
with time. What may start out as a 6 symbol ECC error may eventually end up as
an uncorrectable ECC error, if action is not taken to replace that block.

CAUTION

It is imperative that every effort be used to establish the fact
that errors showing up in the error log are indeed related to a
bad block, before attempting to use this program. If ECC errors
showing up in an error log are the result of a hardware problem,
this program (When used in Automatic Mode) could replace a
considerable number of blocks that are indeed good. If this happens,
and later you fix the problem causing the ECC errors, you should
reformat the HDA/Pack. A description of this situation is given in
RA81-TT-12. The information in this Tech Tip applies to all RA series
drives and would be helpful in this situation.

4.0 DISCUSSION

Normally, disk blocks that are defective generate ECC errors when read. These
ECC errors, as seen from the error log, should all be logged against a
particular LBN (or if multiple bad blocks, a specific number of LBN's). Thus,
if you observe that the ECC error(s) seem to all occur at the same LBN
address(es), you can start to assume that a bad block is the cause. This being
the case, you can either provide this program with the LBN address(es) in
decimal, and have them replaced in Manual Mode, or allow this program the
opportunity to find the bad block(s) (Automatic Mode).
IT IS RECOMMENDED that if you feel sure you have identified the bad blocks, and
have the LBN addresses of them in decimal, you should use Manual Mode to have
them replaced. Manual Mode is a quick efficient method of getting a block
replaced and should be used to save time and effort, however, be sure you know
what your doing and have the right information needed by this program.

In Automatic Mode, this program will run the specified number of passes
on the drives you specify. It does not execute simultaneously on the drives
you specify but works in a "serial" fashion, doing one drive at a time.
One pass is one execution of the program on the drive(s) you specify. IT IS
HIGHLY RECOMMENDED that when using this program in Automatic Mode that you
allow multiple passes to be run. Running this program, in Automatic Mode,
over a weekend or overnight would provide the most benefit.

4.1 Sample Run times in Automatic Mode
    RA81 -- About 9 Minutes
    RA60 -- About 4.5 Minutes
    RA80 -- About 3 Minutes

    NOTE: This time is extended, when replacements occur. Thus, if you
          have many bad blocks, the program will take considerably longer.

4.2 In order to verify that an LBN indeed needs replacement, this program
    (And all other Operating Systems with BBR capability) tries to
    "verify" that the bad block report is not a transient error and in
    fact relates to a defect in the media. If all reports of Bad Blocks
    (Like ECC errors) caused replacement, without question, then a
    substantial possibility would exist that blocks with no defect would
    be replaced. As you know, many times errors are caused by the
    environment (Like static, electrical problems etc). Errors reported
    that are not related to a media defect should not be replaced. Thus,
    an attempt is made to verify that a bad block report can be duplicated
    and therefore related to a higher probability of a media defect being
    the cause. This is the way it happens in the most usual case:

    1) This program uses the MSCP ACCESS command to do large Byte
       count transfers (Read) from the drive being tested. It starts
       with LBN 0 progresses to the highest user LBN on the drive.
       The ACCESS command moves data from the drive to the
       controller, where the data is checked for any errors.
       The data IS NOT actually moved host memory. This way we can
       read data at the fastest possible rate and thereby make it
       possible for more passes to be made across the media in the
       shortest possible amount of time.

    2) If an error (Lets say an 8 symbol ECC error) is detected by
       the controller, a flag gets set called the Bad Block Reported
       flag. This flag is reported to this program by whats called
       an "end packet". This packet describes the success or failure
       of the ACCESS command that we gave the controller. The end
       packet also provides the LBN address that caused the error.

    3) When this program sees this Bad Block Reported Flag, it takes
       the LBN address that caused the Bad Block reported flag to set
       and goes into a series of READ commands to the LBN that
       reported the bad block. A series of 32 reads are attempted.
       If a second Bad Block report occurs, the program terminates
       the testing and replaces the block.

    4) If 32 re-READ's are accomplished without error or bad block
       report then the error is considered a transient and replacement
       is not attempted.

5) HOWEVER, if it is considered a transient error, the LBN address
   is logged in a memory table called the transient error table.
   If on a subsequent passes across the media, this same LBN
   address reports an error or Bad Block, the LBN is replaced
   without argument and without any further testing. In other
   words, if we have a transient error and then later this same
   LBN reports another error, it is replaced.

That is why running this program for multiple passes is beneficial.
The transient error table is kept in memory and is kept updated as
long as the program is running. If the program is terminated, the
table is lost and any entries in it. The benefit of this table is
that any marginal bad blocks that just does not show an error each
time it is read can be replaced.

4.2.1 Transient Error Table Overflow

The Transient Error Table, that is kept in memory as long as
this program is running, has the ability to keep track of
400 Transient errors for each drive being tested. This table
should be of sufficient size for all cases where bad blocks
that are only marginally bad need to be tracked. If this table
overflows, a message is typed and the testing terminates. If
this happens, you have more trouble than what this program can
help with. This message is reported in the following manner:

        Transient error table overflow;
          ADDITIONAL MAINTENANCE REQUIRED.

This is true. If you log more than 400 transients, then the
drive being tested may not have bad blocks causing the
errors you see. Most likely you have a data path problem giving
false indications of ECC errors or something like a worn-out
spindle ground brush.

The recovery from this condition would be to first fix whatever
is causing the high transient error rate then reformat the
media. Most likely their are many replacements that were made
on blocks that were in fact not bad. Reformatting will put the
media back to a known state (Please read RA81-TT-12). You may
want to run several passes of this program, after having
reformatted, just to be sure that the formatter found all the
bad spots.

4.3 System Crash while running this program -- WARNING

```
WARNING      WARNING      WARNING      WARNING      WARNING      WARNING
+--------------------------------------------------------------------+
} If the system crashes (CPU failure, power fail etc) DURING         }
} THE EXECUTION OF THIS PROGRAM (Either Manual or Automatic          }
} Mode) IT IS REQUIRED that you run at least one pass of this        }
} program after recovering from the crash condition.  Abrupt         }
} termination of this program could leave an incomplete replacement. }
} One quick pass of this program would allow for the completion of   }
```

```
                    } any incomplete replacements.  Using the control "C" to terminate   }
                    } this program is not considered an abrupt termination and should    }
                    } not leave any incomplete replacements, although use of control      }
                    } "C" to terminate this program is not recommended.                   }
             +---------------------------------------------------------------------+
```

5.0 STARTING THE PROGRAM

This section describes the hardware questions which must be answered for each
Unit configured for testing. Each question is displayed along with the default
value for the parameter. A carriage return (CR) selects the default value. An
explanation of each question follows.

1) Boot the Diagnostic Media
2) After you have called for the running of this program, you will arrive
   at the diagnostic supervisor prompt - DR>. Typing "start/pass:(the number
   of passes you want run)" at the prompt will begin the execution of this
   program and ask you the following questions:

      CHANGE HW (L) ?   Yes   -Answer "yes" to enable the selection of which
                               drives you want to use and other operating
                               parameters.

      # UNITS (D) 1   - This asks you HOW MANY DRIVES YOU WANT TO TEST.
                               In Automatic Mode this diagnostic will execute
                               on the number of drives you specify in a "serial"
                               fashion (One drive at a time).

      UNIT 0   - Unit 0 is NOT drive 0. This only refers to the
                               first drive you want used and asks for certain
                               information about this first drive to test.

      BUS ADDRESS OF XDA (O) 172150 ? - UNIBUS ADDRESS OF THE CONTROLLER that
                                        the first drive is connected to
                                        is the first UDA50 default
                                        in Octal) "XDA" refers to
                                        the controller -- Like UDA or KDA.

      VECTOR (O) 154   - VECTOR YOU WANT PROGRAMMED INTO THIS
                               CONTROLLER. (RA drive controllers have
                               software provided vectors (Take the default
                               or provide the vector you want assigned).

      BR LEVEL (D) 5   - TAKE THE DEFAULT, unless you have a good
                               reason not to.

      BUS BURST RATE (D) 63 ? - This defines the NUMBER OF LONG WORDS THAT
                                WILL BE TRANSFERRED each time the controller
                                becomes bus master. The default (63) should
                                be good. Operating System Burst Rates must
                                always equal 0 (zero).

      DRIVE # (D) 0   - This is the DRIVE NUMBER TO TEST, in decimal.
                               Default is drive "0".
```

The above questions will be asked for each of the number of Units you specified
to test.  Once the above information has been defined, for each drive you wish
to test, the following questions will be asked.  The following questions are
asked only once and apply to this run of the test.


## 6.0 QUESTIONS ASKED BY THE PROGRAM

### 6.1 DISPLAY OPERATOR HELP (L)   N ?
IT IS HIGHLY RECOMMENDED THAT YOU READ THE OPERATOR HELP INFORMATION.
Answering yes will cause this help file to be typed.  For the Operator
Help file to be printed, it must be resident on the diagnostic media
that this program was obtained from.  This information is not a part
of the program but is a separate file that resides on the diagnostic
media and will be typed when this question is answered with a "Y" (Yes)

### 6.2 AUTOMATIC OR MANUAL REPLACEMENT (A) A ?
The default response of "A" selects Automatic replacement mode; an "M"
selects Manual replacement mode.

If AUTOMATIC REPLACEMENT MODE is selected, the entire user LBN area of
each Unit will be processed, and this program will replace all blocks
found to be bad.  In this mode, up to eight Units can be processed
in a "serial" fashion (One drive at a time).

If MANUAL REPLACEMENT MODE is selected, the operator must specify the
desired block(s) to be replaced.  In this mode, the LBN address sup-
plied to the program must be in decimal (Not Octal, Hexadecimal etc).
Only a single Unit can be processed during each program execution, in
Manual Mode.


### 6.3 Questions Asked In Automatic Replacement Mode

Invoking Automatic Replacement Mode, the following Automatic replacement
mode questions are asked.


#### 6.3.1 AUTOMATIC CRASH RECOVERY (L) Y ?

This parameter determines whether crash recovery will be performed
automatically, upon the detection of an incomplete replacement.
Incomplete replacements can occur due to a "system crash" on an
earlier run of this test or operating system replacement process.
If automatic recovery is selected, the program will complete the
replacement operation before processing the Unit.  If automatic
recovery is disabled, the Unit will be dropped upon the detection
of an incomplete replacement.  WE RECOMMEND TAKING THE DEFAULT YES
ANSWER.  In Manual mode this question is asked only upon detection
of a need for crash recovery.  When an incomplete replacement is
detected, a message similar to the following will occur:

RCT ERROR DETECTED;
  Crash occurred during previous replacement Phase 1.

  Attempting Recovery...

LBN:     679864.  Status:          RBN:     13331.   Operation: SEC

This example indicates that a crash occurred, sometime in the past,
during a phase 1 replacement.  Replacement at phase 1 will assign
a replacement RBN, and attempt recovery.  If this program shows a
crash recovery, it would be wise to allow at least one pass of of
this program, in Automatic Mode, to take place.  No "Status" is listed.
This would be normal for recovery, as the recovery process does not
know the original reason replacement was requested.


## 6.3.2 DISPLAY REPLACEMENTS AS THEY OCCUR (L) Y ?

This parameter determines whether replacement operations are
displayed as they occur.  A response of "Y" will cause replacement
operations to be displayed; a "N" will inhibit the display of
replacement operations.  In Manual replacement mode, display is
permanently enabled, allowing the operator to verify desired LBN's
have been replaced.  If answered with a "Yes", as a replacement is
made the following output will be provided on the console terminal.


## 6.3.2.1 Replacement Status Messages

Each time the program is started, it will type out the following
information.  This information is used during the report of a
replacement operation (Shown Below).  Understanding these "codes"
will allow you to better understand the replacement operation that
was performed and why replacement was requested.


                    LBN Status codes:
                    -----------------
        BBR - Bad Block reported (ECC Error)
        ECH - Hard ECC Error encountered
        HCE - Header Compare Error encountered
        DST - Data Sync Timeout encountered
        FER - Forced Error encountered.
        WFE - Block written with FE (Forced Error) set


                    RBN Operation types:
                    --------------------
        PRI - Primary replacement for LBN.
        SEC - Secondary replacement for LBN.
        UNS - RBN marked unusable.


                            NOTE
        +----------------------------------------------------+
        ) Periods after a number indicate that the number )
        ) is a decimal number.                            )
        +----------------------------------------------------+


                    PRIMARY BAD BLOCK REPLACEMENT
                    -----------------------------
The LBN was tested, and the BBR flag was set repeatedly.  The LBN's
Primary replacement RBN was used to store the LBN's data.

    LBN: 23271. Status: BBR    RBN:    456. Operation: PRI

### SECONDARY BAD BLOCK REPLACEMENT
------------------------------------

The LBN was tested, and the BBR flag was set repeatedly.  The LBN's
Primary replacement RBN is currently used by another LBN.  A Secondary
replacement RBN was used.

    LBN: 23271. Status: BBR    RBN:    455. Operation: SEC

### RBN BECOMES UNUSABLE
-----------------------

Upon reading data from an RBN, an error resulted.  This indicated that
the RBN is bad and must be marked as unusable.  Also, the customer data
residing in the RBN must be moved to a new RBN.  A Secondary replacement
always results from this condition.

    LBN: 23271. Status:       RBN:    456. Operation: UNS
    LBN: 23271. Status: BBR    RBN:    455. Operation: SEC

### LBN WITH AN UNCORRECTABLE ECC ERROR REPLACED
-----------------------------------------------

An LBN is read repeatedly with an uncorrectable ECC error (ECH).  The
indicated replacement operation is performed.  If enabled, the LBN data
is written with the Forced Error (FE) flag set, and the following
information appears.

    LBN: 23271. Status: WFE    RBN:    456. Operation: SEC (or PRI)

If the question about writing uncorrectable ECC errors WITHOUT the
Forced Error Flag is answered with a NO, the corrupted data is written
to the RBN as is, and the following information appears:

    LBN: 23271. Status: ECH    RBN:    456. Operation: SEC (or PRI)

### TRANSIENT ECC ERROR REPORTED
-------------------------------

During the "scan" for bad blocks, the BBR flag was set for a particular
LBN.  Once this condition was noted, the program tries to verify the
report of a bad block.  If the bad block report can not be verified,
the operator is notified of the LBN and NO replacement is attempted.
If you do not feel good about an ECC error being reported and then the
program not being able to verify the error, you can use Manual Mode to
replace the reported LBN.

    LBN: 23271. Status: BBR

                              NOTE
    +----------------------------------------------------------+
    ) Many times the occurrence of an ECC error that is not)
    ) repeatable can indicate a Data Path problem or other )
    ) conditions that are not related to bad spots.        )
    ) If this condition occurs many times during a pass of )

```
                    } this program, I would start to look for a data path  }
                    } problem or some other problem causing transient       }
                    } errors.                                               }
                    +----------------------------------------------------------+
```

6.4 Questions Asked In Manual Replacement Mode

Invoking Manual Replacement Mode. The following Manual Replacement Mode
questions are asked.

   6.4.1 ENTER LBN TO BE REPLACED (DECIMAL) OR <CR> TO EXIT (A) ?

      This is asking for you to specify which LBN you wish to have
      replaced. BE SURE YOU ARE PREPARED TO PROVIDE THE LBN ADDRESS IN
      DECIMAL. Different Operating Systems error loggers display the LBN
      address in various radix's (Like Hex, Octal, Decimal etc) Be sure
      you know what your dealing with and make the proper conversions, if
      necessary.

      The LBN entered is checked against the maximum number of user LBN's
      available (See section 7.12). If the limit check fails, the
      following message appears:

         Limits: LO= 0; HI= dddddd.

      HI would be the maximum LBN address that can be replaced on that
      drive. If the LBN passes the limit check, the replacement is made
      and the operation is displayed, as specified in the paragraphs
      below.

      Once you have provided the LBN address for the replacement, and the
      replacement has occurred, this same question will be asked. If you
      have no more LBN's for replacement, just hit Carriage Return to exit.

   6.4.2 Attempt Crash Recovery (L) Y ?

      This question is only asked upon the detection of a previously
      incomplete replacement. Incomplete replacements can occur due to a
      "system crash" on an earlier run of this test or operating system
      replacement process. If you answer yes, the program will complete
      the replacement operation before processing the requested Manual Mode
      LBN replacement. If you answer no, the Unit will be dropped and the
      incomplete replacement will not be completed. We recommend a
      yes answer. The following is an example of this condition.

         SCRUB1 SFT ERR 00061 ON UNIT 00 TST 001 SUB 000 PC:02506
         RCT error detected

         Crash occurred during previous replacement
         Phase 1

         Attempt crash recovery (L) Y ?

Attempting Recovery...

LBN: 679864. Status: RBN: 13331.   Operation: SEC

This example indicates that a crash occurred, sometime in the past,
during a phase 1 replacement. Replacement at phase 1 will assign
a replacement RBN, and attempt recovery. If this program shows a
crash recovery, it would be wise to allow at least one pass of of
this program, in Automatic Mode, to take place. No "Status" is
listed. This would be normal for recovery, as the recovery process
does not know the original reason replacement was requested.

6.5 Questions Asked During Either Replacement Mode

6.5.1 DISPLAY RCT REPLACEMENT DESCRIPTORS (L) N ?

This parameter affects program execution in both replacement modes.
The Replacement Control Table (RCT) keeps a "log" of all LBN's that
get replaced and the Replacement Block Number (RBN) that is used.
In Automatic replacement mode, display selection will cause the
updated Replacement Control Table (RCT) descriptors to be printed at
the end of each pass, for each Unit processed. In Manual replacement
mode, this question is asked twice: When the Unit is brought Online,
the operator may display the existing RCT replacement descriptors.
After all desired LBN's have been replaced, the operator may display
the updated RCT replacement descriptors.

Remember that the period used after a number indicates that the
number is displayed in decimal.

RBN: 455. is SECONDARY   replacement for LBN: 23271.
RBN: 456. is PRIMARY   replacement for LBN: 23272.
RBN: 457. is UNUSABLE

After having described the "status" of all replacements logged in
the Replacement Control Table, The following summary is provided:

RCT DESCRIPTOR CONTENTS SUMMARY
   Primary   replacement blocks: ddd.
   Secondary replacement blocks: ddd.
   Unusable replacement blocks: ddd.
   Total RBNs on media: dddddd.

Displaying this information will provide an indication of the
TOTAL number of replacements that are "logged" in the RCT. These
replacements may have been made by the Operating System (If the O/S
has the capability of doing Bad Block Replacement), the formatter
or possibly this program. Although their is no "specification" for
how many replacements are too many, you should be aware of certain
conditions that would indicate a problem. An RA81 can accommodate
17,472 total replacements. A good working HDA can have a thousand
primary replacements, however, the number of secondary replacements

should be small (Like on a ratio of about 100 primary's to 1
secondary or so). If you displayed something more than a thousand
primary's or more than a few hundred secondaries, it could indicate
that you may have a data path problem or had one in the past that
was repaired. ECC errors being generated by a Data Path problem can
cause a significant number of good blocks to be replaced before the
data path problem can be repaired. If you see what may appear to be
this condition, you should read RA81-TT-12. The Tech Tip applies to
all RA drives, even though it is written as an RA81 Tech Tip. If
this is the condition you sense, the tech tip will recommend that
you reformat the HDA. Now that this program exists, it would be wise
to not only reformat, to recover from this condition, but to run as
many passes of this program as possible, after having reformatted.
Any bad blocks that the formatter may not have found, may be found
by this program and replaced.

6.5.2 ENABLE REPLACEMENTS (L) Y ?

This is a unique feature of this program. Disabling replacements
(Answering with a NO) makes this program run in a similar fashion
to the HSC50 "Verify" program. In other words, this program will
"Scan" the media and report anything it finds but not do any
replacements. DO NOT (REPEAT DO NOT) EXPECT THE SAME RESULTS FROM
THIS PROGRAM WHEN COMPARED TO THE HSC50 "VERIFY" PROGRAM. THE HSC50
CAN LOWER THE "DRIVE THRESHOLD" WHERE THIS PROGRAM CAN NOT. Using it
in this mode could possibly help you determine the "condition" of
the media, without having any replacements occur. AGAIN -- THIS
PROGRAM IS NOT (REPEAT NOT) INTENDED FOR USE AS A DIAGNOSTIC AND
SHOULD NOT BE RUN ON DRIVES/CONTROLLERS THAT MAY HAVE PROBLEMS
(Other than normal bad blocks and the like).

Using this program in this mode (Answering with a "NO") will disable
all writes and therefore you can run it without the normal
requirement for a back-up and restore. However, we can not be held
responsible for any errors or subsystem problems that may create
situations that cause this program (or subsystem) to do unexpected
things.

WE HIGHLY RECOMMEND THAT YOU RUN IN THIS MODE (ANSWER = NO), WITH
THE DRIVE(S) WRITE PROTECTED !!

What you can do is run with replacements disabled just to see if its
worth your time to do the extensive Backup and Restore operations
that are required when replacements are enabled. Remember, you can
not run this program with replacements enabled and the drives write
protected. One pass may not find all the "marginal" bad spots. Thus
we highly recommend that when running in the "disable replacement"
mode (Answer=NO) that you allow the program to run several passes.

This mode can also be used to get a quick "look" at the number and
types of replaced blocks that are "logged" in the Replacement Control
table (RCT). By going into this mode, and answering the question
"Display Replacement Descriptors" with a yes, you can get these
typed without the Backup and Restore. When analyzing the
descriptors that are typed, be sure you read and understand the
discussion for the question "Display Replacement Descriptors"

(above).

6.5.3 ENABLE WRITE WITH FORCED ERROR FLAG (L) Y ?

This is not going to be easy to explain in 10 words or less. First
you have to understand what a "Forced Error Flag" is. Many people
have the wrong impression of what a "Forced Error" or "Forced Error
Flag" is and what it does. Let me try to briefly explain. YOU NEED
TO KNOW THIS.

When an uncorrectable ECC error is encountered, several attempts are
made to read the data correctly (Using every means available). If
these attempts fail, the block generating the Uncorrectable ECC
error is assumed to be bad and in need of replacement. Thus, if you
have a system that does "Dynamic Bad Block Replacement" (Like VMS,
RSTS, RSX, IAS), or your running this program, the replacement
process will take the uncorrectable (Corrupt) data and move it to a
good replacement block (RBN). Now, we have a condition where we have
corrupt data in a good block and if left this way the corrupt data
would be read with no "indication" that the data is "corrupt".
Therefore, in order to tell a user that the data was at one time
uncorrectable and thus exists now as "corrupted" data (In a good
RBN), the "Forced Error Flag" is attached to the block of
"corrupted" data. When this block is read by a user, this flag
(Inverted EDC character) is also read and is intended to inform the
user that the requested data is "not reliable". THE FORCED ERROR
FLAG IS NOT (REPEAT NOT) AN ERROR!! WHEN READ, IT WILL NOT
MAKE AN ENTRY IN THE SYSTEM ERROR LOG. IT IS REPORTED AS A "STATUS"
CODE 10 (OCTAL) IN A TRANSFER REQUEST "END PACKET" (This program
calls the end packet status field the "Endmsg status", in several
message types that can be displayed)

So why do we have this question. It seems that one or more Operating
Systems have trouble dealing with, or reporting, the "Forced Error
Flag". Also, some others do not have an intelligent way of reporting
the flag to the user (Makes it appear as a "hardware error".
Currently (March 1985) UNIX type systems (ULTRIX, Berkeley UNIX,
AT&T UNIX etc) can "give up" (I assume that means crash) in certain
situations when a "Forced Error Flag" is read. This being what we
are told, we added this question to allow the user to disable the
function that writes the forced error flag to an RBN with the
uncorrectable data.

NOW LISTEN CAREFULLY.

If you answer the question "Enable write with Forced Error Flag"
with a NO, you will put uncorrectable data into a good RBN (Just
normal replacement). The "corrupted" data will read good, with NO
indication of an error or the "corrupted data flag" (Forced
error Flag). Therefore, the possibility of reading corrupted data
with NO indication that it is corrupted. IF YOU FOLLOW THE
RECOMMENDED PROCEDURE FOR RUNNING THIS PROGRAM AND DO THE BACKUP,
AND THEN RESTORE THE BACKED-UP DATA AFTER RUNNING THIS PROGRAM, YOU
WILL NOT HAVE A PROBLEM. Restoring the backed-up data, to a drive
that had this question answered with a NO, should eliminate any
problem condition for these systems. For those systems that do not

know how to handle the "Forced Error Flag", following the procedure
of answering the question with a no and then doing a backed-up data
restore is a REQUIREMENT WITHOUT EXCEPTION (repeat WITHOUT
EXCEPTION)!!

7.0 "HARD" ERROR REPORTING

This program can display information from Error Log Packets and other sources
of error information, that the controller/drive may send. This may not be what
you think, due to the fact that THIS PROGRAM IS NOT (Repeat NOT) INTENDED TO BE
USED ON RA DRIVES, OR CONTROLLERS, THAT ARE DEFECTIVE. You must be sure that
you are trying to eliminate bad blocks, before using this program. This program
will not give you all the details of an error that occurred (Like a Fault) but
will give you the most important information AFTER HAVING GIVEN YOU THE ERROR
INFORMATION IT WILL DROP THE DRIVE AND TERMINATE. This is for your protection.
Doing replacements on a defective drive (or controller) can cause BIG TROUBLE.
Since we can not control the use of the program, we must always assume the
worst and try to prevent one from creating bigger problems from the misuse of
this program.

If you see the following kinds of errors reported, you should discontinue the
use of this program and use the other diagnostics, or system error logger, to
better isolate any problems. If this program reports Hard Errors, like the
following examples, their is always a possibility of a drive/controller
generating a hard error that will cause an incomplete replacement. If you
attempted to use this program, and noticed an error like the following
examples, you should take action to better analyze the problem, using the other
diagnostics. Once you feel that the subsystem is working properly, you should
run at least one pass of this program, in automatic mode, to take care of any
incomplete replacements and any other bad blocks.

The following examples list the possible errors reported by error log packets
that will cause the program to terminate.

                              NOTE
        +--+
        } The little "o" used for the Status/event, Host address and   }
        } Drive Error Code means the information is typed in Octal. The }
        } little "d" means decimal and the little "x" means Hexadecimal. }
        +--+

7.1 Errors Reported by Error Log Packets from the Controller

Their are several sources of error information in an RA subsystem.
One of the most common, is error log packets that the controller
assembles and sends to the host. Their are 5 types of error log
packets. Three of these types can be reported by this program.

7.1.1 Controller Error reported

This error occurs when an Error log packet is received
specifying that a Controller detected an error within itself. The

Unit being tested will be dropped. This type of error will be
reported in the following manner:

              Controller Error Reported:
                 Status/event: oooooo

The Status/Event code will give you an indication of what the
trouble may be. A list of the Status/Event codes and their meanings
are given in section 7.2

7.1.2 Host Memory Access Error Reported

This error occurs when an Error log packet is received
specifying that a Host Memory Access Error was detected. This type
of error packet reports problems that the controller experienced
"dealing" with host memory (Like while doing data transfers etc).
This type of error will be reported in the following manner;

              Host Memory Access Error reported;
                 Status/event: oooooo
                 Host address: oooooo

The host bus (Like UNIBUS) address is displayed and was the one
being used at the time of the error. The Status/Event code will give
you and indication of what the problem was. A list of the
Status/Event codes and their meanings are given in section 7.2

7.1.3 SDI Error Reported

This error occurs when an Error log packet is received
specifying that an SDI Error or Drive Detected Error was detected.
If a Status/Event code of 353 (S/E Codes are in Octal) is reported,
a valid Drive error code may be reported. The Drive Error Code is
displayed along with the Drive dependent information in the packet,
IN HEXADECIMAL. This type of error will be reported in the following
manner.

           SDI Error reported; LBN: dddddd.
              Status/event: oooooo
              Drive error code: xxxxxx
              Drive dependent information (hex): xx xx xx xx xx xx xx xx

The LBN is the Logical Block Number that was being accessed at the
time the error was detected. The LBN is always reported by this
program in decimal. The "Drive Dependent Information", and Drive
Error Code, are typed in HEX because that's the way this information
is provided in most of the drive service manuals. YOU CAN LOOK-UP
THE DRIVE ERROR CODE IN THE DRIVE SERVICE MANUAL FOR THE MEANING OF
THE REPORTED PROBLEM. Most of the drive service manuals (Like RA80
and RA81) provide some information on the meanings of the bytes
displayed for the Drive Dependent information. This information is
sometimes known as the "drive specific status bytes" of the
"extended status". Re-stating this information as the byte
number of the drive extended status, it would appear as:

Drive dependent information :   9 8 11 10 13 12 15 14

Where the byte number is the byte number of the drive specific
status. As mentioned, many of the service manuals provide
information on the meaning of these bytes.

7.2 STATUS/EVENT CODES

The following list shows the various Status/Event codes used in many of the
messages displayed by this program. Certain deviations may exist for specific
controllers. This is the "generic" list. Remember, this program reports
Status/Event codes in Octal.

```
                                    DESCRIPTION
    Octal   Hex    ------------------------------------------------------
        0     0    Successful completion
        1     1    Invalid command (high byte = byte offset of bad command field)
        2     2    Command aborted
        3     3    Drive off line (unknown unit or Online to another controller)
        4     4    Drive available
        5     5    Media Format Error
        6     6    Unit Write Protected
        7     7    Compare error (on compare command or compare modifier)
       10     8    Data Error (Or may have been written with "Forced Error Flag")
       11     9    Host Buffer Access Error
       12     A    Controller Error (Command Time Out/Retry Exceeded)
       13     B    Drive Error
       40    20    Spin down ignored (multi-unit drives only)
       43    23    No volume mounted or drive run/stop switch out
       45    25    Format Control Table unreadable - EDC Error
       51    29    Odd transfer start address in MSCP packet
       52    2A    SERDES  overrun error (controller probably broken)
  *    53    2B    SDI level two response timeout (Or maybe seek incomplete)
                   ('53 also shows up with drive errors as a result NOT CAUSE)
      100    40    Still Connected (multi-unit drives only)
      103    43    Drive inoperative (UDA cannot communicate with drive)
      105    45    Format Control Table unreadable - Invalid Sector Header
      110    48    Header compare error (header not found and no revector)
      111    49    Odd byte count in MSCP packet
      112    4A    EDC error (SERDES broken or EDC written bad -- Controller)
  *   113    4B    Invalid SDI level two response (unsuccessful or trash)
      145    65    Format Control Table unreadable - Data Sync Timeout
      150    68    Data Sync timeout (Data Sync field in sector)
      151    69    UNIBUS nonexistent memory error
      152    6A    Inconsistent controller state
      153    6B    Positioner Error (headers consistent but not on cyl)
      200    80    Duplicate Unit Numbers
      203    83    Drive offline and duplicate unit numbers
      211    89    UNIBUS parity error (UNIBUS read)(Host Memory Parity?)
  *   213    8B    Lost read/write ready during or between transfers
                   ('213 also shows up with drive errors as a result NOT CAUSE)
      245    A5    Drive not 512 Byte format (16 bit format)
      253    AB    Lost Drive clock during data/SDI transfer
      305    C5    Drive not formatted or Format Control Table Corrupted
```

```
 * 313   CB    Lost drive receiver ready during transfer (see 213 comment)
   345   E5    ECC error on FCT read (media format - FCT unreadable)
   350   E8    Uncorrectable ECC error (This uncorrectable data may be
               re-written to an RBN with the Forced Error Flag)
   353   EB    Drive detected error (DRIVE HAD ERROR! -- Find drive error code)
   400   100   Already Online  (in response to ONLINE)
   403   103   Drive offline.  By field service or internal Diagnostics
   410   108   One symbol ECC error
   413   10B   RTDS pulse/parity error (IS UDA M7486 UP TO REV ?)
   450   128   Two symbol ECC error   +----------------------------------------+
   510   148   Three symbol ECC error )* NOTE: Codes 53,113,213 and 313 occur)
   550   168   Four symbol ECC error  )   most often as the RESULT of a "Drive)
   610   188   Five symbol ECC error  )   Error" (S/E Code 353 Octal).  You   )
   650   1A8   Six symbol ECC error   )   should look for a problem in the    )
   710   1C8   Seven symbol ECC error )   drive before believing that these   )
   750   1E8   Eight symbol ECC error )   codes really represent the          )
 10006  1006   Drive software write protected  description of your problem.   )
 20006  2006   Drive hardware write protected  ------------------------------+
```

KEY: EDC = Error Detection Code (Written in each sector)
     SDI = Standard Disk Interface (Drive Bus)
     SERDES = SERializer/DESerializer (In Controller)
     FCT = Format Control Table (Written on media surface)
     ECC = Error Correction Code (Written in each sector)

7.3 CONTROLLER INITIALIZATION ERRORS

The following errors can be reported during controller initialization. These
are basically "HARD" serious errors and usually indicate a very sick
controller. This program, as with any operating system, must first initialize
the controller, to prepare it for use. For example, in the UDA50 and KDA50 type
controllers, this is a 4 step process. Errors that are detected during this
initialization are reported via the following types of messages.

7.3.1 Controller Address Error Occurred While Accessing Controller Register

This error occurs when the program attempts to read a controller
register and a memory access error occurred. This is most likely
caused by the operator specifying an incorrect controller address.
The specified Unit will be dropped from testing. This type of error
will be reported in the following manner:

CONTROLLER ADDRESS ERROR OCCURRED WHILE ACCESSING CONTROLLER REGISTER
  SA address: oooooo
  Check controller address.

The SA address displays the UNIBUS address of the controller register
that was being accessed at the time of the error. This is most
likely caused by the operator specifying an incorrect controller
address. Unit is dropped from testing.

7.3.2 Controller Resident Diagnostics Detected Failure

This error indicates that the controller resident diagnostics
reported an error during initialization. This type of error

will be reported in the following manner:

```
CONTROLLER RESIDENT DIAGNOSTICS DETECTED FAILURE;
   SA address:  000000
   SA contents: 000000
```

The SA register contains the detected error code. The specified Unit
will be dropped from testing. Decoding information for the SA
register contents (error code) is in section 7.4

7.3.3 Step Bit Error In SA Register During Initialization;

This error indicates that during the 4 step (phase) initialization
of the controller a error was detected. This type of error will be
reported in the following manner:

```
STEP BIT ERROR IN SA REGISTER DURING INITIALIZATION;
   SA address:  000000
   SA contents: 000000
   SA expected: 000000
```

The SA address is the UNIBUS address of the controller register that
was being accessed, during the 4 step initialization of the
controller, when the error was detected. The SA contents is what
the controller register contained the displayed the error condition.
SA expected is what this program expected, for correct results
during that step in the initialization. Decoding information for
the SA register contents (error code) is in section 7.4. The
specified Unit will be dropped from testing.

7.3.4 SA Register Did Not Zero After Step 3 Write;

This error occurs when the SA register does not zero during Step 3
of controller initialization. This type of error will be reported
in the following manner:

```
SA REGISTER DID NOT ZERO AFTER STEP 3 WRITE;
   SA address:  000000
   SA contents: 000000
   SA expected: 000000
```

The meanings of the SA address, SA contents and SA expected are the
same as those for the previous error (Section 7.3.3) Decoding
information for the SA register contents (error code) is in section
7.4. The specified Unit will be dropped from testing.

7.3.5 SA Register Error During Initialization.

This is the more common of the Controller Initialization Errors that
can be reported. The SA register Error bit (Bit 15 should be set)
was set during controller initialization. This type of error will
be reported in the following manner:

```
SA REGISTER ERROR DURING INITIALIZATION.
```

```
                        SA address:  oooooo
                        SA contents: oooooo
                        SA expected: oooooo
```

The meanings of the SA address, SA contents and SA expected are the
same as those for the previous error (Section 7.3.3) Decoding
information for the SA register contents (error code) is in section
7.4. The specified Unit will be dropped from testing.

7.3.6 Controller Did Not Clear Ring Structure In Host Memory;

An error occurred during initialization of the Host communication
area. At a certain point during host/controller initialization, the
controller is responsible for clearing the host communication area
(Rings) and this error results from a failure of the controller
to perform this function (As detected by the host).

```
        CONTROLLER DID NOT CLEAR RING STRUCTURE IN HOST MEMORY;
          Host address: oooooo
```

The Host address is a host memory location in the host communication
area that the controller was to have cleared. The problem could be
a problem in the controller or possibly the controller had problems
accessing the hosts memory locations of the communications area.

7.4 CONTROLLER INITIALIZATION (BY SA REGISTER) ERROR CODES

```
     15        11 10                    0
     +-+-+-+-+-+---------------------+              SA REGISTER
     )E)S)S)S)S)    INITIALIZATION    ) Formatted as would be represented when
     )R)4)3)2)1)      ERROR CODE      ) an error occurs during initialization.
     +-+-+-+-+-+---------------------+
     †}-------}
      }      \____ STEP BITS.  Initialization is a four step process.
      }            These bits describe which initialization step the error
      }            occurred in.
      }
      }_____ ERROR BIT.  Qualifies the other bits of the register as
                   containing valid information relating to an error that
                   occurred during initialization.
```

```
                   UDA DURING INITIALIZATION
                    SA REGISTER ERROR CODES
                   ----------------------------
```

Octal	Error Description	Possible FRU
104000	Fatal Sequence Error	7485
104040	D processor ALU	7485
104041	D processor control rom parity error	7485
105102	D proc with no board #2 or RAM Parity error	7486
105105	D proc RAM Buffer Error	7486
105152	D proc SDI Error	7486
105153	D proc Write Mode Wrap SERDES Error	7486

105154	D proc Read Mode SERDES,RSGEN & ECC Error	7486
106040	U proc ALU Error	7485
106041	U proc Control Register Error	7485
106042	U proc DFAIL/Control Rom Parity/BD #1 tst cnt	7485
106047	U proc Constant PROM error with D proc	
	running SDI test	7485
106055	Unexpected trap found,Abort Diagnostic	7485
106071	U proc constant PROM Error	7485
106072	U proc Control ROM Parity Error	7485
106200	Step 1 Data Error (MSB not set)	7485
107103	U proc RAM Parity Error	7486
107107	U proc RAM Buffer Error	7486
107115	Test Count was wrong (BD#2)	7486
112300	Step 2 Error	7485
122240	NPR Error	7485
122300	Step 3 Error	7485
142300	Step 4 Error	7485

7.5 HOST/CONTROLLER COMMUNICATION ERRORS

The following errors can be reported during normal program execution.
These are basically "HARD" serious errors and usually indicate a very
serious problem. Errors that are reported via the "Status/Address" (SA)
register are reported via the following types of messages.

7.5.1 NO INTERRUPT RECEIVED FOR 30 SECONDS

This error occurs when an interrupt timeout occurs during wait for
an MSCP end message. An end massage indicated to the host that the
controller has completed the command that it was given.

NO INTERRUPT RECEIVED FOR 30 SECONDS;
   SA address: oooooo
   SA contents: oooooo

The Host got tired of waiting for a response from the RA drive
controller. If the end packet was not received by the host, the
cause does not always necessarily mean that the controller was at
fault. The SA address is the UNIBUS address of the controller
Status/Address register that was being used at the time
the problem was detected by the host. If the controller detected
a problem, that accounted for the end packet not being delivered
to the host, The Status/Address (SA) register contents should show
an error code and the SA register error bit should be set (Bit 15).
The specified Unit will be dropped from testing.

7.5.2 Fatal Error Reported In SA Register

This error can occur when the Error Bit is set in the SA register,
during normal on-line use of the controller. The error bit (Bit 15)
setting causes the controller to discontinue fetching commands from
the host. The host will read the SA register, detecting that an
error condition exists and display contents of the SA register in

this message.

```
FATAL ERROR REPORTED IN SA REGISTER:
    SA address:  oooooo
    SA contents: oooooo
```

The SA register contents should have the error bit set (bit 15) and
an error code. Section 7.6 has a list of the On-line error codes
and a brief description of their meanings. Do not get these on-line
error codes mixed-up with the initialization error codes that are
described in section 7.4. The specified Unit will be dropped from
testing.

7.6 CONTROLLER COMMUNICATION ERROR CODES

This section provides a list of the various controller error codes that can
reside in the SA register, when the controller detects a hard error during
on-line use of the controller. Do not get these confused with the SA register
initialization error codes described in section 7.4. They are different

```
15          10                0
+-+-+-+-+-+-+----------------+               SA REGISTER
)E)      )     ONLINE     )    Formatted as would be represented when
)R)      )   Error Code   )    the UDA is "ONLINE" to the Host
+-+-+-+-+-+-+----------------+
  †
  )___ Error Bit (Error Code will be displayed in bits 0 - 10) IF bit 15
              is NOT set, the contents of bits 0 - 10 are undefined.
```

UDA ONLINE
SA REGISTER ERROR CODES
----

octal	Error Description	FRU
100001	UNIBUS Packet Read Error	7485*
100002	UNIBUS Packet Write Error	7485*
100003	UDA ROM & RAM Parity Error	7485/7486
100004	UDA RAM Parity Error	7486
100005	UDA ROM Parity error	7485
100006	UNIBUS Ring Read Error	7485*
100007	UNIBUS Ring Write Error	7485*
100010	UNIBUS Interrupt Master Failure	7485
100011	Host Access Timeout Error	7485*
100012	Host Exceeded Command Limit	7485*
100013	UNIBUS Bus Master Failure	7486
100014	DM XFC Fatal Error	7486
100015	Hardware Timeout of Instruction Loop	7485*
100016	Invalid Virtual Ckt Identifier	7485*
100017	Interrupt Write Error on UNIBUS	7485*

* - denotes possible host CPU error.......

7.7 UNIT INITIALIZATION ERRORS

The following errors are reported when an error occurs during the
initialization of a unit (Drive). These are basically "HARD" serious errors
and usually indicate a problem either in a drive or communicating with a
drive. Errors are reported via the "End Message" Status field (Endmsg status:)
If the problem is severe, the specified unit (drive) will not continue being
tested.

                              NOTE
        +--+
        } The meanings of the Endmsg status can be found by looking }
        } it up in the list of Status/Event codes provided in section}
        } 7.2   }
        +--+

7.7.1 Failed Get Unit Status;

"Get Unit Status" is an MSCP command that is used by the host to get
certain information about a drive connected to an RA drive
controller. This error occurs when an attempt to "GET UNIT STATUS"
fails during Unit initialization. This type of error will be
reported in the following manner:

        FAILED GET UNIT STATUS;
            Drive ddd is not accessible.
            Endmsg status: oooooo

Drive ddd is the decimal drive logical unit address that the "Get
Unit Status" was issued for. Possibly the drive port switches are
not pushed in or the incorrect drive address was given to the
program. Errors are reported via the "End Message" Status field
(Endmsg status:) and the various Endmsg status codes can be found
in the table of Status/Event codes in section 7.2

7.7 2 FAILED SET CONTROLLER CHARACTERISTICS;

"SET CONTROLLER CHARACTERISTICS" is an MSCP command that is used by
the host to set certain controller characteristics (Like timeouts and
to enable various kinds of messages). This error occurs when command
completion status indicates a problem executing this command.   This
type of error will be reported in the following manner:

        FAILED SET CONTROLLER CHARACTERISTICS;
            Host access timeout not disabled.
            Default timeout is 60 seconds.
            Error logging is not enabled.
            Endmsg status: oooooo

The default host access timeout is retained, and error logging will
not be enabled. Errors are reported via the "End Message" Status
field (Endmsg status:) and the various Endmsg status codes can be
found in the table of Status/Event codes in section 7.2. Processing
of the specified unit (drive) will continue. However, if this

```
Failed READ; LBN:  dddddd.
  Endmsg flags:       ooo
  Endmsg status:  oooooo
```

This error can be the result of many different conditions. The host
issued a "READ" command to the controller being used for testing and
then received an End Packet indicating that an error was detected
while trying to execute that command. The LBN is the one the "READ"
was attempting to access, at the time the error occurred, as
reported by the End Packet. The Endmsg status can be interpreted from
the list of Status/Event codes in section 7.2. The Endmsg flags are
provided in section 7.10. You should look these up in the charts to
get a possible cause for the error. The Unit (drive) being tested
will be dropped from further testing.

7.9.2 Failed WRITE

This error occurs when an MSCP "WRITE" command fails with an end
message status (Endmsg status:) other than Success or Data Error.
The error is reported in the following manner:

```
Failed WRITE;  LBN:  dddddd.
  Endmsg flags:       ooo
  Endmsg status:  oooooo
```

This error can be the result of many different conditions. The host
issued a "WRITE" command to the controller being used for testing and
then received an End Packet indicating that an error was detected
while trying to execute that command. The LBN is the one the "WRITE"
was attempting to write at the time the error occurred, as
reported by the End Packet. The Endmsg status can be interpreted from
the list of Status/Event codes in section 7.2. The Endmsg flags are
provided in section 7.10. You should look these up in the charts to
get a possible cause for the error. The Unit (drive) being tested
will be dropped from further testing.

7.9.3 Failed ACCESS

This error occurs when an MSCP "ACCESS" command fails with an end
message status (Endmsg status:) other than Success or Data Error.
PLEASE READ THIS COMPLETE ERROR DESCRIPTION. The error is reported
in the following manner:

```
Failed ACCESS;  LBN:  dddddd.
  Endmsg flags:       ooo
  Endmsg status:  oooooo
```

This error can be the result of many different conditions. The host
issued an "ACCESS" command to the controller being used for testing
and then received an End Packet indicating that an error was
detected while trying to execute that command. The LBN is the one
the "ACCESS" was attempting to read at the time the error
occurred, as reported by the End Packet. An ACCESS command is the
MSCP command used by this program to read all the LBN's on the
media. The ACCESS command causes a read operation, checks for any

error conditions (Bad Block), but does not transfer any data to the
host memory. This purpose for this command is to verify that data
can be read without error. The Endmsg status can be interpreted from
the list of Status/Event codes in section 7.2. The Endmsg flags are
provided in section 7.10. You should look these up in the charts to
get a possible cause for the error. The Unit (drive) being tested
will be dropped from further testing.

7.9.4 Failed REPLACE

This error occurs when an MSCP "REPLACE" command fails with an end
message status (Endmsg status:) other than Success or Data Error.
PLEASE READ THIS COMPLETE ERROR DESCRIPTION. The error is reported
in the following manner:

```
Failed REPLACE;  LBN:  dddddd.
  Endmsg flags:      ooo
  Endmsg status:  oooooo
```

This error can be the result of many different conditions. The host
issued a "REPLACE" command to the controller being used for testing
and then received an End Packet indicating that an error was
detected while trying to execute that command. The LBN is the one
the "REPLACE" was attempting to use at the time the error
occurred, as reported by the End Packet. A REPLACE command is the
MSCP command used by programs that do bad block replacement to
actually cause the bad LBN to be logically replaced. The execution
of the REPLACE command is very involved and leaves little room for
error. The header of the bad LBN is written with special codes and
the replacement RBN is allocated to the bad LBN. IF THIS ERROR IS
SEEN, WE RECOMMEND THAT THE MEDIA RE REFORMATTED USING THE FIELD
FORMATTER PROGRAM. HOWEVER, THE FAILURE OF THIS COMMAND COULD BE
DUE TO A DEFECT IN THE DRIVE, OTHER THAN MEDIA BAD BLOCKS. BE SURE
THAT ANY OTHER CAUSES FOR A FAILURE LIKE THIS IS INVESTIGATED AND
ANY APPROPRIATE REPAIRS MADE, BEFORE ATTEMPTING TO REFORMAT THE
MEDIA. A FAILURE LIKE THIS ONE IS ONE OF THE GOOD REASONS FOR USING
THE FORMATTER, HOWEVER, BE SURE NOT TO USE "RECONSTRUCT" MODE OF
THE FORMATTER (SEE RA81-TT-19). The Endmsg status can be interpreted
from the list of Status/Event codes in section 7.2. The Endmsg flags
are provided in section 7.10. You should look these up in the charts
to get a possible cause for the error. The Unit (drive) being tested
will be dropped from further testing.

7.9.5 Command/Response reference number mismatch

In Mass Storage Control Protocol (MSCP) commands that are issued to
the controller are given a "unique" number. In this way, the host
can distinguish this command, and any responses, from any other
commands that may be issued. When a response to a command is
received, the host attempts to associate the "unique" reference
number to a command that has not yet received a response. If a
response reference number can not be matched to a command with the
same number (from commands that have not had responses yet) then
this error occurs. This error is reported in the following manner:

```
Command/response reference number mismatch;
  Command ref:    000000
  Endmsg ref:     000000
  Endmsg flags:      000
  Endmsg status:  000000
```

The user should find that the Command Ref and Endmsg ref do not
match. Why this would happen, is hard to say. The source for this
kind of trouble, is not usually the drive being tested. Rather it
is more likely that a controller or "system" problem exists.
The Endmsg status can be interpreted from the list of Status/Event
codes in section 7.2. The Endmsg flags are provided in section 7.10.
You should look these up in the charts to get a possible cause for
the error. For this kind of error, you may find that the Endmsg
flags and Endmsg status will not indicate an error. In this case,
possibly a "system" type error would be more likely. The Unit
(drive) being tested will be dropped from further testing.

7.9.6 Fatal End Message status detected

An "end message" is the means by which the controller tells the host
about how the command was processed and whether any errors occurred
while executing a command. This error is indicating that a fatal
status was reported by an end message. The error is reported in the
following manner:

```
Fatal end message status detected;
  Endmsg flags:      000
  Endmsg status:  000000
```

Since the error is reporting Fatal end message status, you should
find the an "Endmsg status" code typed. The Endmsg status can be
interpreted from the list of Status/Event codes in section 7.2. The
Endmsg flags are provided in section 7.10. You should look these up
in the charts to get a possible cause for the error. The drive being
tested will be dropped.

7.9.7 Failed Replacement Verification

After the "REPLACE" command has been issued and completed with good
results, the customers data must be written to the replacement block
(RBN). At this point in time, the original bad LBN has been written
with a code that will cause all references to the LBN address to be
"revectored" to the replacement block. Thus the host issues a
"WRITE" command, with the compare modifier, and the customer data
will end up in the replacement block. If this process reports a
problem, this error message will result:

```
Failed Replacement verification;  LBN:  dddddd.
  Retry of replacement operation will be attempted.
  Endmsg flags:      000
  Endmsg status:  000000
```

The program assumes that the cause for the error is that the

replacement block is bad. This can happen. Thus, a retry of the
replacement process takes place. This time another replacement block
is used and the original one is marked unusable. The Endmsg status
can be interpreted from the list of Status/Event codes in section
7.2. The Endmsg flags are provided in section 7.10. You should look
these up in the charts to get a possible cause for the error. The
drive being tested will not be dropped and the replacement retry
will be attempted.

7.10 END MESSAGE FLAGS (Endmsg flags)

FLAGS

Bit flags, collectively called end flags, used to report various
conditions detected due to this command but not directly related to
success or failure. The following flags are defined:

Bad Block Reported: (200 OCTAL -- Bit 7)
    Set if one or more bad blocks were detected and the host is
    expected to perform bad block replacement. Indicates that the
    host should replace the bad block identified by the LBN:

Bad Blocks Unreported: (100 OCTAL -- Bit 6)
    Set if one or more bad blocks were detected and not reported in
    the "first bad block" field of an End Message Packet.

    If the "Bad Block Reported" flag is also set, this indicates
    that two or more bad blocks were detected, and the host (This
    program) should perform bad block replacement. In this case
    the "first bad block" (LBN:) field only reports the first bad
    block in the transfer. If this happens, the program should be
    run for multiple passes.

Error Log Generated: (40 OCTAL -- Bit 5)
    Set if one or more error log messages were generated that
    refer to this command -- i.e., that contain this command's
    command reference number. This flag allows the host to save
    any outstanding command context that it wishes to include in
    the error report. The MSCP server must send the error log
    messages either before or shortly after it sends the end
    message containing this flag. Depending upon the type of error
    log report referenced by this flag, this program will determine
    whether to display the error log report or not. In most cases,
    the Disk Transfer Error Log report is used to report blocks that
    need replacement, thus reporting this error log report is
    somewhat meaningless, since the program will make a replacement
    due to having received this error log type.

7.11 RCT READ/WRITE ERRORS

The following sections list the possible errors reported when a READ or WRITE to
the Replacement Control Table occurs. The RCT does not require special MSCP
commands, just normal READ and WRITE commands with the LBN address such that
the commands are going to reference blocks in the RCT. The error is reported
only once for a particular RCT block. Depending upon the severity of the error

program execution may or may not continue. Their must be at least one good copy
of an RCT block for execution to continue. Their are four copies of the RCT
and at least one good one must be found to do replacements. If no good copies
can be found, then reformatting the media must be accomplished. Reformatting
for this kind of error is a good use of the formatter. However, be sure that a
good copy of the Format Control Table (FCT) exists (See RA81-TT-19).

Replacements can not be done on blocks within the RCT (That's why their are
four copies). Thus, you may see RCT errors frequently and why they are not
always fatal.

7.11.1 Failed RCT READ

Simply stated, a READ command was issued to a block in one of the
copies of the RCT and an error was reported back for that command.
This is not necessarily fatal, as their are four copies of the RCT
and requirements are that at least one copy should have a good block
for the one needed. The error is reported in the following manner:

        Failed RCT READ;
          RCT copy: ddd
          Block no: ddd
          Endmsg status: oooooo

The RCT copy defines which of the four copies were being referenced
when the error was detected. Block no is the Block number (LBN) that
was being referenced within the copy, when the error was detected.
The Endmsg status should provide an indication of what the error
condition was and can be interpreted from the list of Status/Event
codes in section 7.2. Program execution will continue.

7.11.2 Failed RCT WRITE

Simply stated, a WRITE command was issued to a block in one of the
copies of the RCT and an error was reported back for that command.
This is not necessarily fatal, as their are four copies of the RCT
and requirements are that at least one copy should have a good block
for the one needed. The error is reported in the following manner:

        Failed RCT WRITE;
          RCT copy: ddd
          Block no: ddd
          Endmsg status: oooooo

The RCT copy defines which of the four copies were being referenced
when the error was detected. Block no is the Block number (LBN) that
was being referenced within the copy, when the error was detected.
The Endmsg status should provide an indication of what the error
condition was and can be interpreted from the list of Status/Event
codes in section 7.2. Program execution will continue.

7.11.3 Failed READ of all copies of RCT

This is serious. This error occurs when the program detects a

failure during a READ of all copies of a particular RCT block.
Replacements can not be done unless enough good RCT blocks exist to
make-up a good RCT copy. Not only can this program not do
replacements, but neither can any operating system that has the
capability of doing replacements. The error is reported in the
following manner:

        Failed READ of all copies of RCT;
           Block no: ddd
           Endmsg status: oooooo

The Block no is the block (LBN) that was being referenced. This same
block exists in each of the RCT copies. Thus, references to this
block in each of the copies resulted in an error. The Endmsg status
should provide an indication of what the error condition was and
can be interpreted from the list of Status/Event codes in section
7.2.   The Unit (drive) is dropped from testing.

The appropriate recovery for this kind of error, should be to first
make sure that the drive does not have a hardware problem causing
data errors that are not related to the media. If your certain that
the basic drive is functioning properly, the recovery would be to
try reformatting the media. The Format Control Table (FCT) must be
able to be used during the format (Read RA81-TT-19). Once this has
been accomplished, you can re-run this program, to assure yourself
that the drive is now operating properly. If the problem persists,
after having reformatted, the HDA/Pack may need replacement.

7.11.4 Failed WRITE of all copies of RCT

This is serious. This error occurs when the program detects a
failure during a WRITE of all copies of a particular RCT block.
Replacements can not be done unless enough good RCT blocks exist to
make-up a good RCT copy. Not only can this program not do
replacements, but neither can any operating system that has the
capability of doing replacements. The error is reported in the
following manner:

        Failed WRITE of all copies of RCT;
           Block no: ddd
           Endmsg status: oooooo

The Block no is the block (LBN) that was being referenced. This same
block exists in each of the RCT copies. Thus, references to this
block in each of the copies resulted in an error. The Endmsg status
should provide an indication of what the error condition was and
can be interpreted from the list of Status/Event codes in section
7.2.   The Unit (drive) is dropped from testing.

The appropriate recovery for this kind of error, should be to first
make sure that the drive does not have a hardware problem causing
data errors that are not related to the media. If your certain that
the basic drive is functioning properly, the recovery would be to
try reformatting the media. The Format Control Table (FCT) must be
able to be used during the format (Read RA81-TT-19). Once this has
been accomplished, you can re-run this program, to assure yourself

that the drive is now operating properly. If the problem persists,
after having reformatted, the HDA/Pack may need replacement.

7.11.5 No Null descriptor entry found in RCT

This is serious. The Replacement Control Table (RCT) contains
descriptors for 17,000 plus replacement blocks. If they are all
used (No Null Descriptor) then something big is wrong. The
descriptors could in fact all be used or possibly the RCT has
been written with Garbage, which makes it appear that all
replacement blocks have been used. Never-the-less, if this error
occurs, you should read RA81-TT-12. The recovery will be to reformat
the media. If the drive is operating normally, this should
fix the RCT, as long as the Format Control Table (FCT) is good
(See RA81-TT-19). You can re-run this program after having
reformatted, just to be sure the problem has been corrected. The
drive will be dropped form further testing. The error is reported
in the following manner:

```
No Null descriptor entry found in RCT;
   RCT is probably corrupt.
   Reformat media.
```

8.0 INFORMATIONAL MESSAGES

The following sections describe the information which may be observed
on the operator console during program execution.

8.1 Program Initialization

During program initialization, the following messages will appear
on the console for the appropriate replacement mode:

1.   MANUAL BAD BLOCK REPLACEMENT SELECTED.

2.   AUTOMATIC BAD BLOCK REPLACEMENT SELECTED.

8.2 Unit Initialization

Each Unit selected for test is brought Online, and the following
information is displayed for that unit (Drive). Remember a unit is not
necessarily the logical unit address of a drive.

```
INITIALIZATION INFORMATION FOR UNIT n
   Controller:  xDA
   Address:     172150 (Or what you provided when starting the program)
   Drive no:    d
   Drive type:  RAnn
   Volume SN:   dddd.
   Volume size: dddddd.
```

   -- The controller will be a UDA50, KDA50 or whatever it finds

-- Drive no: Is the Drive being initialized logical address
-- Drive Type: Is the model of drive seen (Like RA80/81/60 etc)
-- Volume SN: This is the Serial Number of the HDA or Pack. This
   information comes from the Format Control Table (FCT), which
   is read during the execution of the MSCP "ONLINE" command.
-- Volume size: This is the number of user LBN's on the media. For
   example, the RA81 formatted for a VAX (16 bit) has 891072
   (decimal) LBN's. Actually, starting with LBN zero, the last user
   LBN would be 891071.

8.2.1 Zero RCT size detected -- BBR not supported by this drive

This message is telling you that your running this program on a
drive that does not support Bad Block replacement. The zero RCT
size means that the drive has no "real" Replacement Control Table
and replacements are not possible. If think that you have bad blocks
on a drive like this, you should consult the appropriate service
manual for action you need to take. The message is reported in the
following manner:

```
    Zero RCT size detected:
      BBR not supported by this drive.
      Endmsg status:  000004
```

The Endmsg status will most likely not tell you much (Unless
something really out of the ordinary is happening). The Endmsg
status can be interpreted from the list of Status/Event codes in
section 7.2. The Unit (drive) is dropped from testing.

8.2.2 Scanning the RCT...

Just after typing the Initialization information for a drive that is
going to be tested (Above), the statement "Scanning the RCT..." is
typed. This portion of the program, attempts to find enough good RCT
blocks to account for a good complete copy. Any error that it finds
doing this is displayed. These messages (Shown Below) are only status
messages and DO NOT necessarily mean that the drive is defective. The
status is shown in the following manner:

```
    RCT copy 1, block 184 (     891256.)   Status/event:  000110
    RCT copy 1, block 185 (     891257.)   Status/Event:  000153
    RCT copy 2, block 184 (     892021.)   Status/event:  000350
```

These messages do not become critical until the same block number
shows an error for each copy. In the above example Block 184 is bad
in two copies. This is not fatal, since their are more than 2 copies
in our example. You will get a hard error report, if all copies of a
block are bad. The Endmsg status can be interpreted from the list
of Status/Event codes in section 7.2.

8.3 END OF PASS INFORMATION

The operator is notified of End of Pass, for the drive being tested, by the
following:

1. EOM reached.   Elapsed runtime hh:mm:ss
   The End of Media is reached in Automatic replacement mode
   when the highest LBN on the media has been tested.

2. EOF detected.
   The End of File is reached in Manual replacement mode when
   the operator enters a null LBN for replacement.

When EOP is reached in either Automatic or Manual mode, the following pass
summary is displayed for the drive being tested. This summary will also be
displayed by the operator "PRINT" command at the diagnostic supervisor prompt.

REPLACEMENT INFORMATION FOR THIS PASS

BAD BLOCKS reported: ddd.
ECC error detected: ddd.
FORCED ERRORs detected: ddd.
FORCED ERRORs written: ddd.
PRIMARY replacement: ddd.
SECONDARY replacements: ddd.
RBNs marked unusable:`   ddd.
Total replacements made: ddd.

The entries should be self-explanatory, after all the discussion above. The
Bottom line indicates the total number of replacements that were made, which
should not exceed a few hundred, unless something more is wrong with the drive.
The total secondary replacements should be on a ratio of about less than 10 to
every 100 primary replacements. The number of Forced Errors written, indicates
the number of Uncorrectable ECC (ECH) errors that were encountered. If you have
any forced errors written, or forced errors detected, you should not fail to
restore the customer data from the back-up that was made, before you started
execution of this program. If excessive replacements were made, read RA81-TT-12
for a possible explanation and recovery procedure.

8.4 ELAPSED RUNTIME

In Automatic mode, the following runtime message is periodically displayed to
indicate that execution is in progress:

LBN dddddd.   Elapsed runtime is hh:mm:ss

The LBN listed indicates where on the media the program is currently testing;
the elapsed runtime is zeroed when each Unit is selected for test.

```
                 a
1954                                    ;*LAST REVISION 01-APR-1985
1955                                    .IDENT   /01A/
2496                                    .SBTTL   Program Header
2513
2514                                      .ENABL  AMA,ABS
2515      002000                          .      =    2000
2516
2518                                    ;++
2519                                    ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
2520                                    ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
```

```
2521                              ;--
2522
2524
2525 002000              L$NAME::                ;DIAGNOSTIC NAME
     002000    132               .ASCII  /Z/
     002001    125               .ASCII  /U/
     002002    104               .ASCII  /D/
     002003    114               .ASCII  /L/
     002004    000               .BYTE   0
     002005    000               .BYTE   0
     002006    000               .BYTE   0
     002007    000               .BYTE   0
     002010              L$REV::                 ;REVISION LEVEL
     002010    101               .ASCII  /A/
     002011              L$DEPO::                ;0
     002011    060               .ASCII  /O/
     002012              L$UNIT::                ;NUMBER OF UNITS
     002012    000001            .WORD   T$PTHV
     002014              L$TIML::                ;LONGEST TEST TIME
     002014    000000            .WORD   0
     002016              L$HPCP::                ;POINTER TO H.W. QUES.
     002016    042146            .WORD   L$HARD
     002020              L$SPCP::                ;POINTER TO S.W. QUES.
     0C2020    000000            .WORD   0
     002022              L$HPTP::                ;PTR. TO DEF. H.W. PTABLE
     002022    002132            .WORD   L$HW
     002024              L$SPTP::                ;PTR. TO S.W. PTABLE
     002024    002146            .WORD   L$SW
     002026              L$LADP::                ;DIAG. END ADDRESS
     002026    047476            .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    000000            .WORD   0
     002036              L$APT::                 ;APT EXPANSION
     002036    000000            .WORD   0
     002040              L$DTP::                 ;PTR. TO DISPATCH TABLE
     002040    002124            .WORD   L$DISPATCH
     002042              L$PRIO::                ;DIAGNOSTIC RUN PRIORITY
     002042    000340            .WORD   PRIO7
     002044              L$ENVI::                ;FLAGS DESCRIBE HOW IT WAS SETUP
     002044    000000            .WORD   0
     002046              L$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    004300            .WORD   L$DVTYP
     002062              L$REPP::                ;PTR. TO REPORT CODE
```

```
         002062  035132              .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$DUT::                  ;PTR. TO DROP UNIT CODE
         002072  040146              .WORD   L$DU
         002074             L$LUN::                  ;LUN FOR EXERCISERS TO FILL
         002074  000000              .WORD   0
         002076             L$DESP::                 ;POINTER TO DIAG. DESCRIPTION
         002076  004324              .WORD   L$DESC
         002100             L$LGAD::                 ;GENERATE SPECIAL AUTOLOAD EMT
         002100  104035              EMT     E$LOAD
         002102             L$ETP::                  ;POINTER TO ERRTBL
         002102  002166              .WORD   L$ERRTBL
         002104             L$ICP::                  ;PTR. TO INIT CODE
         002104  035726              .WORD   L$INIT
         002106             L$CCP::                  ;PTR. TO CLEAN-UP CODE
         002106  040124              .WORD   L$CLEAN
         002110             L$ACP::                  ;PTR. TO AUTO CODE
         00211C  040122              .WORD   L$AUTO
         0C2112             L$PRT::                  ;PTR. TO PROTECT TABLE
         002112  035720              .WORD   L$PROT
         002114             L$TEST::                 ;TEST NUMBER
         002114  000000              .WORD   0
         002116             L$DLY::                  ;DELAY COUNT
         002116  000000              .WORD   0
         002120             L$HIME::                 ;PTR. TO HIGH MEM
         002120  000000              .WORD   0
2526
```

```
    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 002122  000002                     .WORD    2
      002124            L$DISPATCH::
      002124  040314                     .WORD    T1
      002126  040624                     .WORD    T2
    9
```

```
     1                                    .SBTTL   DEFAULT HARDWARE P-TABLE
     2
     3                            ;++
     4                            ; THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
     5                            ; 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 002130  000005                     .WORD    L10000-L$HW/2
       002132             L$HW::
       002132             DFPTBL::
    11 002132  172150              .WORD    172150            ; UNIBUS ADDRESS
    12 002134  000154              .WORD    154               ; VECTOR ADDRESS
    13 002136  000005              .WORD    5.                ; BR LEVEL
    14 002140  000077              .WORD    63.               ; UNIBUS BURST RATE
    15 002142  000000              .WORD    0.                ; LOGICAL DRIVE NUMBER
    16
    17 002144             L10000:
```

SEQ 0040

```
    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
    7                     ; AT RUN TIME.
    8                     ;--
    9
   10 002144  000010                    .WORD    L10001-L$SW/2
      002146             L$SW::
      002146             SFPTBL::
   11
   12 002146  000000     OPHELP::        .WORD    0       ; Set to print operator help
   13 002150  000000     MANRPL::        .WORD    0       ; Set if manual replacement
   14 002152  000001     AUTORC::        .WORD    1       ; Set for automatic crash recovery
   15 002154  000001     DSPRPL::        .WORD    1       ; Set to display replacements
   16 002156  000000     DSPRCT::        .WORD    0       ; If set print RCT descriptors
   17 002160  000001     ENBRPL::        .WORD    1       ; Set to disable replacement
   18 002162  000001     ENBERRL::       .WORD    1       ; Set to enable Errorlog packets
   19 002164  000001     ENBWFE::        .WORD    1       ; Enable write FE
   20
   21 002166             L10001:
```

```
     1                              .SBTTL   Global equates section
     2
     4                         ;++
     5                         ; THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
     6                         ; ARE USED IN MORE THAN ONE TEST.
     7                         ;--
     8
     9                         ;
                              ; 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
                              ;
                              ;                              ; BIT POSITION IN SECOND STATUS WORD
     000040                  EF.START==      32.        ; (100000) START COMMAND WAS ISSUED
     000037                  EF.RESTART==    31.        ; (040000) RESTART COMMAND WAS ISSUED
     000036                  EF.CONTINUE==   30.        ; (020000) CONTINUE COMMAND WAS ISSUED
     000035                  EF.NEW==        29.        ; (010000) A NEW PASS HAS BEEN STARTED
     000034                  EF.PWR==        28.        ; (004000) A POWER-FAIL/POWER-UP OCCURRED
                              ;
                              ;
                              ; PRIORITY LEVEL DEFINITIONS
                              ;
     000340                  PRI07== 340
     000300                  PRI06== 300
     000240                  PRI05== 240
     000200                  PRI04== 200
     000140                  PRI03== 140
     000100                  PRI02== 100
```

```
        000040                  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   XDA bit definitions
  2
  3
  4                    ;xDA SA 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                    ;xDA SA REGISTER ERROR STATUS BITS
 14
 15      003777        SA.ERC  = 003777                 ;ERROR CODE
 16
 17
 18                    ;xDASA 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                    ;xDASA REGISTER STEP 1 RESPONSE BITS
 32
 33      002000        SA.NV   = 002000                 ;NON SETTABLE INTERRUPT VECTOR
 34      001000        SA.QB   = 001000                 ;22 BIT ADDRESS BUS
 35      000400        SA.DI   = 000400                 ;ENHANCED DIAGNOSTICS
 36      000100        SA.MP   = 000100                 ;MAPPING BIT
 37      ;               000377                         ;ALL BITS RESERVED
 38
 39
 40                    ;xDASA REGISTER STEP 2 SEND BITS
 41
 42      000001        SA.PRG  = 000001                 ;ENABLE VAX UNIBUS ADAPTER PURGE INTERRUPT
 43      ;               177776                         ;LOW ORDER MESSAGE RING BYTE ADDRESS
 44
 45
 46                    ;xDASA REGISTER STEP 2 RESPONSE BITS
 47
 48      000007        SA.MSE  = 000007                 ;MESSAGE RING LENGTH ECHO
 49      000070        SA.CME  = 000070                 ;COMMAND RING LENGTH ECHO
 50      ;               000100                         ;RESERVED
 51      000200        SA.STE  = 000200                 ;STEP ECHO
 52      003400        SA.CTP  = 003400                 ;CONTROLLER TYPE
 53
 54
 55                    ;xDASA REGISTER STEP 3 SEND BITS
 56
 57      ;               077777                         ;HIGH ORDER MESSAGE RING BYTE ADDRESS
```

```
58        100000              SA.TST  = 100000              ;PURGE POLE TEST ENABLE
59
60
61                            ;xDASA REGISTER STEP 3 RESPONSE BITS
62
63        000177              SA.VCE  = 000177              ;INTERRUPT VECTOR ECHO
64        000200              SA.INE  = 000200              ;INTERRUPT ENABLE ECHO
65        000400              SA.NVE  = 000400              ;VECTOR NOT PROGRAMMABLE
66                            ;         003000              ;RESERVED
67
68
69                            ;xDASA REGISTER STEP 4 SEND BITS
70
71        000001              SA.GO   = 000001              ;GO BIT TO START xDA FIRMWARE
72        000002              SA.LFC  = 000002              ;LAST FAILURE CODE REQUEST
73        000374              SA.BST  = 000374              ;BURST LEVEL
74
75
76                            ;xDASA REGISTER STEP 4 RESPONSE BITS
77
78        000017              SA.MCV  = 000017              ;xDA MICROCODE VERSION
79        000360              SA.CNT  = 000360              ;CONTROLLER TYPE
80                            ;         003400              ;RESERVED
```

```
     1                                    .SBTTL  Host Communication area definitions
     2
     3                                 ;COMMAND/MESSAGE RING BIT DEFINITIONS
     4
     5          100000                 RG.OWN  = 100000                        ;SET WHEN xDA 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
    25          000000                 HC.INT  = 0.                            ;INTERRUPT INDICATOR WORDS START
    26
    27          000004                 HC.MSG  = HC.INT+HC.ISZ                 ;MESSAGE RING START
    28          000006                 HC.MCT  = HC.MSG+2.                     ;MESSAGE RING CONTROL WORD
    29
    30          000010                 HC.CMD  = HC.MSG+HC.RSZ                 ;COMMAND RING START
    31          000012                 HC.CCT  = HC.CMD+2.                     ;COMMAND RING CONTROL WORDS
    32
    33          000014                 HC.MEV  = HC.CMD+HC.RSZ                 ;MESSAGE ENVELOPE START
    34          000020                 HC.MPK  = HC.MEV+HC.ESZ                 ;MESSAGE PACKET START
    35
    36          000100                 HC.CEV  = HC.MPK+HC.PSZ                 ;COMMAND ENVELOPE START
    37          000104                 HC.CPK  = HC.CEV+HC.ESZ                 ;COMMAND PACKET START
    38
    39          000164                 HC.SIZ  = HC.CPK+HC.PSZ                 ;TOTAL SIZE OF HOST COMMUNICATION AREA
```

```
 1              .SBTTL   Host Communication area layout
 2
 3      ;       +-------------------------------------------------+
 4      ;   HC.INT )           INTERRUPT INDICATORS              )   4 BYTES
 5      ;          )                                             )
 6      ;       +-------------------------------------------------+
 7      ;   HC.MSG )         MESSAGE (RESPONSE) RING             )   4 BYTES
 8      ;   HC.MCT )                                             )
 9      ;       +-------------------------------------------------+
10      ;   HC.CMD )             COMMAND RING                    )   4 BYTES
11      ;   HC.CCT )                                             )
12      ;       +-------------------------------------------------+
13      ;   HC.MEV )       MESSAGE (RESPONSE) ENVELOPE           )   4 BYTES
14      ;          )                                             )
15      ;   HC.MPK )       MESSAGE (RESPONSE) PACKET             )  48 BYTES
16      ;          )                                             )
17      ;          )                                             )
18      ;          )                                             )
19      ;          )                                             )
20      ;       +-------------------------------------------------+
21      ;   HC.CEV )           COMMAND ENVELOPE                  )   4 BYTES
22      ;          )                                             )
23      ;   HC.CPK )           COMMAND PACKET                    )  48 BYTES
24      ;          )                                             )
25      ;          )                                             )
26      ;          )                                             )
27      ;          )                                             )
28      ;       +-------------------------------------------------+
29      ;
30      ;          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 ATTENTION 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                                    ;GENERIC COMMAND PACKET OFFSETS
    4
    5          000000                    P.CRF   = 0.                              ;COMMAND REFERENCE NUMBER
    6          000004                    P.UNIT  = 4.                              ;UNIT NUMBER
    7          000010                    P.OPCD  = 8.                              ;OPCODE
    8          000012                    P.MOD   = 10.                             ;MODIFIERS
    9          000014                    P.BCNT  = 12.                             ;BYTE COUNT
   10          000020                    P.BUFF  = 16.                             ;BUFFER DESCRIPTOR
   11          000020                    P.UADR  = 16.                             ;UNIBUS ADDRESS OF BUFFER DESCRIPTOR
   12          000034                    P.LBN   = 28.                             ;LOGICAL BLOCK NUMBER
   13
   14
   15                                    ;ABORT AND GET COMMAND STATUS COMMAND PACKET OFFSETS
   16
   17          000014                    P.OTRF  = 12.                             ;OUTSTANDING REFERENCE NUMBER
   18
   19
   20                                    ;ONLINE AND SET UNIT CHARACTERISTICS COMMAND PACKET OFFSETS
   21
   22          000016                    P.UNFL  = 14.                             ;UNIT FLAGS
   23          000020                    P.HSTI  = 16.                             ;HOST IDENTIFIER / RESERVED
   24          000034                    P.ELGF  = 28.                             ;ERROR LOG FLAGS
   25          000040                    P.SHUN  = 32.                             ;SHADOW UNIT
   26          000042                    P.CPSP  = 34.                             ;COPY SPEED
   27
   28
   29                                    ;REPLACE COMMAND PACKET OFFSETS
   30
   31          000014                    P.RBN   = 12.                             ;REPLACEMENT BLOCK NUMBER
   32
   33
   34                                    ;SET CONTROLLER CHARACTERISTICS COMMAND PACKET OFFSETS
   35
   36          000014                    P.VRSN  = 12.                             ;MSCP VERSION
   37          000016                    P.CNTF  = 14.                             ;CONTROLLER FLAGS
   38          000020                    P.HTMO  = 16.                             ;HOST TIMEOUT
   39          000022                    P.USEF  = 18.                             ;USE FRACTION
   40          000024                    P.TIME  = 20.                             ;QUAD-WORD TIME AND DATE
   41
   42
   43                                    ;MAINTENANCE READ AND MAINTENANCE WRITE COMMAND PACKET OFFSETS
   44
   45          000034                    P.RGID  = 28.                             ;REGION ID
   46          000040                    P.RGOF  = 32.                             ;REGION OFFSET
   47
   48
   49                                    ;EXECUTE SUPPLIED PROGRAM COMMAND PACKET OFFSETS
   50
   51          0C0024                    P.DMDT  = 20.                             ;DMDT TERMINAL ADDRESS (MAINT WRITE ONLY)
   52          000034                    P.OVRL  = 28.                             ;BUFFER DESCRIPTOR FOR OPERLAYS
```

```
         1                              .SBTTL   End packet offsets
         2
         3                           ;GENERIC END PACKET OFFSETS
         4
         5          000000           P.CRF   = 0.                          ;COMMAND REFERENCE NUMBER
         6          000004           P.UNIT  = 4.                          ;UNIT NUMBER
         7          000010           P.OPCD  = 8.                          ;OPCODE (ALSO CALLED ENDCODE)
         8          000011           P.FLGS  = 9.                          ;END PACKET FLAGS
         9          000012           P.STS   = 10.                         ;STATUS
        10          000014           P.BCNT  = 12.                         ;BYTE COUNT
        11          000034           P.FBBK  = 28.                         ;FIRST BAD BLOCK
        12
        13
        14                           ;        Error log packet offsets
        15
        16          000000           L.CRF   ==       0.                   ; Command reference number
        17          000010           L.FMT   ==       8.                   ; Format specifier
        18          000011           L.FLGS  ==       9.                   ; Error log flags
        19          000012           L.EVNT  ==       10.                  ; Event code
        20          000030           L.BADR  ==       24.                  ; Host address
        21          000050           L.HDCD  ==       40.                  ; Header code (LBN)
        22          000054           L.SDI   ==       44.                  ; SDI information
        23          000060           L.DS    ==       L.SDI+4              ; Drive status info
        24
        25                           ;GET COMMAND STATUS END PACKET OFFSETS
        26
        27          000014           P.OTRF  = 12.                         ;OUTSTANDING REFERENCE NUMBER
        28          000020           P.CMST  = 16.                         ;COMMAND STATUS
        29
        30
        31                           ;GET UNIT STATUS END PACKET OFFSETS
        32
        33          000014           P.MLUN  = 12.                         ;MULTI-UNIT CODE
        34          000016           P.UNFL  = 14.                         ;UNIT FLAGS
        35          000020           P.HSTI  = 16.                         ;HOST IDENTIFIER
        36          000024           P.UNTI  = 20.                         ;UNIT IDENTIFIER
        37          000034           P.MEDI  = 28.                         ;MEDIA TYPE IDENTIFIER
        38          000040           P.SHUN  = 32.                         ;SHADOW UNIT
        39          000042           P.SHST  = 34.                         ;SHADOW STATUS
        40          000044           P.TRKS  = 36.                         ;TRACK SIZE
        41          000046           P.GRPS  = 38.                         ;GROUP SIZE
        42          000050           P.CYLS  = 40.                         ;CYLINDER SIZE
        43          000054           P.RCTS  = 44.                         ;RCT TABLE SIZE
        44          000056           P.RBNS  = 46.                         ;RBNS / TRACK
        45          000057           P.RCTC  = 47.                         ;RCT COPIES
        46
        47
        48                           ;ONLINE AND SET UNIT CHARACTERISTICS END PACKET AND AVAILABLE
        49                           ;ATTENTION MESSAGE OFFSETS
        50
        51          000014           P.MLUN  = 12.                         ;MULTI-UNIT CODE
        52          000016           P.UNFL  = 14.                         ;UNIT FLAGS
        53          000020           P.HSTI  = 16.                         ;HOST IDENTIFIER
        54          000024           P.UNTI  = 20.                         ;UNIT IDENTIFIER
        55          000032           P.MODEL = P.UNTI+6                    ; Model Identifier
        56          000033           P.CLASS = P.UNTI+7                    ; Class Identifier
        57          000034           P.MEDI  = 28.                         ;MEDIA TYPE IDENTIFIER
```

```
58          000040              P.SHUN  = 32.                    ;SHADOW UNIT
59          000042              P.SHST  = 34.                    ;SHADOW STATUS
60          000044              P.UNSZ  = 36.                    ;UNIT SIZE
61          000050              P.VSER  = 40.                    ;VOLUME SERIAL NUMBER
62
63
64                              ;SET CONTROLLER CHARACTERISTICS END PACKET OFFSETS
65
66          000014              P.VRSN  = 12.                    ;MSCP VERSION
67          000016              P.CNTF  = 14.                    ;CONTROLLER FLAGS
68          000020              P.CTMO  = 16.                    ;CONTROLLER TIMEOUT
69          000022              P.CNCL  = 18.                    ;CONTROLLER COMMAND LIMIT
70          000024              P.CNTI  = 20.                    ;CONTROLLER ID
71
72
73                              ;GET DUST STATUS END PACKET OFFSETS    .
74
75          000014              P.DEXT  = 12.                    ;EXTENSION FOR DOWNLINE LOADABLE PROGRAM
76          000017              P.DFLG  = 15.                    ;FLAGS
77          000020              P.DPRG  = 16.                    ;PROGRESS INDICATOR FOR REMOTE PROGRAM
78          000024              P.DTMO  = 20.                    ;TIMEOUT
```

```
     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
    28
    29                          ;       Data error subcodes
    30
    31      000110              SB.HCE =        110          ; Header compare error
    32      000150              SB.DST =        150          ; Data sync timeout
    33      000350              SB.ECC =        350          ; Uncorrectable ECC
    34
```

```
    1                                   .SBTTL   Controller table definitions
    2
    3                                 ;ONE TABLE WILL BE SET UP BY INITIALIZE SECTION FOR EACH xDA 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 CTLRTBL.
    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
   15        000040                 CT.U50  = BIT5                      ;CONTROLLER IS xDA50 IF SET/xDA52 IF CLEARED
   16        000020                 CT.REQ  = BIT4                      ;BUFFER HAS BEEN GIVEN TO xDA FOR REQUEST
   17                                                                   ;SET WHENEVER READ STUD DATA COMMAND GIVEN TO xDA
   18        000010                 CT.MSG  = BIT3                      ;MESSAGE RESPONSE RECEIVED
   19                                                                   ;WHENEVER THIS BIT IS SET, CT.CMD IS CLEARED
   20        000004                 CT.CMD  = BIT2                      ;COMMAND ISSUED, WAITING FOR RESPONSE
   21
   23        000000                 C.UADR  = 0                        ;UNIBUS ADDRESS OF xDAIP REGISTER
   24        000002                 C.VEC   = 2                        ;VECTOR ADDRESS/BR LEVEL
   25        000004                 C.BST   = 4                        ;BURST LEVEL
   26        000006                 C.DRVN  = 6                        ;DRIVE NUMBER TO TEST
   27        000010                 C.JSR   = 10                       ;INTERRUPT SERVICE ROUTINE FOR CONTROLLER
   28        000012                 C.JAD   = 12                       ;THESE TWO WORDS LOADED WITH [JSR R0    xDASRV]
   29        000014                 C.FLG   = 14                       ;FLAGS
   30        000016                 C.HCOM  = 16                       ;BEGINNING ADRS OF HOST COMM AREA IN MEMORY
   31        000020                 C.DRVT  = 20                       ;POINTER TO DRIVE TABLE
   32        000022                 C.TO    = 22                       ;TIMEOUT COUNTER
   33        000024                 C.TOH   = 24                       ;   (TWO WORDS)
   34        000026                 C.REF   = 26                       ;COMMAND REFERENCE NUMBER
   35
   36        000030                 C.SIZE  = 30                       ;SIZE OF CONTROLLER TABLE IN BYTES
```

```
    1
    2                                 ;DRIVE TABLE DEFINITIONS
    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        000077                   DT.UNT = 000077                     ; LOGICAL UNIT NUMBER OF DRIVE
    9
   10        100000                   DT.AVL = BIT15                      ; SET WHEN NOT AVAILABLE FOR TESTING
   11        040000                   D.IW   = BIT14                      ;INITIAL WRITE
   12        020000                   D.DCY  = BIT13                      ;DIAGNOSTIC CYLINDERS
   13        010000                   D.ECC  = BIT12                      ;ECC CORRECTION ENABLED
   14        004000                   D.RO   = BIT11                      ;READ ONLY
   15        002000                   D.WO   = BIT10                      ;WRITE ONLY
   16        001000                   D.RET  = BIT9                       ;RETRIES ENABLED
   17        000400                   D.CYL  = BIT8                       ;START/END CYLINDERS SPECIFIED
   18        000100                   D.SEQ  = BIT6                       ;SEQUENTIAL ACCESS
   19        000040                   D.BE   = BIT5                       ;BEGIN/END BLOCKS USED
   20        000020                   D.TR   = BIT4                       ;WHEN D.BE=0: 1 - TRACKS, 0 - GROUPS
   21        000010                   D.WC   = BIT3                       ;WRITE CHECKS ENABLED
   22        000004                   D.WCA  = BIT2                       ;ALWAYS WRITE CHECK
   23        000002                   D.DC   = BIT1                       ;DATA COMPARES ENABLED
   24        000001                   D.DCA  = BIT0                       ;ALWAYS DATA COMPARE
   25
   27        000000                   D.PAT  = 0                          ;DATA PATTERN NUMBER
   28        000002                   D.BB   = 2                          ;BAD BLOCK COUNT
   29        000004                   D.BB01 = 4                          ;BAD BLOCK 1
   30        000010                   D.BB02 = 10                         ;          2
   31        000014                   D.BB03 = 14                         ;          3
   32        000020                   D.BB04 = 20                         ;          4
   33        000024                   D.BB05 = 24                         ;          5
   34        000030                   D.BB06 = 30                         ;          6
   35        000034                   D.BB07 = 34                         ;          7
   36        000040                   D.BB08 = 40                         ;          8
   37        000044                   D.BB09 = 44                         ;          9
   38        000050                   D.BB10 = 50                         ;         10
   39        000054                   D.BB11 = 54                         ;         11
   40        000060                   D.BB12 = 60                         ;         12
   41        000064                   D.BB13 = 64                         ;         13
   42        000070                   D.BB14 = 70                         ;         14
   43        000074                   D.BB15 = 74                         ;         15
   44        000100                   D.BB16 = 100                        ;         16
```

```
     1
     2      000104              D.BEC   = 104        ;BEGIN/END SET COUNT
     3      000106              D.BGN1  = 106        ;BEGIN BLOCK 1
     4      000112              D.END1  = 112        ;END
     5      000116              D.BGN2  = 116        ;BEGIN BLOCK 2
     6      000122              D.END2  = 122        ;END
     7      000126              D.BGN3  = 126        ;BEGIN BLOCK 3
     8      000132              D.END3  = 132        ;END
     9      000136              D.BGN4  = 136        ;BEGIN BLOCK 4
    10      000142              D.END4  = 142        ;END
    11      000146              D.BCYL  = 146        ;BEGIN CYLINDER
    12      000152              D.ECYL  = 152        ;END CYLINDER
    13      000156              D.SERN  = 156        ;DRIVE SERIAL NUMBER
    14
    15      000164              D.SIZE  = 164        ;SIZE OF DRIVE TABLE IN BYTES
```

```
    1                          .SBTTL  Miscellaneous definitions
    2
    3    000620      CACHESIZE        ==      400.        ; Number entries in Caching table
    4    001000      BLKSIZ  == 512.                      ; Number of bytes per block
    5    000200      GRPSIZ  == 128.                      ; ACCESS group size
    6    000200      ENTRYS  == 128.                      ; Number entries per RCT descriptor
    7
```

```
     1                                          .SBTTL   RCT Logical block 0 definitions
     2
     3              000000                VOLSN   == 0                    ;Volume serial number    (LO WORD)
     4              000010                RFLAGS  == 10                   ;Replacement flags
     5              000014                LBNLO   == 14                   ;LBN being replaced       (LO WORD)
     6              000020                RBNLO   == 20                   ;RBN replacement          (LO WORD)
     7              000024                BADRBN  == 24                   ;Bad RBN                  (LO WORD)
     8              000030                CACHID  == 30                   ;Cache ID                 (LO WORD)
     9              000040                INCARN  == 40                   ;Cache incarnation no.    (LO WORD)
    10              000044                TIMDAT  == 44                   ;VAX time and date        (LO WORD)
    11
    12
    13                                    ;REPLACEMENT FLAG DEFINITIONS
    14
    15              000001                WBC     == BIT0                 ;Write-Back caching
    16              000002                VWP     == BIT1                 ;Volume write protect
    17              000200                FEF     == BIT7                 ;Forced Error flag
    18              020000                BRF     == BIT13                ;Bad RBN flag
    19              040000                P2      == BIT14                ;Phase 2 flag
    20              100000                P1      == BIT15                ;Phase 1 flag
    21
```

```
    1                                 .SBTTL   Global data section
    2                             ;
    3                             ;        Program data areas
    4
    5 002166                      L$ERRTBL::
      002166  000000              ERRTYP::       .WORD   0
      002170  000000              ERRNBR::       .WORD   0
      002172  000000              ERRMSG::       .WORD   0
      002174  000000              ERRBLK::       .WORD   0
    6
    7 002176  000000              FFREE::        .WORD   0       ;FIRST FREE WORD IN MEMORY
    8 002200  000000              FSIZE::        .WORD   0       ;SIZE OF FREE MEMORY IN WORDS
    9 002202  000000              LOADFLG::      .WORD   0       ; Cleared by program load
   10 002204  000000              RTFLAG::       .WORD   0       ; Runtime flag
   11 002206  000000              CLKFLG::       .WORD   0       ; Set if KW11 present
   12
   13                             ;        KW11 clock parameters
   14
   15 002210  000000              KW.CSR: .WORD  0               ;CSR OF CLOCK
   16 002212  000000              KW.BRL: .WORD  0               ;BR LEVEL
   17 002214  000000              KW.VEC: .WORD  0               ;VECTOR
   18 002216  000000              KW.HZ:  .WORD  0               ;HERTZ (50. OR 60.)
   19 002220  000000  000000      KW.EL:  .WORD  0,0             ;ELAPSED TIME
   20
   21 002224  000000              NXMAD:  .WORD  0               ;SET TO ALL ONES BY NON-EXISTANT ADDRESS
   22 002226  177777              KTMEM:  .WORD  -1              ;SET TO ALL ONES IF NO KT EXISTS
   23
   24
   25                             ;        UUT tables and pointers
   26
   27 002230                      CTLRLST::      .BLKW   10.     ; Controller address table list
   28 002254                      CACHLST::      .BLKW   10.     ; Bad block caching table list
   29 002300  000000              TBLNDX::       .WORD   0       ; Table index
   30 002302  000000              PTABLE::       .WORD   0       ; PTABLE address ptr
   31 002304  000000              CTLRTBL::      .WORD   0       ; Address of current controller table
   32 002306  000000              DRIVTBL::      .WORD   0       ; Address of current drive table
   33 002310  000000              CACHTBL::      .WORD   0       ; Current caching table
   34 002312  000000              RCTTBL::       .WORD   0       ; RCT bad block address table
   35
   36                             ;        SA register parameter storage
   37
   38 002314  000000              SADDR::        .WORD   0       ; SA register address
   39 002316  000000              SAVAL::        .WORD   0       ; Saved SA value
   40 002320  000000              SAEXP::        .WORD   0       ; Expected SA value
   41
   42 002322  000000              IPVAL::        .WORD   0       ; Saved IP value
   43
```

```
    1                                    ;
    2                                    ;          Communications tables and pointers
    3
    4  002324  000000                    HCOM::           .WORD   0           ; Address of host communication area
    5  002326  000000                    CMDPCK::         .WORD   0           ; Address of command packet
    6  002330  000000                    MSGPCK::         .WORD   0           ; Address of message packet
    7  002332  000000                    INTRCV::         .WORD   0           ; Interrupt received flag
    8
    9  002334  001000  000000            BCNT::           .WORD   BLKSIZ,0    ; Number of bytes to transfer
   10  002340  042472                    UADR::           .WORD   DBUFF       ; contains UNIBUS buffer address
   11
   12                                    ;          Command packet parameter storage
   13
   14  002342  000000                    CMDREF::         .WORD   0           ; Command reference
   15  002344  000000                    OPCODE::         .WORD   0           ; MSCP command
   16  002346  000000                    CMDMOD::         .WORD   0           ; Command modifiers
   17
   18                                    ;          End packet parameter storage
   19
   20  002350  000000                    ENDREF::         .WORD   0           ; End packet reference
   21  002352  000000                    ECODE::          .WORD   0           ; MSCP endcode
   22  002354  000000                    EMSTAT::         .WORD   0           ; End msg status
   23  002356  000000                    FLAGS::          .WORD   0           ; End msg flags
   24
   25                                    ;          Error packet parameter storage
   26
   27  002360  000000                    ERRREF::         .WORD   0           ; Error reference
   28  002362  000000                    EVENT::          .WORD   0           ; Event code
   29  002364  000000                    ERFLAG::         .WORD   0           ; Error msg flags
   30  002366  000000                    ERRFMT::         .WORD   0           ; Error format
   31  002370  000000                    EXTMSK::         .WORD   0           ; Extended error maskflags
   32
   33  002372  000000                    ERLFMT::         .WORD   0           ; Error log format spec
   34  002374  000000  000000            ERLLBN::         .WORD   0,0         ; Error log LBN
   35  002400  000000  000000            HADDR::          .WORD   0,0         ; Error log host address
   36  002404  000000                    DCODE::          .WORD   0           ; Drive error code
   37  002406                            ERLDS::          .BLKW   4           ; Error log drive/status error info
   38
```

```
   1
   2                              ;         Drive parameter area
   3
   4 002416  000000              RCTSIZ::           .WORD   0           ; Number of blocks in the RCT (w/ pad)
   5 002420  000000              TRKSIZ::           .WORD   0           ; Number of LBN'S per track
   6 002422     000              RBNS::             .BYTE   0           ; Number of REPLACEMENT BLOCKS per track
   7 002423     000              COPIES::           .BYTE   0           ; Number of copies of the RCT
   8 002424  000000              RCTCOPY::          .WORD   0           ; Current RCT copy
   9 002426  000000              RCTMASK::          .WORD   0           ; RCT bad copy mask
  10 002430  000000  000000      UNSIZ::            .WORD   0,0         ; Number of blocks in of LBN space
  11                                                                   ; RCT start
  12 002434  000000              PRIMRY::.WORD      0                  ;primary replacement counter
  13 002436  000000              SCNDRY::.WORD      0                  ;secondary replacement counter
  14 002440  000000              UNUSE:: .WORD      0                  ;unusable replacement counter
  15
  16 002442  000000              STATUS::           .WORD   0           ; Subroutine return value
  17 002444  000000              TMPSTAT::          .WORD   0           ; Multi-READ/WRITE temp status
  18 002446  000000  000000      LBN::              .WORD   0,0         ;LBN being replaced
  19 002452  000000  000000      RBN::              .WORD   0,0         ;RBN replacement
  20 002456  000000  000000      RCTLBN::           .WORD   0,0         ; RCT LBN
  21 002462  000000  000000      MATRBN::           .WORD   0,0         ;RBN which matched LBN being replaced
  22 002466  000000              RECOVR::           .WORD   0           ;recovery flag, Phase 1 or 2 recovery
  23                                                                   ; no recovery=0, recovery=1
  24 0C2470  000000              RDVALD::.WORD      0                  ; Valid data flag for READLBN
  25 002472     000              VIBUFF::.BYTE      0                  ;Valid image flag,
  26 002473     000              VBUFF1::.BYTE      0                  ;Valid buffer one flag,
  27 002474     000              VBUFF2::.BYTE      0                  ;Valid buffer two flag,
  28 002475     000              ERRCNT::.BYTE      0                  ;error count
  29                                                .EVEN
  30
  31 002476  000000              RCTBLK::.WORD      0                  ;RCT logical block
  32 002500  000000              MATBLK::.WORD      0                  ;RCT logical block for matched LBN
  33 002502  000000              RCTOFF::.WORD      0                  ;RBN offset within RCT logical block
  34 002504  000000              MATOFF::.WORD      0                  ;RBN offset within RCT logical block
  35                                                                   ;for matched LBN
  36 002506  000000              STROFF::.WORD      0                  ;starting offset within RCT block
  37 002510  000000              DELTA:: .WORD      0                  ;change in offset while for RBN search
  38 002512     000              EMPTY:: .BYTE      0                  ;empty flag
  39 002513     000              RESCAN::.BYTE      0                  ;re-scan flag
  40 002514     000              MATFLG::.BYTE      0                  ;match status.
  41                                                                   ; 0=No match for LBN being replaced
  42                                                                   ; 1=RBN in 'MATRBN' was match for LBN
  43                                                                   ;    being replaced
  44 002515     000              SFLAG:: .BYTE      0                  ;search status.
  45                                                                   ; 0=Primary empty
  46                                                                   ; 1=Secondary empty
  47                                                                   ; 2=Full table (no match)
  48                                                .EVEN
  49
  50
  51                              ;         LBNSTAT: LBN status word
  52
  53          000001             TF.FER   ==        BIT0                ; FER
  54          000002             TF.HCE   ==        BIT1                ; HCE
  55          000004             TF.DST   ==        BIT2                ; DST
  56          000010             TF.ECC   ==        BIT3                ; ECC
  57          000020             TF.SEX   ==        BIT4                ; EF.SEX
```

```
 58           000040              TF.LOG   ==     BIT5                  ; EF.LOG
 59           000100              TF.BBU   ==     BIT6                  ; EF.BBU
 60           000200              TF.BBR   ==     BIT7                  ; EF.BBR
 61           000400              TF.WFE   ==     BIT8                  ; WFE
 62           100000              TF.BAD   ==     BIT15                 ; Set if image is invalid
 63
 64 002516    000000              LBNSTAT::       .WORD    0            ; LBN status word
 65 002520    000000              RPLSTAT::       .WORD    0            ; Replacement status
 66 002522    000000              LBNDX::         .WORD    0            ; LBN descriptor index
 67 002524    000000              RDSTAT::        .WORD    0            ; READLBN read status
 68 002526    000000              WRSTAT::        .WORD    0            ; READLBN write status
 69
 70 002530    000000              FERCNT::        .WORD    0            ; FER count
 71 002532    000000              HCECNT::        .WORD    0            ; HCE count
 72 002534    000000              DSTCNT::        .WORD    0            ; DST error count
 73 002536    000000              ECCCNT::        .WORD    0            ; Uncorrectable ECC count
 74 002540    000000              BBRCNT::        .WORD    0            ; BBR count
 75 002542    000000              WFECNT::        .WORD    0            ; WRITE Forced error count
 76
 77
 78 002544    000000              STEP::          .WORD    0            ; Current STEP value
 79 002546    000000              NXTSTEP::       .WORD    0            ; Next STEP value
 80 002550    000000              ERRSTEP::       .WORD    0            ; Error STEP indicator
 81 0C2552    000000              BASEMSG::       .WORD    0            ; Error message address
 82 002554    000000              INCRLBN::       .WORD    0            ; LBN Increment
 83 002556    000000              RDCNT::         .WORD    0            ; Reread counter
 84 002560    000000              PHASE::         .WORD    0            ; Replacement phase
 85
 86 002562    000000   000000     GLBN::          .WORD    0,0          ; ACCESS group LBN
 87 002566    000000   000001     GBCNT::         .WORD    0,1          ; ACCESS group byte count
 88 002572    000200              BLKCNT::        .WORD    GRPSIZ       ; ACCESS block count
 89 002574    000000   000000     ACCESS::        .WORD    0,0          ; ACESS number
 90 002600    000000   000000     MINLBN::        .WORD    0,0          ; Lowest LBN
 91 002604    000000   000000     MAXLBN::        .WORD    0,0          ; Highest LBN
 92 002610    000000              REPEAT::        .WORD    0            ; Flag endless loop
 93 002612    000000              ONLINE::        .WORD    0            ; Flags unit online
 94 002614    000000              BUSFLAG::       .WORD    0            ; Bus type flag
 95                                                                    ; 0=UNIBUS, 1=QBUS
 96 002616    000000              VSNPTR::        .WORD    0            ; Ptr to VSN text
 97 002620                        VSNBUF::        .BLKB    24.          ; VSN text
 98
 99
```

```
    1                          .SBTTL   Operator interface parameters
    2
    3 002650  000000           DEBUG:: .WORD   0                    ; Debug flag
    4 002652  000000           BACKUP::.WORD   0                    ; Backup flag
    5
    6 002654                   REPLY:: .BLKB   12                   ; Reply
    7
    8 002666                   TTYIN:: .BLKW   12                   ;TTY CHARACTER INPUT BUFFER
    9 002712  000000           CMDIN:: .WORD   0                    ;COMMAND CHARACTER INPUT BUFFER
   10 002714  000000           OPTIN:: .WORD   0                    ;OPTION CHARACTER INPUT BUFFER
   11
   12 002716                   LINBUF:         .BLKB   128.         ; Print buffer
   13
   14
```

```
  1                                   .SBTTL  SA register error parameters
  2
  3                                   ;       SA register error equates
  4
  5              000001               B.SADDR =       BIT0                    ; Print SA address
  6              000002               B.SAVAL =       BIT1                    ; Print SA contents
  7              000004               B.SAEXP =       BIT2                    ; Print SA expected value
  8
  9                                   ;       SA register msg table
 10
 11 003116                           SARMSG::
 12 003116      003165                       .WORD   XSADDR                  ; "SA address:    xxxxxx"
 13 003120      003136                       .WORD   XSAVAL                  ; "SA contents:   xxxxxx"
 14 003122      003214                       .WORD   XSAEXP                  ; "SA expected:   xxxxxx"
 15 003124      000000                       .WORD   0                       ; terminator
 16
 17                                   ;       SA register value table
 18
 19 003126                           SARVAL::
 20 003126      002314                       .WORD   SADDR                   ; SA address
 21 003130      002316                       .WORD   SAVAL                   ; Actual value
 22 003132      002320                       .WORD   SAEXP                   ; Expected value
 23 003134      000000                       .WORD   0                       ; terminator
 24
 25                                   ;       SA register error extended text
 26
 27                                           .NLIST  BEX
 28 003136      116    042    040     XSAVAL: .ASCIZ  /N"  SA contents:   "016/
 29 003165      116    042    040     XSADDR: .ASCIZ  /N"  SA address:    "016/
 30 003214      116    042    040     XSAEXP: .ASCIZ  /N"  SA expected:   "016/
 31                                           .LIST   BEX
 32                                           .EVEN
 33
```

```
     1                                  .SBTTL  Drive model parameters
     2
     3          000000                  M.UNKN          =       0
     4          000001                  M.RA80          =       1
     5          000004                  M.RA60          =       4
     6          000005                  M.RA81          =       5
     7          000013                  M.RA82          =       11.
     8
     9 003244   000000          MODEL:          .WORD   0               ; Unit model no.
    10 003246   000000          MDLNDX:         .WORD   0               ; Index to model table
    11
    12 003250                   MDLTBL:
    13 003250   000000                          .WORD   M.UNKN          ; "Unknown"
    14 003252   000001                          .WORD   M.RA80          ; "RA80"
    15 003254   000004                          .WORD   M.RA60          ; "RA60"
    16 003256   000005                          .WORD   M.RA81          ; "RA81"
    17 003260   000013                          .WORD   M.RA82          ; "RA82"
    18
    19          000012          MDLSIZ  =       .-MDLTBL
    20
    21 003262                   MDLTEXT:
    22 003262   003274                          .WORD   UNKNOWN                 ; 0  "Unknown"
    23 003264   003304                          .WORD   RA80                    ; 1  "RA80"
    24 003266   003311                          .WORD   RA60                    ; 4  "RA60"
    25 003270   003316                          .WORD   RA81                    ; 5  "RA81"
    26 003272   003323                          .WORD   RA82                    ; 11 "RA82"
    27
    28                                  .NLIST  BEX
    29 003274   125  116  113   UNKNOWN:        .ASCIZ          /UNKNOWN/
    30 003304   122  101  070   RA80:           .ASCIZ          /RA80/
    31 003311   122  101  066   RA60:           .ASCIZ          /RA60/
    32 003316   122  101  070   RA81:           .ASCIZ          /RA81/
    33 003323   122  101  070   RA82:           .ASCIZ          /RA82/
    34                                  .LIST   BEX
    35                                  .EVEN
    36
```

```
 1                              .SBTTL  MSCP error message parameters
 2
 3                      ;       MSCP error equates
 4
 5      000001          B.CMDRF =       BIT0                            ; Print cmd ref number
 6      000002          B.CODE  =       BIT1                            ; Print opcode
 7      000004          B.MOD   =       BIT2                            ; Print cmd modifiers
 8      000010          B.EMREF =       BIT3                            ; Print end msg ref
 9      000020          B.ECODE =       BIT4                            ; Print endcode
10      000040          B.FLAGS =       BIT5                            ; Print end flags
11      000100          B.ESTAT =       BIT6                            ; Print end msg status
12
13      000140          B.MSCP  =       B.FLAGS+B.ESTAT
14
15                      ;       MSCP extended msg table
16
17 003330              MSCPMSG::
18 003330  003370              .WORD   XCMDRF                          ; "Command ref:     xxxxxx"
19 003332  003421              .WORD   XCODE                           ; "MSCP command:       xxx"
20 003334  003454              .WORD   XMOD                            ; "Cmd modifiers:   xxxxxx"
21 003336  003506              .WORD   XENDRF                          ; "Endmsg ref:      xxxxxx"
22 003340  003537              .WORD   XECODE                          ; "MSCP endcode:       xxx"
23 003342  003572              .WORD   XFLAGS                          ; "Endmsg flags:       xxx"
24 0C3344  003625              .WORD   XESTAT                          ; "Endmsg status:   xxxxxx"
25 0C3346  000000              .WORD   0                               ; terminator
26
27                      ;       MSCP extended value table
28
29 003350              MSCPVAL::
30 003350  002342              .WORD   CMDREF                          ; Command reference number
31 003352  002344              .WORD   OPCODE                          ; MSCP command
32 003354  002346              .WORD   CMDMOD                          ; Command modifiers
33 003356  002350              .WORD   ENDREF                          ; End msg reference number
34 003360  002352              .WORD   ECODE                           ; End code
35 003362  002356              .WORD   FLAGS                           ; End msg flags
36 003364  002354              .WORD   EMSTAT                          ; End msg status
37 003366  000000              .WORD   0                               ; terminator
38
39                      ;       MSCP extended error text
40
41                              .NLIST  BEX
42 003370    116    042    040 XCMDRF: .ASCIZ  /N"  Command ref:    "016/
43 003421    116    042    040 XCODE:  .ASCIZ  /N"  MSCP command:      "08/
44 003454    116    042    040 XMOD:   .ASCIZ  /N"  Cmd modifiers:  "016N/
45 003506    116    042    040 XENDRF: .ASCIZ  /N"  Endmsg ref:     "016/
46 003537    116    042    040 XECODE: .ASCIZ  /N"  MSCP endcode:      "08/
47 003572    116    042    040 XFLAGS: .ASCIZ  /N"  Endmsg flags:      "08/
48 003625    116            XESTAT: .ASCII  /N/
49 003626    042    040    040 XSTAT:  .ASCIZ  /"  Endmsg status: "016N2/
50                              .LIST   BEX
51                              .EVEN
52
```

```
    1                               .SBTTL  RCT error message parameters
    2
    3                       ;       RCT error equates
    4
    5         000001        B.BLOCK =       BITO                    ; Print Block number
    6         000002        B.COPY  =       BIT1                    ; Print copy number
    7         000004        B.PHASE =       BIT2                    ;
    8
    9         000003        B.RCT   =       B.BLOCK!B.COPY          ; RCT error bits
   10
   11                       ;       RCT extended msg table
   12
   13 003660                       RCTMSG::
   14 003660 003700                        .WORD   XBLOCK          ; "RCT Block no:    ddd"
   15 003662 003732                        .WORD   XCOPY           ; "RCT copy:        ddd"
   16 003664 003764                        .WORD   XPHASE          ; "Replacement phase:  ddd"
   17 003666 000000                        .WORD   0               ; terminator
   18
   19                       ;       RCT extended value table
   20
   21 003670                       RCTVAL::
   22 003670 002476                        .WORD   RCTBLK          ; "Block no:  ddd"
   23 003672 002424                        .WORD   RCTCOPY         ; "RCT copy:  ddd"
   24 0C3674 002466                        .WORD   RECOVR          ; "Replacement phase:  ddd"
   25 003676 000000                        .WORD   0               ; terminator
   26
   27                               .NLIST  BEX
   28 003700    116     042     040 XBLOCK: .ASCIZ  /N"  RCT block no: "D16"."/
   29 003732    116     042     040 XCOPY:  .ASCIZ  /N"  RCT copy:      "D16"."/
   30 003764    116     042     040 XPHASE: .ASCIZ  /N"  Replacement phase: "D16/
   31                               .LIST   BEX
   32                               .EVEN
```

```
    1                              .SBTTL  Errorlog error message parameters
    2
    3                          ;       Errorlog messages and equates
    4
    5          000001         B.EVNT  =       BITO                    ; Print status/event
    6          000002         B.HADDR =       BIT1                    ; Print host address
    7          000004         B.DCODE =       BIT2                    ; Print drive error code
    8          000010         B.DERR  =       BIT3                    ; Print drive error info
    9
   10                          ;       Errorlog extended msg table
   11
   12 004020                 ERLMSG::
   13 004020  004034                  .WORD   XEVENT                  ; "Status/event: xxxxxx"
   14 004022  004063                  .WORD   XHADDR                  ; "Host address: xxxxxx"
   15 004024  000000                  .WORD   0                       ; terminator
   16
   17                          ;       Errorlog extended value table
   18
   19 004026                 ERLVAL::
   20 004026  002362                  .WORD   EVENT                   ; Event code
   21 004030  002400                  .WORD   HADDR                   ; Host address
   22 004032  000000                  .WORD   0                       ; terminator
   23
   24                          ;       Errorlog extended error text
   25
   26                                  .NLIST  BEX
   27 004034  116             XEVENT: .ASCII  /N/
   28 004035  042   040   040 XEVNT:  .ASCIZ  \"  Status/event: "016\
   29 004063  116   042   040 XHADDR: .ASCIZ  /N"  Host address: "032/
   30 004112  116   042   040 XDCODE: .ASCIZ  /N"  Drive error code (hex): "H8/
   31 004152  116   042   040 XDINFO: .ASCIZ  \N"  Drive status/error info (hex): "\
   32 004217  042   040   042 XDHEX:  .ASCIZ  /"  "H8/
   33                                  .LIST   BEX
   34                                  .EVEN
   35
```

```
 1
 2          000000              NDX.RD  =       0                   ; Index to READ
 3          000001              NDX.WR  =       1                   ; Index to WRITE
 4          000002              NDX.ACC =       2                   ; Index to ACCESS
 5          000003              NDX.RPL =       3                   ; Index to REPLACE
 6
 7 004226                       MSCPNDX:
 8 004226    000000                     .WORD   0                   ; Index to msg ptr
 9
10 004230                       MSCPTBL:
11 004230    004242                     .WORD   XREAD               ; "READ"
12 004232    004250                     .WORD   XWRITE              ; "WRITE"
13 0C4234    004257                     .WORD   XACCESS             ; "ACCESS"
14 004236    004267                     .WORD   XREPLACE            ; "REPLACE"
15 004240    000000                     .WORD   0                   ; Terminator
16
17                                      .NLIST  BEX
18 004242    122   105   101  XREAD:    .ASCIZ  /READ /
19 004250    127   122   111  XWRITE:   .ASCIZ  /WRITE /
20 004257    101   103   103  XACCESS:  .ASCIZ  /ACCESS /
21 004267    122   105   120  XREPLACE: .ASCIZ  /REPLACE /
22                                      .LIST   BEX
23                                      .EVEN
24
```

```
     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
    18                                  ;NAMES OF DEVICES SUPPORTED BY PROGRAM
    19                                  ;
    20 004300                          L$DVTYP::
       004300    114    157    147             .ASCIZ   %Logical disk drive%
                                              .EVEN
    21
    22
    23                                  ; TEST DESCRIPTION
    24                                  ;
    25 004324                          L$DESC::
       004324    102    141    144             .ASCIZ   /Bad Block Replacement Utility/
                                              .EVEN
    26
    27                                  ; FORMAT STATEMENTS USED IN PRINT CALLS
    28                                  ;
    29
```

```
    1                                    .SBTTL  Informational message text
    2
    3                            ;       Configuration information messages
    4
    5 004362      116      062      042  INF100: .ASCII  /N2"BAD BLOCK REPLACEMENT UTILITY."/
    6 004424      116      062      042          .ASCII  /N2"WARNING:  All drives configured for test"/
    7 004500      116      042      115          .ASCIZ  /N"MUST have customer data backed-up."/
    8
    9 004546      116      042      114  INF101: .ASCIZ  /N"Line clock not available."/
   10
   11 004603      116      042      111  INF102: .ASCII  /N"It is recommended that you read the operator help"/
   12 004667      116      042      151          .ASCII  /N"information before proceeding."/
   13
   14 004731      116      062      042  INF103: .ASCIZ  /N2"Restore customer data to drive after last pass."N/
   15
   16                            ;       Initialization information messages
   17
   18 005016      116      064      042  INF110: .ASCII  /N4"INITIALIZATION INFORMATION FOR UNIT "D8/
   19 005070      116      042      040          .ASCII  /N"  Controller:     "A/
   20 005113      116      042      040          .ASCII  /N"  Address:        "O16/
   21 005140      116      042      040          .ASCIZ  /N"  Drive:          "D8/
   22
   23 005165      116      042      040  INF111: .ASCII  /N"  Drive type:     "A/
   24 005210      116      042      040          .ASCII  /N"  Volume SN:      "A"."/
   25 005236      116      042      040          .ASCIZ  /N"  Volume size:  "D32"."N2/
   26
   27                            ;       End of pass information messages
   28
   29 005271      116      062      042  INF200: .ASCIZ  /N2"REPLACEMENT INFORMATION FOR THIS PASS"/
   30 005343      116      062      042  INF201: .ASCII  /N2"  BAD BLOCKs reported:          "D16"."/
   31 005414      116      042      040          .ASCII  /N"  Header compare errors:      "D16"."/
   32 005464      116      042      040          .ASCII  /N"  Data sync timeout errors:   "D16"."/
   33 005534      116      042      040          .ASCII  /N"  Uncorrectable ECC errors:   "D16"."/
   34 005604      116      042      040          .ASCII  /N"  FORCED ERROR flags detected: "D16"."/
   35 005654      116      042      040          .ASCIZ  /N"  FORCED ERROR flags written:  "D16"."/
   36 005725      116      062      042  INF202: .ASCII  /N2"  PRIMARY   replacements:      "D16"."/
   37 005776      116      042      040          .ASCII  /N"  SECONDARY replacements:      "D16"."/
   38 006046      116      042      040          .ASCII  /N"  RBN's marked UNUSABLE:       "D16"."/
   39 006116      116      042      040          .ASCIZ  /N"  Total blocks replaced:       "D16"."N/
   40
   41                            ;       RCT descriptor information messages
   42
   43 006170      116      062      042  INF210: .ASCIZ  /N2"RCT DESCRIPTOR INFORMATION FOR UNIT "D8N/
   44 006244      116      042      122  INF211: .ASCIZ  /N"RBN:"D28". is PRIMARY   replacement for LBN:"D28"."/
   45 006332      116      042      122  INF212: .ASCIZ  /N"RBN:"D28". is SECONDARY replacement for LBN:"D28"."/
   46 006420      116      042      122  INF213: .ASCIZ  /N"RBN "D28". is UNUSABLE"/
   47
   48 006452      116      062      042  INF220: .ASCIZ  /N2"RCT DESCRIPTOR SUMMARY FOR THIS UNIT"/
   49 006523      116      042      040  INF221: .ASCII  /N"  PRIMARY   replacement blocks: "D16"."/
   50 006574      116      042      040          .ASCII  /N"  SECONDARY replacement blocks: "D16"."/
   51 006645      116      042      040          .ASCII  /N"  UNUSABLE  replacement blocks: "D16"."/
   52 006716      116      042      040          .ASCIZ  /N"  "D16". RBNs used out of "D16". RBNs on media."N/
   53
```

```
     1
     2                                    ;           Automatic BBR information
     3
     4 007002       116       062       042   INF301: .ASCIZ    /N2"SCANNING RCT..."N/
     5 007027       116       062       042   INF302: .ASCIZ    /N2"SCANNING LBN AREA..."N/
     6 007061       116       062       042   INF310: .ASCIZ    /N2"MANUAL BAD BLOCK REPLACEMENT SELECTED."N/
     7 007135       116       062       042   INF311: .ASCIZ    /N2"  EOF detected.    "/
     8 007164       116       062       042   INF320: .ASCIZ    /N2"AUTOMATIC BAD BLOCK REPLACEMENT SELECTED."/
     9 007242       116       062       042   INF321: .ASCII    /N2"  LBN Status codes:"/
    10 007271       116       042       040           .ASCII    /N"  BBR - Bad Block reported."/
    11 007327       116       042       040           .ASCII    /N"  ECH - Hard ECC Error encountered."/
    12 007375       116       042       040           .ASCII    /N"  HCE - Header Compare Error encountered."/
    13 007451       116       042       040           .ASCII    /N"  DST - Data Sync Timeout encountered."/
    14 007522       116       042       040           .ASCII    /N"  FER - Forced Error encountered."/
    15 007566       116       042       040           .ASCIZ    /N"  WFE - Block written with FE."/
    16 007630       116       062       042   INF322: .ASCII    /N2"  RBN Operation types:"/
    17 007662       116       042       040           .ASCII    /N"  PRI - Primary replacement for LBN."/
    18 007731       116       042       040           .ASCII    /N"  SEC - Secondary replacement for LBN."/
    19 010002       116       042       040           .ASCII    /N"  UNS - RBN marked Unusable."N/
    20 010043       116       042       040   INF323: .ASCIZ    /N"  LBN:"D28".    "/
    21 010065       116       062       042   INF324: .ASCIZ    /N2"  EOM reached.      "/
    22
    23                                    ;           Replacement information messages
    24
    25 010114       116       042       040   INF401: .ASCIZ    /N"  RCT copy "D16", block "D16" ("D28".)"/
    26
    27 010166       116       042       040   INF405: .ASCIZ    /N"  LBN:"D28"."/
    28 010206       042       040       040   INF406: .ASCIZ    /"  Status: "A/
    29 010224       042       040       040   INF407: .ASCIZ    /"  RBN:"D28"."/
    30 010243       042       040       040   INF408: .ASCIZ    /"  Operation: "A/
    31
    32                                    ;           RCT information messages
    33
    34 010264       116       062       042   INF601: .ASCIZ    /N2"Attempting Recovery..."N/
    35 010320       116       062       042   INF602: .ASCIZ    /N2"RCT Recovery failed; MEDIA INTEGRITY UNCERTAIN."N/
    36 010405       116       062       042   INF603: .ASCIZ    /N2"Attempting to find another replacement block..."N/
    37
```

```
    1                                        .SBTTL   Error messege text
    2
    3
    4                                   ;          Program initialization messages
    5
    6 010472      120      122      117  HDR001: .ASCIZ   /PROGRAM INITIALIZATION ERROR/
    7 010527      116      042      106  ERM001: .ASCII   /N"Failed dynamic storage allocation for Unit "D3";"/
    8 010612      116      042      040          .ASCIZ   /N"  Restart program and select fewer Units."/
    9
   10                                   ;          Controller initialization messages
   11
   12 010667      103      117      116  HDR101: .ASCIZ   /CONTROLLER INITIALIZATION ERROR/
   13 010727      116      042      101  ERM101: .ASCII   /N"Address error occured while accessing contoller register;"/
   14 011023      116      042      040          .ASCIZ   /N"  Check contoller address."/
   15 011061      116      042      105  ERM102: .ASCIZ   /N"Error detected during controller initialization;"/
   16 011145      116      042      103  ERM103: .ASCIZ   /N"Controller did not clear ring structure in host memory;"/
   17 011240      116      042      123  ERM104: .ASCIZ   /N"SA register not zero after step 3 write;"/
   18
   19                                   ;          Unit initialization errors
   20
   21 011314      125      116      111  HDR201: .ASCIZ   /UNIT INITIALIZATION ERROR/
   22 011346      116      042      106  ERM201: .ASCII   /N"Failed GET UNIT STATUS command;"/
   23 011410      116      042      040          .ASCIZ   /N"  Drive "D8" is not accessable."/
   24 011453      116      042      106  ERM202: .ASCII   /N"Failed SET CONTROLLER CHARACTERISTICS command;"/
   25 011534      116      042      040          .ASCII   /N"  Host access timeout not disabled."/
   26 011602      116      042      040          .ASCII   /N"  Default timeout is 60 seconds."/
   27 011645      116      042      040          .ASCII   /N"  Error log packets not enabled."/
   28 011711      116      042      106  ERM203: .ASCII   /N"Failed ONLINE command;"/
   29 011743      116      042      132  ERM204: .ASCII   /N"Zero RCT size detected;"/
   30 011775      116      042      040          .ASCIZ   /N"  BBR not supported by this drive."/
   31
   32                                   ;          MSCP cmd/msg error messages
   33
   34 012043      115      123      103  HDR301: .ASCIZ   /MSCP STATUS ERROR DETECTED/
   35 012076      116      042      106  ERM300: .ASCII   /N"Failed "A"command; LBN:"D28"."/
   36 012137      116      042      106  ERM301: .ASCII   /N"Failed READ command; LBN:"D28"."/
   37 012202      116      042      106  ERM302: .ASCII   /N"Failed WRITE command; LBN:"D28"."/
   38 012246      116      042      106  ERM303: .ASCII   /N"Failed ACCESS command; LBN:"D28"."/
   39 012313      116      042      106  ERM304: .ASCII   /N"Failed REPLACE command; LBN:"D28"."/
   40 012361      116      042      106  ERM305: .ASCII   /N"Failed replacement verification; LBN:"D28"."/
   41 012437      116      042      040          .ASCIZ   /N"  Retry of replacement operation will be attempted."/
   42 012526      116      042      103  ERM306: .ASCIZ   \N"Command/message reference number mismatch; LBN:"D28"."\
   43 012617      116      042      125  ERM307: .ASCIZ   /N"Unrecognized endcode; LBN:"D28"."/
   44
```

```
 1
 2                                    ;        Controller communication errormessages
 3
 4 012663      110      117      123 HDR401: .ASCIZ  \HOST/CONTROLLER COMMUNICATION ERROR\
 5 012727      116      042      116 ERM401: .ASCIZ  /N"No interrupt received for 30 seconds;"/
 6 013000      116      042      106 ERM402: .ASCIZ  /N"Fatal error reported in SA register;"/
 7 013050      116      042      123 ERM403: .ASCIZ  /N"Serious exception reported;"/
 8
 9                                    ;        Program errors
10
11 013107      120      162      157 HDR501: .ASCIZ  /Program error/
12 013125      116      042      115 ERM501: .ASCIZ  /N"Media overrun detected; LBN:"D28"."/
13
14                                    ;        RCT read/write errors
15
16 013173      122      103      124 HDR601: .ASCIZ  /RCT ERROR DETECTED/
17 013216      116      042      106 ERM601: .ASCIZ  /N"Failed RCT READ;"/
18 013242      116      042      106 ERM602: .ASCIZ  /N"Failed RCT WRITE;"/
19 013267      116      042      106 ERM603: .ASCIZ  /N"Failed READ of all copies of RCT;"/
20 013334      116      042      106 ERM604: .ASCIZ  /N"Failed WRITE to all copies of RCT;"/
21 013402      116      042      103 ERM605: .ASCIZ  /N"Crash occured during previous replacement operation;"/
22 013472      116      042      116 ERM606: .ASCII  /N"No NULL descriptor entry found in RCT;"/
23 013543      116      042      040         .ASCIZ  /N"  RCT is probably corrupt; REFORMAT MEDIA."/
24 013621      116      042      124 ERM607: .ASCII  /N"Transient error table overflow;"/
25 013663      116      042      040         .ASCIZ  /N"  ADDITIONAL MAINTENANCE REQUIRED."/
26
27                                    ;        Error log messages
28
29 013731      105      122      122 HDR701: .ASCIZ  /ERROR LOG PACKET RECEIVED/
30 013763      116      042      125 ERM700: .ASCIZ  /N"Unrecognized Errorlog format;"/
31 014024      116      042      103 ERM701: .ASCIZ  /N"Controller Error reported;"/
32 014062      116      042      110 ERM702: .ASCIZ  /N"Host Memory Access Error reported;"/
33 014130      116      042      123 ERM703: .ASCIZ  /N"SDI Error reported;"/
34 014157      116      042      123 ERM704: .ASCIZ  /N"Small Disk Error reported;"/
35
```

```
     1                              .SBTTL  Formatted text
     2
     3                          ;       Information text
     4
     5 014215      045    116    000  F.CRLF: .ASCIZ  /%N/                        ; FAO New line
     6 014220      045    124    000  F.TEXT: .ASCIZ  /%T/                        ; FAO Text
     7
     8 014223      101    000         FMT.ASC:.ASCIZ  /A/                         ; FAO .ASCIZ
     9
    10 014225      125    104    101  DEVNAME:.ASCIZ  /UDA/                       ; Controller name
    11 014231      116    000         CRLF:   .ASCIZ  /N/
    12 014233      042    040    040  RNTIM:  .ASCIZ  /"  Elapsed runtime "D16":"/
    13 014266      104    071    042  RNTIM1: .ASCIZ  /D9":"/
    14 014274      104    071    000  RNTIM2: .ASCIZ  /D9/
    15 014277      042    040    077  LMT28:  .ASCIZ  /" ?Exceeded 28 bit input limit"N/
    16 014340      042    040    077  INVCHR: .ASCIZ  /" ?Invalid character"N/
    17 014367      116    042    114  OLIMIT: .ASCIZ  /N"Limits:  LO= 0;  HI= "D32"."N/
    18 014427      116    042    105  ERRME1: .ASCIZ  /N"Error processing message string"N/
    19 014473      116    042    115  MANDIS: .ASCIZ  /N"Manual intervention is disabled"N/
    20 014537      116    042    040  XRCT:   .ASCIZ  /N"  RCT block number: "D16."N/
    21 014575      116    042    040  XLBN:   .ASCIZ  /N"  LBN:"D28"."/
    22 014615      116    042    040  XRBN:   .ASCIZ  /N"  RBN:"D28"."/
    23 014635      116    042    040  XSTEP:  .ASCIZ  /N"  Replacement STEP: "D8"."/
    24                              .EVEN
    25
    26                          ; Unformatted text
    27
    28 014672      132    125    104  HLPFILE:.ASCIZ  /ZUDLAO.HLP/
    29
    30 014705      105    156    164  TXT000: .ASCIZ  /Enter LBN to be replaced (D) or <CR> to exit/
    31 014762      122    102    116  TXT001: .ASCIZ  /RBN /
    32 014767      114    102    116  TXT002: .ASCIZ  /LBN /
    33 014774      105    156    164  TXT005: .ASCIZ  /Enter starting group LBN (D)/
    34 015031      101    164    164  TXT006: .ASCIZ  /Attempt crash recovery/
    35
    36 015060      105    156    164  TX.LB1: .ASCIZ  /Enter first LBN/
    37 015100      105    156    164  TX.LB2: .ASCIZ  /Enter last LBN/
    38
    39 015117      120    122    111  TX.PRI:         .ASCIZ  /PRI /
    40 015124      123    105    103  TX.SEC:         .ASCIZ  /SEC /
    41 015131      125    116    123  TX.UNS:         .ASCIZ  /UNS /
    42 015136      106    105    122  TX.FER:         .ASCIZ  /FER /
    43 015143      105    103    110  TX.ECH:         .ASCIZ  /ECH /
    44 015150      102    102    122  TX.BBR:         .ASCIZ  /BBR /
    45 015155      102    102    125  TX.BBU:         .ASCIZ  /BBU /
    46 015162      123    105    130  TX.SEX:         .ASCIZ  /SEX /
    47 015167      114    117    107  TX.LOG:         .ASCIZ  /LOG /
    48 015174      040    040    040  TX.BLK:         .ASCIZ  /    /
    49 015201      127    106    105  TX.WFE:         .ASCIZ  /WFE /
    50 015206      110    103    105  TX.HCE:         .ASCIZ  /HCE /
    51 015213      104    123    124  TX.DST:         .ASCIZ  /DST /
    52                                      .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                                ;--
     8
```

```
 1                                      .SBTTL  Error processing routines
 2
 3                                      ;ERR001:  General case error routine
 4                                      ;
 5                                      ;        Prints basic error msg from BASEMSG with Drive number
 6
 7 015220                              ERR001::
 8
 9 015220  010146                              MOV     R1,-(SP)                ;;PUSH R1 ON STACK
10
11 015222  013701   002304                     MOV     CTLRTBL,R1              ; Load controller table ptr
12 015226  016546   000006                     MOV     C.DRVN(R5),-(SP)              ;PUSH C.DRVN(R5) ON STACK
   015232  013746   002552                     MOV     BASEMSG,-(SP)           ;PUSH BASEMSG ON STACK
   015236  004137   033254                     JSR     R1,LPNTB                     ;CALL LPNTB PRINT ROUTINE
   015242  000004                              .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
13 015244  013746   002354                     MOV     EMSTAT,-(SP)            ;PUSH EMSTAT ON STACK
   015250  012746   003625                     MOV     #XESTAT,-(SP)           ;PUSH #XESTAT ON STACK
   015254  004137   033254                     JSR     R1,LPNTB                     ;CALL LPNTB PRINT ROUTINE
   015260  000004                              .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
14
15 015262  012601                              MOV     (SP)+,R1                ;;POP STACK INTO R1
16 015264                              L10002:
   015264  104423                              TRAP    C$MSG
17
18                                      ;ERR101:  Controller initialization error routine
19                                      ;
20                                      ;        Prints Basic msg in BASEMSG
21                                      ;        Prints extended SA register info
22
23 015266                              ERR101::
24
25 015266  010146                              MOV     R1,-(SP)                ;;PUSH R1 ON STACK
   015270  010246                              MOV     R2,-(SP)                ;;PUSH R2 ON STACK
   015272  010346                              MOV     R3,-(SP)                ;;PUSH R3 ON STACK
26
27 015274  013746   002552                     MOV     BASEMSG,-(SP)           ;PUSH BASEMSG ON STACK
   015300  004137   033254                     JSR     R1,LPNTB                     ;CALL LPNTB PRINT ROUTINE
   015304  000002                              .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
28
29 015306  012702   003116                     MOV     #SARMSG,R2              ; Load text table ptr
30 015312  012703   003126                     MOV     #SARVAL,R3              ; Load value table ptr
31 015316  004737   016042                     JSR     PC,CALR1                ; Print extended info
32
33 015322  012603                              MOV     (SP)+,R3                ;;POP STACK INTO R3
   015324  012602                              MOV     (SP)+,R2                ;;POP STACK INTO R2
   015326  012601                              MOV     (SP)+,R1                ;;POP STACK INTO R1
34
35 015330  000167                              .WORD   J$JMP
   015332  000000                              .WORD   L10003-2-.
36
37 015334                              L10003:
   015334  104423                              TRAP    C$MSG
38
```

```
  1                              ;ERR201:  Unit initialization error routine
  2                              ;
  3                              ;         Prints basic msg in BASEMSG
  4                              ;         Prints extended end message info
  5
  6
  7 015336                       ERR201::
  8
  9 015336   010146                      MOV     R1,-(SP)                ;;PUSH R1 ON STACK
    015340   010246                      MOV     R2,-(SP)                ;;PUSH R2 ON STACK
    015342   010346                      MOV     R3,-(SP)                ;;PUSH R3 ON STACK
 10
 11 015344   013746   002552             MOV     BASEMSG,-(SP)           ;PUSH BASEMSG ON STACK
    015350   004137   033254             JSR     R1,LPNTB                      ;CALL LPNTB PRINT ROUTINE
    015354   000002                      .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
 12
 13 015356   012702   003330             MOV     #MSCPMSG,R2             ; Load text table ptr
 14 015362   012703   003350             MOV     #MSCPVAL,R3             ; Load value table ptr
 15 015366   004737   016042             JSR     PC,CALR1               ; Print MSCP info
 16
 17 015372   012603                      MOV     (SP)+,R3                ;;POP STACK INTO R3
    015374   012602                      MOV     (SP)+,R2                ;;POP STACK INTO R2
    015376   012601                      MOV     (SP)+,R1                ;;POP STACK INTO R1
 18
 19 015400   000167                      .WORD   J$JMP
    015402   000000                      .WORD   L10004-2-.
 20
 21 015404                       L10004:
    015404   104423                       TRAP    C$MSG
 22
 23                              ;ERR300:  General MSCP error routine
 24                              ;
 25                              ;         Prints ERM300 with MSCP operation text ptr in MSCPNDX and LBN
 26                              ;         Prints extended end message info
 27
 28 015406                       ERR300::
 29
 30 015406   010146                      MOV     R1,-(SP)                ;;PUSH R1 ON STACK
    015410   010246                      MOV     R2,-(SP)                ;;PUSH R2 ON STACK
    015412   010346                      MOV     R3,-(SP)                ;;PUSH R3 ON STACK
 31
 32 015414   013701   004226             MOV     MSCPNDX,R1             ; Load index
 33 015420   042701   177774             BIC     #↑C3,R1                ; Clear illegal values
 34 015424   006301                      ASL     R1                     ; Shift left
 35
 36 015426   013746   002450             MOV     LBN+2,-(SP)             ;PUSH LBN+2 ON STACK
    015432   013746   002446             MOV     LBN,-(SP)               ;PUSH LBN ON STACK
    015436   016146   004230             MOV     MSCPTBL(R1),-(SP)             ;PUSH MSCPTBL(R1) ON STACK
    015442   012746   012076             MOV     #ERM300,-(SP)          ;PUSH #ERM300 ON STACK
    015446   004137   033254             JSR     R1,LPNTB                      ;CALL LPNTB PRINT ROUTINE
    015452   000010                      .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
 37
 38 015454   012702   003330             MOV     #MSCPMSG,R2            ; Load text table ptr
 39 015460   012703   003350             MOV     #MSCPVAL,R3            ; Load value table ptr
 40 015464   004737   016042             JSR     PC,CALR1               ; Print MSCP info
 41
 42 015470   012603                      MOV     (SP)+,R3                ;;POP STACK INTO R3
```

```
        015472  012602                      MOV     (SP)+,R2              ;;POP STACK INTO R2
        015474  012601                      MOV     (SP)+,R1              ;;POP STACK INTO R1
43
44  015476  000167                          .WORD   J$JMP
        015500  000000                       .WORD   L10005-2-.
45
46  015502                          L10005:
        015502  104423                      TRAP    C$MSG
47
48                                  ;ERR301:  MSCP error routine
49                                  ;
50                                  ;         Prints basic msg in BASEMSG with LBN in command packet
51                                  ;         Prints extended end message info
52
53  015504                          ERR301::
54
55  015504  010146                          MOV     R1,-(SP)              ;;PUSH R1 ON STACK
        015506  010246                      MOV     R2,-(SP)              ;;PUSH R2 ON STACK
        015510  010346                      MOV     R3,-(SP)              ;;PUSH R3 ON STACK
56
57  015512  016401  000140                  MOV     HC.CPK+P.LBN(R4),R1   ; Unload LBN (low)
58  015516  016402  000142                  MOV     HC.CPK+P.LBN+2(R4),R2 ; Unload LBN (high)
59  015522  010246                          MOV     R2,-(SP)              ;PUSH R2 ON STACK
        015524  010146                      MOV     R1,-(SP)              ;PUSH R1 ON STACK
        015526  013746  002552              MOV     BASEMSG,-(SP)         ;PUSH BASEMSG ON STACK
        015532  004137  033254              JSR     R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
        015536  000006                      .WORD   ARG.CT                ;ARGUMENT COUNT * 2
60
61  015540  012702  003330                  MOV     #MSCPMSG,R2           ; Load text table ptr
62  015544  012703  003350                  MOV     #MSCPVAL,R3           ; Load value table ptr
63  015550  004737  016042                  JSR     PC,CALR1              ; Print MSCP info
64
65  015554  012603                          MOV     (SP)+,R3              ;;POP STACK INTO R3
        015556  012602                      MOV     (SP)+,R2              ;;POP STACK INTO R2
        015560  012601                      MOV     (SP)+,R1              ;;POP STACK INTO R1
66
67  015562  000167                          .WORD   J$JMP
        015564  000000                       .WORD   L10006-2-.
68
69  015566                          L10006:
        015566  104423                      TRAP    C$MSG
70
71                                  ;ERR302:  MSCP error routine
72                                  ;
73                                  ;         Prints basic msg in BASEMSG with LBN for ACCESS error
74                                  ;         Prints extended end message info
75
76  015570                          ERR302::
77
78  015570  010146                          MOV     R1,-(SP)              ;;PUSH R1 ON STACK
        015572  010246                      MOV     R2,-(SP)              ;;PUSH R2 ON STACK
        015574  010346                      MOV     R3,-(SP)              ;;PUSH R3 ON STACK
79
80  015576  013746  002450                  MOV     LBN+2,-(SP)           ;PUSH LBN+2 ON STACK
        015602  013746  002446              MOV     LBN,-(SP)             ;PUSH LBN ON STACK
        015606  013746  002552              MOV     BASEMSG,-(SP)         ;PUSH BASEMSG ON STACK
        015612  004137  033254              JSR     R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
```

```
      015616  000006                          .WORD   ARG.CT              ;ARGUMENT COUNT * 2
   81
   82 015620  012702  003330                  MOV     #MSCPMSG,R2         ; Load text table ptr
   83 015624  012703  003350                  MOV     #MSCPVAL,R3         ; Load value table ptr
   84 015630  004737  016042                  JSR     PC,CALR1            ; Print MSCP info
   85
   86 015634  012603                          MOV     (SP)+,R3            ;;POP STACK INTO R3
      015636  012602                          MOV     (SP)+,R2            ;;POP STACK INTO R2
      015640  012601                          MOV     (SP)+,R1            ;;POP STACK INTO R1
   87
   88 015642  000167                          .WORD   J$JMP
      015644  000000                          .WORD   L10007-2-.
   89
   90 015646                          L10007:
      015646  104423                          TRAP    C$MSG
   91
```

```
    1                                ;ERR401:  Host/controller communication error routine
    2                                ;
    3                                ;         Prints basic msg in BASEMSG
    4                                ;         Prints extended end message info
    5
    6
    7 015650                         ERR401::
    8
    9 015650  010146                         MOV     R1,-(SP)               ;;PUSH R1 ON STACK
      015652  010246                         MOV     R2,-(SP)               ;;PUSH R2 ON STACK
      015654  010346                         MOV     R3,-(SP)               ;;PUSH R3 ON STACK
   10
   11 015656  013746  002552                 MOV     BASEMSG,-(SP)          ;PUSH BASEMSG ON STACK
      015662  004137  033254                 JSR     R1,LPNTB                         ;CALL LPNTB PRINT ROUTINE
      015666  000002                         .WORD   ARG.CT                 ;ARGUMENT COUNT * 2
   12
   13 015670  012702  003116                 MOV     #SARMSG,R2             ; Load text table ptr
   14 015674  012703  003126                 MOV     #SARVAL,R3             ; Load value table ptr
   15 015700  004737  016042                 JSR     PC,CALR1               ; Print MSCP info
   16
   17 015704  012603                         MOV     (SP)+,R3               ;;POP STACK INTO R3
      015706  012602                         MOV     (SP)+,R2               ;;POP STACK INTO R2
      015710  012601                         MOV     (SP)+,R1               ;;POP STACK INTO R1
   18
   19 015712  000167                         .WORD   J$JMP
      015714  000000                         .WORD   L10010-2-.
   20
   21 015716                         L10010:
      015716  104423                         TRAP    C$MSG
   22
   23                                ;ERR601:  RCT error routine
   24                                ;
   25                                ;         Prints basic msg in BASEMSG
   26                                ;         Prints extended RCT info
   27
   28 015720                         ERR601::
   29
   30 015720  010146                         MOV     R1,-(SP)               ;;PUSH R1 ON STACK
      015722  010246                         MOV     R2,-(SP)               ;;PUSH R2 ON STACK
      015724  010346                         MOV     R3,-(SP)               ;;PUSH R3 ON STACK
   31
   32 015726  013746  002552                 MOV     BASEMSG,-(SP)          ;PUSH BASEMSG ON STACK
      015732  004137  033254                 JSR     R1,LPNTB                         ;CALL LPNTB PRINT ROUTINE
      015736  000002                         .WORD   ARG.CT                 ;ARGUMENT COUNT * 2
   33
   34 015740  012702  003660                 MOV     #RCTMSG,R2             ; Load text table ptr
   35 015744  012703  003670                 MOV     #RCTVAL,R3             ; Load value table ptr
   36 015750  004737  016042                 JSR     PC,CALR1               ; Print RCT info
   37
   38 015754  012603                         MOV     (SP)+,R3               ;;POP STACK INTO R3
      015756  012602                         MOV     (SP)+,R2               ;;POP STACK INTO R2
      015760  012601                         MOV     (SP)+,R1               ;;POP STACK INTO R1
   39
   40 015762  000167                         .WORD   J$JMP
      015764  000000                         .WORD   L10011-2-.
   41
   42 015766                         L10011:
```

```
      015766  104423                          TRAP    C$MSG
   43
```

```
     1
     2                              ;ERR701:  Error Log packet error routine
     3                              ;
     4                              ;         Prints basic msg in BASEMSG
     5                              ;         Prints extended error log packet info
     6
     7 015770                      ERR701::
     8
     9 015770  010146                      MOV     R1,-(SP)              ;;PUSH R1 ON STACK
       015772  010246                      MOV     R2,-(SP)              ;;PUSH R2 ON STACK
       015774  010346                      MOV     R3,-(SP)              ;;PUSH R3 ON STACK
    10
    11 015776  013746  002552              MOV     BASEMSG,-(SP)         ;PUSH BASEMSG ON STACK
       016002  004137  033254              JSR     R1,LPNTB              ;CALL LPNTB PRINT ROUTINE
       016006  000002                      .WORD   ARG.CT                ;ARGUMENT COUNT * 2
    12 016010  013746  002362              MOV     EVENT,-(SP)           ;PUSH EVENT ON STACK
       016014  012746  004034              MOV     #XEVENT,-(SP)         ;PUSH #XEVENT ON STACK
       016020  004137  033254              JSR     R1,LPNTB                   ;CALL LPNTB PRINT ROUTINE
       016024  000004                      .WORD   ARG.CT                ;ARGUMENT COUNT * 2
    13
    14 016026  012603                      MOV     (SP)+,R3              ;;POP STACK INTO R3
       016030  012602                      MOV     (SP)+,R2              ;;POP STACK INTO R2
       016032  012601                      MOV     (SP)+,R1              ;;POP STACK INTO R1
    15
    16 016034  000167                      .WORD   J$JMP
       016036  000000                      .WORD   L10012-2-.
    17
    18 016040                      L10012:
       016040  104423                      TRAP    C$MSG
    19
```

```
    1                                    .SBTTL   Pre-programmed subroutines
    2
    3                                 ;CALR1: Pre-programmed routine 1
    4                                 ;       Prints extended error info
    5                                 ;
    6                                 ;Inputs:
    7                                 ;       EXTMSK   Extended message mask
    8                                 ;       R2       Pointer to ASCIZ msg address table
    9                                 ;       R3       Pointer to value address table
   10
   11 016042                         CALR1:
   12
   13 016042                         20$:
   14 016042  032737 000001 002370          BIT      #1,EXTMSK            ; Test if bit set
   15 016050  001406                        BEQ      30$                  ; B-anch if NO
   16
   17 016052  017346 000000                 MOV      @(R3),-(SP)          ;PUSH @(R3) ON STACK
      016056  011246                        MOV      (R2),-(SP)           ;PUSH (R2) ON STACK
      016060  004137 033264                 JSR      R1,LPNTX                    ;CALL LPNTX PRINT ROUTINE
      016064  000004                        .WORD    ARG.CT               ;ARGUMENT COUNT * 2
   18 016066                         30$:
   19 016066  006237 002370                 ASR      EXTMSK               ; Shift to next position
   20 016072  062703 000002                 ADD      #2,R3                ; Bump to next value
   21 016076  062702 000002                 ADD      #2,R2                ; Bump to next msg
   22 016102  005712                        TST      (R2)                 ; Test for null msg
   23 016104  001356                        BNE      20$                  ; Branch if NO
   24 016106                       - 40$:
   25 016106  012746 014231                 MOV      #CRLF,-(SP)          ;PUSH #CRLF ON STACK
      016112  004137 033254                 JSR      R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
      016116  000002                        .WORD    ARG.CT               ;ARGUMENT COUNT * 2
   26 016120  000207                        RTS      PC                   ; Return
   27
   28                                 ;Pre-programmed routine 2
   29                                 ;
   30                                 ;Not used
   31
   32 016122                         CALR2:
   33 016122  000207                        RTS      PC                   ; Return
   34
   35                                 ;Pre-programmed routine 3
   36                                 ;
   37                                 ;Not used
   38
   39 016124                         CALR3:
   40 016124  000207                        RTS      PC                   ; Return
   41
```

```
  1                                  ;Pre-programmed routine 4
  2                                  ;
  3                                  ;Not used
  4
  5 016126                  CALR4:
  6 016126   000207              RTS     PC                        ; Return
  7
  8                                  ;Pre-programmed routine 5
  9                                  ;
 10                                  ;Not used
 11
 12 016130                  CALR5:
 13 016130   000207              RTS     PC                        ; Return
 14
 15                                  ;Pre-programmed routine 6
 16                                  ;
 17                                  ;Not used
 18
 19 016132                  CALR6:
 20 016132   000207              RTS     PC                        ; Return
 21
 22
 23                          .SBTTL  Error vector table
 24
 25                          ;       Pre-programmed error routine dispatch table
 26
 27 016134   016042         ERR.TB: .WORD   CALR1                 ; Print SA address
 28 016136   016122                 .WORD   CALR2                 ; Print SA value
 29 016140   016124                 .WORD   CALR3                 ; Print SA expected value
 30 016142   016126                 .WORD   CALR4                 ; Print MSCP parms
 31 016144   016130                 .WORD   CALR5                 ; Print LBN, RBN
 32 016146   016132                 .WORD   CALR6                 ; Not used
 33 016150   016132                 .WORD   CALR6                 ; Not used
 34 016152   016132                 .WORD   CALR6                 ; Not used
 35
 36          000010                 ERR.SZ  = <.-ERR.TB>/2
 37
 38                          .SBTTL  Interrupt service routines
 39                                  ;
 40                                  ;
 41                                  ;
 42                                  ;
 43                                  ;
 44                                  ;
 45
```

```
    1                                 .SBTTL  NXMI    Nonexistant memory routine
    2                                 ;NXMI
    3                                 ;
    4                                 ;Nonexistant memory service routine
    5                                 ;
    6                                 ;Inputs:
    7                                 ;       NXMAD set to zero
    8                                 ;Outputs:
    9                                 ;       NXMAD set to ones if nonexistant trap occured.
   10
   11 016154                         NXMI::
   12 016154  012737  177777  002224         MOV     #-1,NXMAD
   13 016162                         L10013:
      016162  000002                         RTI
   14
```

```
 1                              .SBTTL  xDASRV   Interrupt service routine
 2                              ;xDASRV
 3                              ;
 4                              ;xDA interrupt service routine.  Marks xDA controller table that an
 5                              ;interrupt has been received.
 6                              ;
 7                              ;This routine is called by a [JSR R0,xDASRV] instruction from within
 8                              ;the controller table.  The PC stored in R0 is the address of the C.FLG
 9                              ;word in the controller table.  The STACK contains the saved contents
10                              ;of R0 followed by the interrupted PC and PS.
11                              ;
12                              ;Inputs:
13                              ;     R0      Address of C.FLG word in controller table
14                              ;     STACK   Saved contents of R0
15                              ;
16                              ;Outputs:
17                              ;     CT.CMD  Cleared and CT.MSG set in C.FLG word of controller table
18                              ;     R0      Restored from STACK
19                              xDASRV::
20 016164
21 016164  052710  000010              BIS     #CT.MSG,(R0)            ;SET CT.MSG
22 016170  012600                      MOV     (SP)+,R0               ;;POP STACK INTO R0
23 016172                      L10014:
   016172  000002                      RTI
24
```

```
    1                                   .SBTTL  KW11I    Clock interrupt service routine
    2                                   ;KW11I
    3                                   ;
    4                                   ;Clock interrupt service routine
    5
    6 016174                            KW11I::
    7 016174   062737  000001  002220           ADD     #1,KW.EL                    ;COUNT THE INTERRUPT
    8 016202   005537  002222                   ADC     KW.EL+2
    9 016206   012777  000105  163774           MOV     #KW.OUT,@KW.CSR             ;RESTART THE CLOCK
   10 016214                            L10015:
      016214   000002                           RTI
   11 016216                            INTSRV::
   12 016216   005237  002332                   INC     INTRCV                      ; FLAG INTERRUPT AS RECEIVED
   13 016222                            L10016:
      016222   000002                           RTI
   14
```

```
     1                              .SBTTL  RESET    Reset xDA
     2                              ;RESET
     3                              ;
     4                              ;Reset all xDA-50S in the controller tables
     5                              ;
     6                              ;Inputs:
     7                              ;     CTLRLST Contains the address of the 1st controller table
     8                              ;
     9                              ;Outputs:
    10                              ;     NONE
    11                              ;
    12
    13 016224                      RESET:
    14 016224  010246                      MOV     R2,-(SP)              ;;PUSH R2 ON STACK
       016226  010446                      MOV     R4,-(SP)              ;;PUSH R4 ON STACK
    15 016230  005037 002224               CLR     NXMAD                 ; CLEAR NON-EXISTANT MEMORY ADDRESS
    16 016234  012737 177776 002300        MOV     #-2,TBLNDX            ; Reset controller table index
    17
    18 016242  012746 000340               MOV     #PRIO7,-(SP)
       016246  012746 016154               MOV     #NXMI,-(SP)
       016252  012746 000004               MOV     #4,-(SP)
       016256  012746 000003               MOV     #3,-(SP)
       016262  104437                      TRAP    C$SVEC
       016264  062706 000010               ADD     #10,SP
    19                                                                   ;>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<
    20 016270  104422                      TRAP    C$BRK
    21 016272                      10$:
    22 016272  062737 000002 002300        ADD     #2,TBLNDX             ; Bump table index
    23 016300  013702 002300               MOV     TBLNDX,R2             ; Load index value
    24 016304  016204 002230               MOV     CTLRLST(R2),R4        ; Load address
    25 016310  001421                      BEQ     50$                   ; Branch if done
    26 016312                      20$:
    27 016312  010437 002304               MOV     R4,CTLRTBL            ; and save
    28 016316  005034                      CLR     @(R4)+                ; INITIALIZE xDA
    29 016320  005737 002224               TST     NXMAD                 ; Test for address error
    30 016324  001762                      BEQ     10$                   ; Branch if NO
    31 016326                      30$:
    32 016326  012737 011061 002552        MOV     #ERM102,BASEMSG       ; "Memory error..."
    33 016334  012737 000001 002370        MOV     #B.SADDR,EXTMSK       ; Flags address only
    34 016342  104455                      TRAP    C$ERDF
       016344  000146                      .WORD   102
       016346  010667                      .WORD   HDR101
       016350  015266                      .WORD   ERR101
    35 016352  000747                      BR      10$                   ; and continue
    36 016354                      50$:
    37 016354  012700 000004               MOV     #4,R0
       016360  104436                      TRAP    C$CVEC
    38 016362  012604                      MOV     (SP)+,R4              ;;POP STACK INTO R4
       016364  012602                      MOV     (SP)+,R2              ;;POP STACK INTO R2
    39 016366  000207                      RTS     PC
    40
```

```
    1                                .SBTTL  RCTERR  Process RCT block error
    2                                ;RCTERR
    3                                ;
    4                                ;This routine updates the RCT error table.  The table contains one entry
    5                                ;for each RCT block.  The entry is a bit field for up to 16 copies of each
    6                                ;RCT block.  A new RCT error will set the corresponding copy bit in its entry.
    7                                ;
    8                                ;Inputs:
    9                                ;       RCTCOPY RCT copy count
   10                                ;       RCTTBL  Address of RCT error table
   11                                ;
   12                                ;Outputs:
   13                                ;       STATUS  Exit status
   14                                ;               1:  New bad RCT block
   15                                ;               0:  Bad RCT block not new
   16                                ;              -1:  All copies of RCT block bad
   17                                ;
   18                                ;Calling Seq:
   19                                ;
   20                                ;       JSR     PC,RCTERR
   21                                ;       .WORD   <error exit>
   22                                ;
   23
   24 016370                        RCTERR:
   25
   26 016370  010146                        MOV     R1,-(SP)                ;;PUSH R1 ON STACK
      016372  010246                        MOV     R2,-(SP)                ;;PUSH R2 ON STACK
   27
   28 016374  113700  002424                MOVB    RCTCOPY,R0              ; Load number of RCT copies
   29 016400  012737  000001  002426        MOV     #1,RCTMASK             ; Initialize copy mask
   30 016406                        10$:
   31 016406  005300                        DEC     R0                      ; Decrement copy number
   32 016410  001403                        BEQ     20$                     ; Branch if zero
   33 016412  006337  002426                ASL     RCTMASK                 ; Else, shift mask
   34 016416  000773                        BR      10$                     ; And continue
   35 016420                        20$:
   36 016420  013702  002312                MOV     RCTTBL,R2               ; Load table address
   37 016424  013700  002476                MOV     RCTBLK,R0               ; Load RCT block no.
   38 016430  006300                        ASL     R0                      ; Times two for index
   39 016432  060002                        ADD     R0,R2                   ; Add to table start
   40 016434  033712  002426                BIT     RCTMASK,(R2)            ; Test if this block already bad
   41 016440  001057                        BNE     60$                     ; Branch if YES
   42 016442                        30$:
   43 016442  013704  002324                MOV     HCOM,R4                 ; Load comm area ptr
   44 016446  016446  000142                MOV     HC.CPK+P.LBN+2(R4),-(SP)           ;PUSH HC.CPK+P.LBN+2(R4) ON STACK
      016452  016446  000140                MOV     HC.CPK+P.LBN(R4),-(SP)             ;PUSH HC.CPK+P.LBN(R4) ON STACK
      016456  013746  002476                MOV     RCTBLK,-(SP)            ;PUSH RCTBLK ON STACK
      016462  013746  002424                MOV     RCTCOPY,-(SP)           ;PUSH RCTCOPY ON STACK
      016466  012746  010114                MOV     #INF401,-(SP)           ;PUSH #INF401 ON STACK
      016472  004137  033254                JSR     R1,LPNTB                        ;CALL LPNTB PRINT ROUTINE
      016476  000012                        .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
   45 016500  013746  002354                MOV     EMSTAT,-(SP)            ;PUSH EMSTAT ON STACK
      016504  012746  004035                MOV     #XEVNT,-(SP)            ;PUSH #XEVNT ON STACK
      016510  004137  033254                JSR     R1,LPNTB                        ;CALL LPNTB PRINT ROUTINE
      016514  000004                        .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
   46
   47 016516  053712  002426                BIS     RCTMASK,(R2)            ; Else, mark this block bad
```

```
48 016522  005000              CLR    R0                  ; Clear register
49 016524  113700  002423      MOVB   COPIES,R0           ; Load number of copies
50 016530  011202              MOV    (R2),R2             ; Load entry
51 016532  000241              CLC                        ; Clear carry
52 016534              35$:
53 016534  006202              ASR    R2                  ; Shift LSB into Carry
54 016536  103011              BCC    50$                 ; Branch if bit clear
55 016540  005300              DEC    R0                  ; Decrement copy count
56 016542  001374              BNE    35$                 ; Continue until zero
57
58                    ;       Fatal error exit
59 016544              40$:                                ; All copies bad
60 016544  012602              MOV    (SP)+,R2            ;;POP STACK INTO R2
   016546  012601              MOV    (SP)+,R1            ;;POP STACK INTO R1
61 016550  017616  000000      MOV    @(SP),(SP)          ; Load fatal error return
62 016554  012700  177777      MOV    #-1,R0              ; Load fatal status
63 016560  000207              RTS    PC                  ; Return
64
65                    ;       Report new bad RCT block
66 016562              50$:
67 016562  012602              MOV    (SP)+,R2            ;;POP STACK INTO R2
   016564  012601              MOV    (SP)+,R1            ;;POP STACK INTO R1
68 016566  062716  000002      ADD    #2,(SP)             ; Bump past error return
69 016572  012700  000001      MOV    #1,R0               ; Report new bad RCT block
70 016576  000207              RTS    PC                  ; Return
71
72                    ;       Bad RCT block was not new
73 016600              60$:
74 016600  012602              MOV    (SP)+,R2            ;;POP STACK INTO R2
   016602  012601              MOV    (SP)+,R1            ;;POP STACK INTO R1
75 016604  062716  000002      ADD    #2,(SP)             ; Bump past error return
76 016610  012700  000000      MOV    #0,R0               ; Report bad RCT block not new
77 016614  000207              RTS    PC                  ; Return
78
```

```
    1                                        .SBTTL  ERRLOG  Process error log packet
    2                                        ;ERRLOG
    3                                        ;
    4                                        ;This routine processes all ERRORLOG msg packets.  Errorlog types 0, 1, and 3
    5                                        ;will cause a DEVICE FATAL error.  Disk transfer errors will print msg.
    6                                        ;
    7                                        ;Inputs:
    8                                        ;       HCOM    Host comm. area
    9                                        ;Outputs:
   10                                        ;       STATUS  Exit status
   11                                        ;                0:  Disk transfer errorlog packet (no msg)
   12                                        ;               -1:  Fatal errorlog packet (err msg)
   13                                        ;
   14                                        ;Calling Seq:
   15                                        ;
   16                                        ;       JSR     PC,ERRLOG
   17                                        ;       .WORD   <error exit>
   18                                        ;
   19
   20 016616                                 ERLPTBL:
   21 016616   016734                                .WORD   ERRL0                   ; Controller error
   22 016620   016772                                .WORD   ERRL1                   ; Memory access error
   23 016622   017066                                .WORD   ERRL2                   ; Disk transfer error
   24 016624   017224                                .WORD   ERRL3                   ; SDI error
   25 016626   017462                                .WORD   ERRL4                   ; Small disk error
   26
   27           000005                        ERLPSIZ =      <.-ERLPTBL>/2            ; maximum index
   28
   29 016630   000000  000000                 ERLAST: .WORD   0,0                     ; Last LBN
   30
   31 016634                                  ERRLOG:
   32 016634   010146                                MOV     R1,-(SP)                ;;PUSH R1 ON STACK
   33
   34 016636   013704  002324                        MOV     HCOM,R4                 ; Load comm area ptr
   35 016642   116437  000030  002372                MOVB    HC.MPK+L.FMT(R4),ERLFMT ; Save format specifier
   36 016650   016437  000032  002362                MOV     HC.MPK+L.EVNT(R4),EVENT ; Save event code
   37 016656   113701  002372                        MOVB    ERLFMT,R1               ; Load formst spec
   38 016662   122701  000005                        CMPB    #ERLPSIZ,R1             ; Compare with max
   39 016666   003017                                BGT     10$                     ; Branch if OK
   40
   41                                         ;       Unrecognized errorlog format
   42 016670                                  5$:
   43 016670   012764  140000  000006                MOV     #RG.OWN+RG.FLG,HC.MCT(R4); Return msg descriptor to controller
   44 016676   012737  000001  002370                MOV     #B.EVNT,EXTMSK          ; Load extended msg mask
   45 016704   012737  013763  002552                MOV     #ERM700,BASEMSG         ; Load basic msg address
   46 016712   104455                                TRAP    C$ERDF
      016714   001274                                .WORD   700
      016716   013731                                .WORD   HDR701
      016720   015770                                .WORD   ERR701
   47 016722   000137  017514                        JMP     ERLEXIT                 ; Exit
   48
   49                                         ;       Process errorlog
   50 016726                                  10$:
   51 016726   006301                                ASL     R1                      ; Shift index
   52 016730   000171  016616                        JMP     @ERLPTBL(R1)            ; Process errorlog
   53
```

```
    1
    2                                        ;       Controller error
    3 016734                         ERRL0:
    4 016734  012764  140000  000006         MOV     #RG.OWN+RG.FLG,HC.MCT(R4); Return msg descriptor to controller
    5 016742  012737  000001  002370         MOV     #B.EVNT,EXTMSK            ; Load extended msg mask
    6 016750  012737  014024  002552         MOV     #ERM701,BASEMSG          ; Load basic msg address
    7 016756  104455                         TRAP    C$ERDF
      016760  001275                         .WORD   701
      016762  013731                         .WORD   HDR701
      016764  015770                         .WORD   ERR701
    8 016766  000137  017514                 JMP     ERLEXIT                  ; Exit
    9
   10                                        ;       Memory access error
   11 016772                         ERRL1:
   12 016772  016437  000050  002400         MOV     HC.MPK+L.BADR(R4),HADDR; Save
   13 017000  016437  000052  002402         MOV     HC.MPK+L.BADR+2(R4),HADDR+2; address
   14
   15 017006  012764  140000  000006         MOV     #RG.OWN+RG.FLG,HC.MCT(R4); Return msg descriptor to controller
   16
   17 017014  012737  000001  002370         MOV     #B.EVNT,EXTMSK           ; Load extended msg mask
   18 017022  012737  014062  002552         MOV     #ERM702,BASEMSG          ; Load basic msg address
   19 017030  104455                         TRAP    C$ERDF
      017032  001276                         .WORD   702
      017034  013731                         .WORD   HDR701
      017036  015770                         .WORD   ERR701
   20 017040  013746  002402                 MOV     HADDR+2,-(SP)            ;PUSH HADDR+2 ON STACK
      017044  013746  002400                 MOV     HADDR,-(SP)              ;PUSH HADDR ON STACK
      017050  012746  004063                 MOV     #XHADDR,-(SP)            ;PUSH #XHADDR ON STACK
      017054  004137  033264                 JSR     R1,LPNTX                      ;CALL LPNTX PRINT ROUTINE
      017060  000006                         .WORD   ARG.CT                   ;ARGUMENT COUNT * 2
   21 017062  000137  017514                 JMP     ERLEXIT                  ; Exit
   22
```

```
   1
   2                                    ;        Disk transfer error
   3 017066                            ERRL2:
   4 017066  016437  000070  002374     MOV    HC.MPK+L.HDCD(R4),ERLLBN; Save
   5 017074  016437  000072  002376     MOV    HC.MPK+L.HDCD+2(R4),ERLLBN+2; LBN
   6 017102  012764  140000  000006     MOV    #RG.OWN+RG.FLG,HC.MCT(R4); Return msg descriptor to controller
   7
   8 017110  000436                      BR     20$                      ; Branch if NO
   9 017112  023737  002374  016630      CMP    ERLLBN,ERLAST            ; Else, test last LBN (low)
  10 017120  001004                      BNE    10$                      ; Branch if not same
  11 017122  023737  002376  016632      CMP    ERLLBN+2,ERLAST+2        ; Test LBN (high)
  12 017130  001426                      BEQ    20$                      ; Branch if same
  13 017132                            10$:
  14 017132  013737  002374  016630      MOV    ERLLBN,ERLAST            ; Save
  15 017140  013737  002376  016632      MOV    ERLLBN+2,ERLAST+2        ; Last LBN
  16
  17 017146  013746  002376             MOV    ERLLBN+2,-(SP)           ;PUSH ERLLBN+2 ON STACK
     017152  013746  002374             MOV    ERLLBN,-(SP)             ;PUSH ERLLBN ON STACK
     017156  012746  010166             MOV    #INF405,-(SP)            ;PUSH #INF405 ON STACK
     017162  004137  033264             JSR    R1,LPNTX                      ;CALL LPNTX PRINT ROUTINE
     017166  000006                     .WORD  ARG.CT                   ;ARGUMENT COUNT * 2
  18 017170  013746  002362             MOV    EVENT,-(SP)              ;PUSH EVENT ON STACK
     017174  012746  004035             MOV    #XEVNT,-(SP)             ;PUSH #XEVNT ON STACK
     017200  004137  033264             JSR    R1,LPNTX                      ;CALL LPNTX PRINT ROUTINE
     017204  000004                     .WORD  ARG.CT                   ;ARGUMENT COUNT * 2
  19 017206                            20$:
  20 017206  012601                      MOV    (SP)+,R1                 ;;POP STACK INTO R1
  21 017210  062716  000002             ADD    #2,(SP)                  ; Bump past error return
  22 017214  012737  000000  002442      MOV    #0,STATUS                ; Load success
  23 017222  000207                      RTS    PC                       ; Return to caller
  24
```

```
    1
    2                                      ;      SDI error
    3 017224                        ERRL3:
    4 017224    016437  000070  002374     MOV    HC.MPK+L.HDCD(R4),ERLLBN; Save
    5 017232    016437  000072  002376     MOV    HC.MPK+L.HDCD+2(R4),ERLLBN+2; LBN
    6 017240    062704  000100             ADD    #HC.MPK+L.DS,R4           ; Add offset to status/error info
    7 017244    012701  002406             MOV    #ERLDS,R1                ; Load local address
    8 017250    012702  000004             MOV    #4,R2                   ; Load counter
    9 017254                        10$:
   10 017254    012421                     MOV    (R4)+,(R1)+             ; Move data
   11 017256    005302                     DEC    R2                      ; Decrement count
   12 017260    001375                     BNE    10$                     ; continue until zero
   13 017262                        15$:
   14 017262    013704  002324             MOV    HCOM,R4                 ; Reload comm area ptr
   15 017266    012764  140000  000006     MOV    #RG.OWN+RG.FLG,HC.MCT(R4); Return msg descriptor to controller
   16
   17 017274    012737  000001  002370     MOV    #B.EVNT,EXTMSK          ; Load extended msg mask
   18 017302    012737  014130  002552     MOV    #ERM703,BASEMSG         ; Load basic msg address
   19 017310    104455                     TRAP   C$ERDF
      017312    001277                     .WORD  703
      017314    013731                     .WORD  HDR701
      017316    015770                     .WORD  ERR701
   20
   21 017320    013746  002376             MOV    ERLLBN+2,-(SP)          ;PUSH ERLLBN+2 ON STACK
      017324    013746  002374             MOV    ERLLBN,-(SP)            ;PUSH ERLLBN ON STACK
      017330    012746  014575             MOV    #XLBN,-(SP)             ;PUSH #XLBN ON STACK
      017334    004137  033264             JSR    R1,LPNTX                       ;CALL LPNTX PRINT ROUTINE
      017340    000006                     .WORD  ARG.CT                  ;ARGUMENT COUNT * 2
   22
   23 017342    005737  003246             TST    MDLNDX                  ; Test for known model
   24 017346    001421                     BEQ    30$                     ; Branch if zero
   25
   26 017350    012701  000006             MOV    #6,R1                   ; Load drive code byte index
   27 017354    122737  000004  003244     CMPB   #M.RA60,MODEL           ; Test for RA60
   28 017362    001001                     BNE    20$                     ; Branch if NO
   29 017364    005201                     INC    R1                      ; Else, change index for RA60
   30 017366                        20$:
   31 017366    116137  002406  002404     MOVB   ERLDS(R1),DCODE         ; Save drive error code
   32 017374    013746  002404             MOV    DCODE,-(SP)             ;PUSH DCODE ON STACK
      017400    012746  004112             MOV    #XDCODE,-(SP)           ;PUSH #XDCODE ON STACK
      017404    004137  033264             JSR    R1,LPNTX                       ;CALL LPNTX PRINT ROUTINE
      017410    000004                     .WORD  ARG.CT                  ;ARGUMENT COUNT * 2
   33 017412                        30$:
   34 017412    012746  004152             MOV    #XDINFO,-(SP)           ;PUSH #XDINFO ON STACK
      017416    004137  033264             JSR    R1,LPNTX                       ;CALL LPNTX PRINT ROUTINE
      017422    000002                     .WORD  ARG.CT                  ;ARGUMENT COUNT * 2
   35 017424    012702  000010             MOV    #8.,R2                  ; Load byte count
   36 017430    012701  002406             MOV    #ERLDS,R1               ; Load ptr to status info
   37 017434                        35$:
   38 017434    112104                     MOVB   (R1)+,R4                ; Load status
   39 017436    010446                     MOV    R4,-(SP)                ;PUSH R4 ON STACK
      017440    012746  004217             MOV    #XDHEX,-(SP)            ;PUSH #XDHEX ON STACK
      017444    004137  033264             JSR    R1,LPNTX                       ;CALL LPNTX PRINT ROUTINE
      017450    000004                     .WORD  ARG.CT                  ;ARGUMENT COUNT * 2
   40 017452    005302                     DEC    R2                      ; Decrement count
   41 017454    001367                     BNE    35$                     ; continue until zero
   42 017456    000137  017514             JMP    ERLEXIT                 ; Exit
```

```
  1
  2                                     ;       Small disk error
  3 017462                      ERRL4:
  4 017462   012764 140000 000006        MOV     #RG.OWN+RG.FLG,HC.MCT(R4); Return msg descriptor to controller
  5 017470   012737 000001 002370        MOV     #B.EVNT,EXTMSK       ; Load extended msg mask
  6 017476   012737 014157 002552        MOV     #ERM704,BASEMSG      ; Load basic msg address
  7 017504   104455                      TRAP    C$ERDF
    017506   001300                      .WORD   704
    017510   013731                      .WORD   HDR701
    017512   015770                      .WORD   ERR701
  8
  9 017514                      ERLEXIT:
 10 017514   013704 002324               MOV     HCOM,R4             ; Reload comm area ptr
 11 017520   005737 002650               TST     DEBUG               ; Test for DEBUG
 12 017524   001007                      BNE     20$                 ; Branch if yes
 13 017526                      10$:
 14 017526   012601                      MOV     (SP)+,R1            ;;POP STACK INTO R1
 15 017530   017616 000000               MOV     @(SP),(SP)          ; Load fatal return
 16 017534   012737 177777 002442        MOV     #-1,STATUS          ; Fatal status
 17 017542   000207                      RTS     PC                  ; and return
 18 017544                      20$:
 19 017544   012601                      MOV     (SP)+,R1            ;;POP STACK INTO R1
 20 017546   062716 000002               ADD     #2,(SP)             ; Bump past error return
 21 017552   012737 000000 002442        MOV     #0,STATUS           ; Load success
 22 017560   000207                      RTS     PC                  ; Return to caller
 23
```

```
   1                              .SBTTL  CACHTST Bad block cache test
   2                              ;CACHTST
   3                              ;
   4                              ;Inputs:
   5                              ;       LBN     Current LBN
   6                              ;       CACHTBL Address of caching table
   7                              ;
   8                              ;Outputs:
   9                              ;       STATUS  Exit status
  10                              ;          1:  LBN already in caching table
  11                              ;          0:  LBN entered in caching table
  12                              ;         -1:  Caching table full
  13                              ;
  14                              ;Calling Seq:
  15                              ;
  16                              ;       JSR     PC,CACHTST
  17                              ;       .WORD   <error exit>
  18                              ;
  19
  20 017562                      CACHTST:
  21 017562  010146                      MOV     R1,-(SP)           ;;PUSH R1 ON STACK
     017564  010246                      MOV     R2,-(SP)           ;;PUSH R2 ON STACK
     017566  010346                      MOV     R3,-(SP)           ;;PUSH R3 ON STACK
  22 017570  013702  002310              MOV     CACHTBL,R2         ; Load table start
  23 017574  005000                      CLR     R0                 ; Clear cache entry count
  24 017576                      20$:
  25 017576  005762  000002              TST     2(R2)              ; Test for null
  26 017602  100461                      BMI     50$                ; Branch if YES
  27
  28 017604  023712  002446              CMP     LBN,(R2)           ; Compare low order LBN
  29 017610  001007                      BNE     30$                ; Branch if not same
  30 017612  016201  000002              MOV     2(R2),R1           ; Else, load high order LBN
  31 017616  042701  170000              BIC     #170000,R1         ; Clear flags
  32 017622  020137  002450              CMP     R1,LBN+2           ; Compare high order LBN
  33 017626  001431                      BEQ     40$                ; Branch if same
  34 017630                      30$:
  35 017630  062702  000004              ADD     #4,R2              ; Bump to next entry
  36 017634  005200                      INC     R0                 ; Increment count
  37 017636  022700  000620              CMP     #CACHESIZE,R0      ; Compare with max
  38 017642  003355                      BGT     20$                ; Branch if less
  39 017644                      35$:
  40 017644  012737  013621  002552      MOV     #ERM607,BASEMSG    ; "transient error overflow..."
  41 017652  012737  000000  002370      MOV     #0,EXTMSK          ; No extended msg
  42 017660  104455                      TRAP    C$ERDF
     017662  001137                      .WORD   607
     017664  013173                      .WORD   HDR601
     017666  015720                      .WORD   ERR601
  43 017670  012603                      MOV     (SP)+,R3           ;;POP STACK INTO R3
     017672  012602                      MOV     (SP)+,R2           ;;POP STACK INTO R2
     017674  012601                      MOV     (SP)+,R1           ;;POP STACK INTO R1
  44 017676  017616  000000              MOV     @(SP),(SP)         ; Else, load error return
  45 017702  012737  177777  002442      MOV     #-1,STATUS         ; Set fatal status
  46 017710  000207                      RTS     PC                 ; and return
  47 017712                      40$:
  48 017712  012712  177777              MOV     #-1,(R2)           ; Reset LBN (low)
  49 017716  012762  177777  000002      MOV     #-1,2(R2)          ; Reset LBN (high)
  50
```

```
51 017724  012603                      MOV     (SP)+,R3          ;;POP STACK INTO R3
   017726  012602                      MOV     (SP)+,R2          ;;POP STACK INTO R2
   017730  012601                      MOV     (SP)+,R1          ;;POP STACK INTO R1
52 017732  062716  000002              ADD     #2,(SP)           ; Bump past error return
53 017736  012737  000001  002442      MOV     #1,STATUS         ; Set replace status
54 017744  000207                      RTS     PC                ; and return
55 017746                       50$:
56 017746  013712  002446              MOV     LBN,(R2)          ; Save LBN
57 017752  013762  002450  000002      MOV     LBN+2,2(R2)       ;
58 017760  012603                      MOV     (SP)+,R3          ;;POP STACK INTO R3
   017762  012602                      MOV     (SP)+,R2          ;;POP STACK INTO R2
   017764  012601                      MOV     (SP)+,R1          ;;POP STACK INTO R1
59 017766  062716  000002              ADD     #2,(SP)           ; Bump past error return
60 017772  012737  000000  002442      MOV     #0,STATUS         ; Set replace status
61 020000  000207                      RTS     PC                ; and return
62
```

```
 1                                      .SBTTL  TYPSTAT Type replacement status
 2                                      ;TYPSTAT
 3                                      ;
 4                                      ;THIS ROUTINE TYPES THE LBN, RBN REPLACEMENT INFORMATION TO THE TTY.
 5                                      ;
 6                                      ;INPUTS:
 7                                      ;       SFLAG (Search status flag),
 8                                      ;       MATFLG (Match status flag),
 9                                      ;       LBN (Lo order word of LBN being replaced),
10                                      ;       LBN+2 (Hi order word of LBN being replaced),
11                                      ;       RBN (Lo order word of RBN replacement),
12                                      ;       RBN+2 (Hi order word of RBN replacement),
13                                      ;       MATRBN (Lo order word of RBN which matched the LBN being replaced),
14                                      ;       MATRBN+2 (Hi order word of RBN which matched the LBN being replaced)
15                                      ;
16                                      ;OUTPUTS:
17                                      ;       TYPE APPROPRIATE MESSAGE TO THE TTY
18                                      ;
19                                      ;       LBNSTAT: LBN status word
20                                      ;
21
22 020002                      TYPSTAT:
23 020002  010146                      MOV     R1,-(SP)               ;;PUSH R1 ON STACK
   020004  010246                      MOV     R2,-(SP)               ;;PUSH R2 ON STACK
24
25 020006                      10$:
26 020006  105737  002514              TSTB    MATFLG                 ; Test for LBN match
27 020012  001444                      BEQ     20$                    ; Branch if NO, else print msg
28
29                              ;       Print UNUSABLE operation
30 020014                      15$:
31 020014  005237  002440              INC     UNUSE                  ; Increment number of UNUSABLE RBNs
32 020020  013746  002450              MOV     LBN+2,-(SP)            ;PUSH LBN+2 ON STACK
   020024  013746  002446              MOV     LBN,-(SP)              ;PUSH LBN ON STACK
   020030  012746  010166              MOV     #INF405,-(SP)          ;PUSH #INF405 ON STACK
   020034  004137  033254              JSR     R1,LPNTB                   ;CALL LPNTB PRINT ROUTINE
   020040  000006                      .WORD   ARG.CT                 ;ARGUMENT COUNT * 2
33 020042  012746  015174              MOV     #TX.BLK,-(SP)          ;PUSH #TX.BLK ON STACK
   020046  012746  010206              MOV     #INF406,-(SP)          ;PUSH #INF406 ON STACK
   020052  004137  033254              JSR     R1,LPNTB                   ;CALL LPNTB PRINT ROUTINE
   020056  000004                      .WORD   ARG.CT                 ;ARGUMENT COUNT * 2
34 020060  013746  002464              MOV     MATRBN+2,-(SP)         ;PUSH MATRBN+2 ON STACK
   020064  013746  002462              MOV     MATRBN,-(SP)           ;PUSH MATRBN ON STACK
   020070  012746  010224              MOV     #INF407,-(SP)          ;PUSH #INF407 ON STACK
   020074  004137  033254              JSR     R1,LPNTB                   ;CALL LPNTB PRINT ROUTINE
   020100  000006                      .WORD   ARG.CT                 ;ARGUMENT COUNT * 2
35 020102  012746  015131              MOV     #TX.UNS,-(SP)          ;PUSH #TX.UNS ON STACK
   020106  012746  010243              MOV     #INF408,-(SP)          ;PUSH #INF408 ON STACK
   020112  004137  033254              JSR     R1,LPNTB                   ;CALL LPNTB PRINT ROUTINE
   020116  000004                      .WORD   ARG.CT                 ;ARGUMENT COUNT * 2
36 020120  105037  002514              CLRB    MATFLG                 ; Clear flag
37 020124                      20$:
38 020124  012702  000000              MOV     #0,R2                  ; Zero index
39 020130  013701  002516              MOV     LBNSTAT,R1             ; Load status
40 020134  100422                      BMI     30$                    ; Branch if error status
41 020136  003111                      BGT     50$                    ; Else, branch if flag set
42
```

```
 43                                  ;        Print LBN with blank status
 44 020140                        25$:
 45 020140   013746   002450         MOV     LBN+2,-(SP)        ;PUSH LBN+2 ON STACK
    020144   013746   002446         MOV     LBN,-(SP)          ;PUSH LBN ON STACK
    020150   012746   010166         MOV     #INF405,-(SP)      ;PUSH #INF405 ON STACK
    020154   004137   033254         JSR     R1,LPNTB                 ;CALL LPNTB PRINT ROUTINE
    020160   000006                  .WORD   ARG.CT             ;ARGUMENT COUNT * 2
 46 020162   012746   015174         MOV     #TX.BLK,-(SP)      ;PUSH #TX.BLK ON STACK
    020166   012746   010206         MOV     #INF406,-(SP)      ;PUSH #INF406 ON STACK
    020172   004137   033254         JSR     R1,LPNTB                 ;CALL LPNTB PRINT ROUTINE
    020176   000004                  .WORD   ARG.CT             ;ARGUMENT COUNT * 2
 47 020200   000400                  BR      30$                ; Exit
 48
 49                                  ;        Print LBN status code
 50 020202                        30$:
 51 020202   042701   177760         BIC     #^C17,R1           ; CLear all but data codes
 52 020206   001436                  BEQ     40$                ; Branch if None set
 53 020210                        32$:
 54 020210   036201   020612         BIT     BITTBL(R2),R1      ; Test for bit set
 55 020214   001426                  BEQ     35$                ; Branch if bit clear
 56
 57 020216   013746   002450         MOV     LBN+2,-(SP)        ;PUSH LBN+2 ON STACK
    020222   013746   002446         MOV     LBN,-(SP)          ;PUSH LBN ON STACK
    020226   012746   010166         MOV     #INF405,-(SP)      ;PUSH #INF405 ON STACK
    020232   004137   033254         JSR     R1,LPNTB                 ;CALL LPNTB PRINT ROUTINE
    020236   000006                  .WORD   ARG.CT             ;ARGUMENT COUNT * 2
 58 020240   016246   020624         MOV     STATBL(R2),-(SP)          ;PUSH STATBL(R2) ON STACK
    020244   012746   010206         MOV     #INF406,-(SP)      ;PUSH #INF406 ON STACK
    020250   004137   033254         JSR     R1,LPNTB                 ;CALL LPNTB PRINT ROUTINE
    020254   000004                  .WORD   ARG.CT             ;ARGUMENT COUNT * 2
 59 020256   005272   020636         INC     @CNTTBL(R2)        ; Increment counter
 60 020262   046237   020612  002516 BIC     BITTBL(R2),LBNSTAT ; Clear so we won't print again
 61 020270   000405                  BR      40$                ; Exit loop
 62 020272                        35$:
 63 020272   062702   000002         ADD     #2,R2              ; Else, bump index
 64 020276   005762   020624         TST     STATBL(R2)         ; Test address
 65 020302   100342                  BPL     32$                ; continue if not null
 66
```

```
  1
  2                                      ;         Test for Write w/FE
  3 020304                              40$:
  4 020304  032737  000400  002516       BIT     #TF.WFE,LBNSTAT      ; Test for WFE
  5 020312  001451                        BEQ     60$                 ; Branch if NO
  6
  7 020314  013746  002450               MOV     LBN+2,-(SP)          ;PUSH LBN+2 ON STACK
    020320  013746  002446               MOV     LBN,-(SP)            ;PUSH LBN ON STACK
    020324  012746  010166               MOV     #INF405,-(SP)        ;PUSH #INF405 ON STACK
    020330  004137  033254               JSR     R1,LPNTB                  ;CALL LPNTB PRINT ROUTINE
    020334  000006                        .WORD   ARG.CT              ;ARGUMENT COUNT * 2
  8 020336  012746  015201               MOV     #TX.WFE,-(SP)        ;PUSH #TX.WFE ON STACK
    020342  012746  010206               MOV     #INF406,-(SP)        ;PUSH #INF406 ON STACK
    020346  004137  033254               JSR     R1,LPNTB                  ;CALL LPNTB PRINT ROUTINE
    020352  000004                        .WORD   ARG.CT              ;ARGUMENT COUNT * 2
  9 020354  005237  002542               INC     WFECNT              ; Increment count
 10 020360  000426                        BR      60$                 ; Type replacement info
 11
 12                                      ;         Test for BBR
 13 020362                              50$:
 14 020362  032737  000200  002516       BIT     #TF.BBR,LBNSTAT      ; Test for BBR
 15 020370  001422                        BEQ     60$                 ; Branch if NO
 16
 17 020372  013746  002450               MOV     LBN+2,-(SP)          ;PUSH LBN+2 ON STACK
    020376  013746  002446               MOV     LBN,-(SP)            ;PUSH LBN ON STACK
    020402  012746  010166               MOV     #INF405,-(SP)        ;PUSH #INF405 ON STACK
    020406  004137  033254               JSR     R1,LPNTB                  ;CALL LPNTB PRINT ROUTINE
    020412  000006                        .WORD   ARG.CT              ;ARGUMENT COUNT * 2
 18 020414  012746  015150               MOV     #TX.BBR,-(SP)        ;PUSH #TX.BBR ON STACK
    020420  012746  010206               MOV     #INF406,-(SP)        ;PUSH #INF406 ON STACK
    020424  004137  033254               JSR     R1,LPNTB                  ;CALL LPNTB PRINT ROUTINE
    020430  000004                        .WORD   ARG.CT              ;ARGUMENT COUNT * 2
 19 020432  005237  002540               INC     BBRCNT              ; Increment count
 20
```

```
    1                                          ;         Print replacement operation
    2 020436                          60$:
    3 020436   005737  002520                  TST      RPLSTAT                    ; Test replacement status
    4 020442   001455                           BEQ      100$                       ; Exit if none
    5
    6 020444   122737  000000  002515            CMPB     #0,SFLAG                   ; Test for PRIMARY search status
    7 020452   001023                           BNE      80$                        ; Branch if NO
    8
    9                                          ;         Print PRIMARY operation
   10
   11 020454   005237  002434                  INC      PRIMRY                     ; Increment PRIMARY RBN count
   12 020460   013746  002454                  MOV      RBN+2,-(SP)                ;PUSH RBN+2 ON STACK
      020464   013746  002452                  MOV      RBN,-(SP)                  ;PUSH RBN ON STACK
      020470   012746  010224                  MOV      #INF407,-(SP)              ;PUSH #INF407 ON STACK
      020474   004137  033254                  JSR      R1,LPNTB                        ;CALL LPNTB PRINT ROUTINE
      020500   000006                          .WORD    ARG.CT                     ;ARGUMENT COUNT * 2
   13 020502   012746  015117                  MOV      #TX.PRI,-(SP)              ;PUSH #TX.PRI ON STACK
      020506   012746  010243                  MOV      #INF408,-(SP)              ;PUSH #INF408 ON STACK
      020512   004137  033254                  JSR      R1,LPNTB                        ;CALL LPNTB PRINT ROUTINE
      020516   000004                          .WORD    ARG.CT                     ;ARGUMENT COUNT * 2
   14 020520   000426                          BR       100$                       ; and exit
   15 020522                          80$:
   16 020522   122737  000001  002515            CMPB     #1,SFLAG                   ; Test for SECONDARY search status
   17 020530   001022                           BNE      100$                       ; Branch if NO
   18
   19                                          ;         Print SCEONDARY operation
   20
   21 020532   005237  002436                  INC      SCNDRY                     ; Increment SECONDARY RBN count
   22 020536   013746  002454                  MOV      RBN+2,-(SP)                ;PUSH RBN+2 ON STACK
      020542   013746  002452                  MOV      RBN,-(SP)                  ;PUSH RBN ON STACK
      020546   012746  010224                  MOV      #INF407,-(SP)              ;PUSH #INF407 ON STACK
      020552   004137  033254                  JSR      R1,LPNTB                        ;CALL LPNTB PRINT ROUTINE
      020556   000006                          .WORD    ARG.CT                     ;ARGUMENT COUNT * 2
   23 020560   012746  015124                  MOV      #TX.SEC,-(SP)              ;PUSH #TX.SEC ON STACK
      020564   012746  010243                  MOV      #INF408,-(SP)              ;PUSH #INF408 ON STACK
      020570   004137  033254                  JSR      R1,LPNTB                        ;CALL LPNTB PRINT ROUTINE
      020574   000004                          .WORD    ARG.CT                     ;ARGUMENT COUNT * 2
   24 020576                          100$:
   25 020576   012737  000000  002520            MOV      #0,RPLSTAT                 ; Zero replacement status
   26 020604   012601                           MOV      (SP)+,R1                   ;;POP STACK INTO R1
      020606   012602                           MOV      (SP)+,R2                   ;;POP STACK INTO R2
   27 020610   000207                           RTS      PC                         ;RETURN
   28
   29
   30 020612                          BITTBL:
   31 020612   000001                          .WORD    TF.FER
   32 020614   000002                          .WORD    TF.HCE
   33 020616   000004                          .WORD    TF.DST
   34 020620   000010                          .WORD    TF.ECC
   35 020622   177777                          .WORD    -1
   36
   37 020624                          STATBL:
   38 020624   015136                          .WORD    TX.FER                     ; BIT0
   39 020626   015206                          .WORD    TX.HCE                     ; BIT1
   40 020630   015213                          .WORD    TX.DST                     ; BIT2
   41 020632   015143                          .WORD    TX.ECH                     ; BIT3
   42 020634   177777                          .WORD    -1                         ; Terminator
```

```
43
44 020636                    CNTTBL:
45 020636  002530                   .WORD   FERCNT              ; FER count
46 020640  002532                   .WORD   HCECNT              ; HCE count
47 020642  002534                   .WORD   DSTCNT              ; DST count
48 020644  002536                   .WORD   ECCCNT              ; ECC count
49 020646  177777                   .WORD   -1                  ; Terminator
50
```

```
 1                                    .SBTTL  RNTIME   Print runtime
 2                                ;RNTIME
 3                                ;
 4                                ;PRINT RUNTIME
 5                                ;
 6                                ;INPUTS:
 7                                ;        KW.EL - CONTAINS ELAPSED TIME
 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  020650  005737  002210       RNTIME: TST     KW.CSR                  ;CHECK IF A CLOCK PRESENT
15  020654  001470                       BEQ     RNTIMX                  ;BRANCH IF NOT
16  020656  010046                       MOV     R0,-(SP)                ;;PUSH R0 ON STACK
    020660  010346                       MOV     R3,-(SP)                ;;PUSH R3 ON STACK
    020662  010446                       MOV     R4,-(SP)                ;;PUSH R4 ON STACK
    020664  010546                       MOV     R5,-(SP)                ;;PUSH R5 ON STACK
17  020666  013703  002220               MOV     KW.EL,R3                ;GET ELAPSED TIME
18  020572  013704  002222               MOV     KW.EL+2,R4
19  020676  013700  002216               MOV     KW.HZ,R0                ;GET SPEED OF CLOCK
20  020702  004737  034436               JSR     PC,DIVIDE               ;COMPUTE SECONDS OF ELAPSED TIME
21  020706  012700  000074               MOV     #60.,R0                 ;NOW DIVIDE BY 60
22  020712  004737  034436               JSR     PC,DIVIDE               ;  TO COMPUTE MINUTES
23  020716  010546                       MOV     R5,-(SP)                ;;PUSH R5 ON STACK
24  020720  004737  034436               JSR     PC,DIVIDE               ;DIVIDE BY 60 AGAIN
25  020724  010346                       MOV     R3,-(SP)                ;PUSH R3 ON STACK
    020726  012746  014233               MOV     #RNTIM,-(SP)            ;PUSH #RNTIM ON STACK
    020732  004137  033302               JSR     R1,LPNT                 ;CALL LPNT PRINT ROUTINE
    020736  000004                       .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
26  020740  020527  000011               CMP     R5,#9.                  ;IF MINUTES 9 OR LESS
27  020744  003004                       BGT     1$
28  020746  112700  000060               MOVB    #'0,R0                  ;STORE #'0 IN R0 AND
    020752  004737  033066               JSR     PC,PRINTC               ;PRINT THE CHARACTER.
29  020756                        1$:
    020756  010546                       MOV     R5,-(SP)                ;PUSH R5 ON STACK
    020760  012746  014266               MOV     #RNTIM1,-(SP)           ;PUSH #RNTIM1 ON STACK
    020764  004137  033302               JSR     R1,LPNT                 ;CALL LPNT PRINT ROUTINE
    020770  000004                       .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
30  020772  012605                       MOV     (SP)+,R5                ;;POP STACK INTO R5
31  020774  020527  000011               CMP     R5,#9.                  ;IF 9 OR LESS
32  021000  003004                       BGT     2$
33  021002  112700  000060               MOVB    #'0,R0                  ;STORE #'0 IN R0 AND
    021006  004737  033066               JSR     PC,PRINTC               ;PRINT THE CHARACTER.
34  021012                        2$:
    021012  010546                       MOV     R5,-(SP)                ;PUSH R5 ON STACK
    021014  012746  014274               MOV     #RNTIM2,-(SP)           ;PUSH #RNTIM2 ON STACK
    021020  004137  033302               JSR     R1,LPNT                 ;CALL LPNT PRINT ROUTINE
    021024  000004                       .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
35  021026  012605                       MOV     (SP)+,R5                ;;POP STACK INTO R5
    021030  012604                       MOV     (SP)+,R4                ;;POP STACK INTO R4
    021032  012603                       MOV     (SP)+,R3                ;;POP STACK INTO R3
    021034  012600                       MOV     (SP)+,R0                ;;POP STACK INTO R0
36  021036                        RNTIMX:
    021036  112700  000040               MOVB    #'  ,R0                 ;STORE #'  IN R0 AND
    021042  004737  033066               JSR     PC,PRINTC               ;PRINT THE CHARACTER.
```

       37 021046 000207   RTS   PC

```
    1                                    .SBTTL  UNITINI Initialize Unit
    2                                 ;
    3                                 ;UNITINI:
    4                                 ;       Perform the following initialization steps for the UUT:
    5                                 ;
    6                                 ;          Load controller table values
    7                                 ;          Call XDAINT to initialize controller
    8                                 ;          Issue GET UNIT STATUS command
    9                                 ;          Issue SET CONTROLLER CHAR command
   10                                 ;          Put drive ONLINE
   11                                 ;          Allocate and clear RCT error table
   12                                 ;          Read RCT block 0 and format Volume serial number
   13                                 ;          Scan RCT using MULTI-READ
   14                                 ;          Call STEP3 to test for crash during replacement
   15                                 ;
   16                                 ;Inputs:
   17                                 ;       HCOM     Host communication area
   18                                 ;       CTLRTBL  Controller table
   19                                 ;
   20                                 ;Outputs:
   21                                 ;       STATUS  Exit status
   22                                 ;        0:  Disk transfer errorlog packet (no msg)
   23                                 ;       -1:  Fatal errorlog packet (err msg)
   24                                 ;
   25                                 ;Calling Seq:
   26                                 ;
   27                                 ;       JSR     PC,UNITINI
   28                                 ;       .WORD   <error exit>
   29                                 ;
   30
   31 021050                         UNITINI:
   32 021050   013705   002304           MOV     CTLRTBL,R5           ; Load Table address
   33 021054   011537   002314           MOV     (R5),SADDR           ; Save controller address
   34 021060   016546   000002           MOV     C.VEC(R5),-(SP)      ;;PUSH C.VEC(R5) ON STACK
   35 021064   011604                     MOV     (SP),R4              ;GET BR LEVEL/VECTOR OF xDA AND
   36 021066   042704   177000           BIC     #^C<CT.VEC>,R4       ;SAVE THE VECTOR ADDRESS.
   37 021072   012600                     MOV     (SP)+,R0             ;;POP STACK INTO R0
   38 021074   042700   170777           BIC     #^C<CT.BRL>,R0       ;SAVE THE BR LEVEL AND
   39 021100   000300                     SWAB    R0                   ;PUT IT INTO LO BYTE.
   40 021102   010501                     MOV     R5,R1                ;GET ADDRESS OF CONTROLLER TABLE AND
   41 021104   062701   000010           ADD     #C.JSR,R1            ;USE THE TABLE ADDRESS OF THE 'JSR'
   42                                                                 ;INSTRUCTION IN THE CONTROLLER TABLE. AS
   43                                                                 ;THE INTERRUPT SERVICE ROUTINE.
   44 021110   010046                     MOV     R0,-(SP)
      021112   010146                     MOV     R1,-(SP)
      021114   010446                     MOV     R4,-(SP)
      021116   012746   000003           MOV     #3,-(SP)
      021122   104437                     TRAP    C$SVEC
      021124   062706   000010           ADD     #10,SP
   45 021130   006204                     ASR     R4                   ;DIVIDE VECTOR ADDRESS BY 4 TO
   46 021132   006204                     ASR     R4                   ;USE IN xDA INITIALIZE SUBROUTINE.
   47 021134   004737   036670           JSR     PC,XDAINT            ;RING LENGTH, INTERRUPTS AND VECTOR ADDR
   48 021140   001002                     BNE     100$                 ;BRANCH IF NO ERROR, ELSE
   49 021142   000137   022172           JMP     900$                 ; Error exit
   50
```

```
    1                               ;       Get UNIT status
    2 021146                        100$:
    3 021146   004737   030626              JSR     PC.INTCPK               ;INITIALIZE THE COMMAND PACKET
    4                                                                       ; Setup to issue OP.GUS command
      021152   012700   000003              MOV     #OP.GUS,R0              ; Load MSCP opcode
      021156   004737   030204              JSR     PC.ISSUE               ; Issue command to the xDA
      021162   022172                       900$                           ; Here if fatal error,
      021164   001423                       BEQ     120$                   ; Branch if command successful,
    5
    6 021166   013700   002354              MOV     EMSTAT,R0              ; Reload emdmsg status
    7 021172   042700   177740              BIC     #↑CST.MSK,R0           ; Clear any subcodes
    8 021176   022700   000004              CMP     #ST.AVL,R0            ; See if unit is available
    9 021202   001414                       BEQ     120$                   ; Branch if YES
   10 021204                        110$:                                  ; Else, report fatal error
   11 021204   012737   011346   002552     MOV     #ERM201,BASEMSG        ; "Drive not accessable"
   12 021212   012737   000100   002370     MOV     #B.ESTAT,EXTMSK        ; Print status
   13 021220   104455                       TRAP    C$ERDF
      021222   000311                       .WORD   201
      021224   011314                       .WORD   HDR201
      021226   015220                       .WORD   ERR001
   14 021230   000137   022172              JMP     900$                   ; Error exit
   15 021234                        120$:
   16 021234   013704   002324              MOV     HCOM,R4               ; Load comm area address
   17 021240   016437   000064   002420     MOV     HC.MPK+P.TRKS(R4),TRKSIZ; Get the track size
   18 021246   016437   000074   002416     MOV     HC.MPK+P.RCTS(R4),RCTSIZ; Number of blocks in the RCT
   19 021254   003014                       BGT     130$                   ; Continue if RCT size > 0
   20
   21 021256   012737   011743   002552     MOV     #ERM204,BASEMSG        ; "Zero RCT size..."
   22 021264   012737   000000   002370     MOV     #0,EXTMSK             ; No extended info
   23 021272   104455                       TRAP    C$ERDF
      021274   000314                       .WORD   204
      021276   011314                       .WORD   HDR201
      021300   015220                       .WORD   ERR001
   24 021302   000137   022172              JMP     900$                   ; Error exit
   25 021306                        130$:
   26 021306   116437   000076   002422     MOVB    HC.MPK+P.RBNS(R4),RBNS  ; Number of RBN'S
   27 021314   116437   000077   002423     MOVB    HC.MPK+P.RCTC(R4),COPIES; Number of copies of RBN's.
   28
   29                               ;       Set controller characteristics
   30 021322                        200$:
   31 021322   012764   000000   000124     MOV     #0,HC.CPK+P.HTMO(R4)   ; Disable host access timeout
   32 021330   012700   000000              MOV     #0,R0                 ; Assume no error log
   33 021334   005737   002162              TST     ENBERRL               ; Test error log enabled
   34 021340   001402                       BEQ     220$                   ; Branch if NO
   35 021342   052700   000020              BIS     #CF.THS,R0            ; Else, set error log enabled
   36 021346                        220$:
   37 021346   010064   000122              MOV     R0,HC.CPK+P.CNTF(R4)  ; Load characteristics
   38                                                                     ; Setup to issue OP.SCC command
      021352   012700   000004              MOV     #OP.SCC,R0            ; Load MSCP opcode
      021356   004737   030204              JSR     PC.ISSUE              ; Issue command to the xDA
      021362   022172                       900$                          ; Here if fatal error,
      021364   001412                       BEQ     300$                  ; Branch if command successful,
   39                                                                     ; REPORT ERROR
   40 021366   012737   011453   002552     MOV     #ERM202,BASEMSG       ; Failed disable host timeout..."
   41 021374   012737   000100   002370     MOV     #B.ESTAT,EXTMSK       ; Print status
   42 021402   104456                       TRAP    C$ERHRD
      021404   000312                       .WORD   202
```

```
       021406  011314                    .WORD    HDR201
       021410  015220                    .WORD    ERR001
43
```

```
  1
  2                                            ;           Put the drive ONLINE
  3 021412                                     300$:
  4                                                                                ; Setup to issue OP.ONL command
    021412    012700  000011                   MOV       #OP.ONL,R0               ; Load MSCP opcode
    021416    004737  030204                   JSR       PC,ISSUE                 ; Issue command to the xDA
    021422    022172                            900$                              ; Here if fatal error.
    021424    001423                            BEQ       320$                    ; Branch if command successful.
  5
  6 021426    013700  002354                   MOV       EMSTAT,R0               ; Reload emdmsg status
  7 021432    042700  177740                   BIC       #†CST.MSK,R0            ; Clear any subcodes
  8 021436    022700  000400                   CMP       #ST.AOL,R0              ; See if unit is already online
  9 021442    001414                            BEQ       320$                    ; Branch if YES
 10                                                                                ; Else, Report fatal error
 11 021444    012737  011711  002552           MOV       #ERM203,BASEMSG         ; "Drive not ONLINE"
 12 021452    012737  000100  002370           MOV       #B.ESTAT,EXTMSK         ; Print status
 13 021460    104455                            TRAP      C$ERDF
    021462    000313                            .WORD     203
    021464    011314                            .WORD     HDR201
    021466    015220                            .WORD     ERR001
 14 021470    000137  022172                   JMP       900$                    ; Error exit
 15 021474                                     320$:
 16 021474    013704  002324                   MOV       HCOM,R4                 ; Load comm area address
 17 021500    016437  000064  002430           MOV       HC.MPK+P.UNSZ(R4),UNSIZ ; Get LO order word and
 18 021506    016437  000066  002432           MOV       HC.MPK+P.UNSZ+2(R4),UNSIZ+2; HI order word of unit size
 19 021514    116437  000052  003244           MOVB      HC.MPK+P.MODEL(R4),MODEL; Save model type
 20
 21 021522    012700  000012                   MOV       #MDLSIZ,R0              ; Load max model table index
 22 021526                                     325$:
 23 021526    123760  003244  003250           CMPB      MODEL,MDLTBL(R0)        ; Compare with table contents
 24 021534    001403                            BEQ       330$                    ; Branch if match
 25 021536    162700  000002                   SUB       #2,R0                   ; Decrement index
 26 021542    003371                            BGT       325$                    ; Continue until zero
 27 021544                                     330$:
 28 021544    010037  003246                   MOV       R0,MDLNDX               ; Save model index
 29
 30                                            ;           Allocate new RCT error table
 31 021550                                     340$:
 32 021550    013746  002176                   MOV       FFREE,-(SP)             ;;PUSH FFREE ON STACK
    021554    013746  002200                   MOV       FSIZE,-(SP)             ;;PUSH FSIZE ON STACK
 33 021560    013701  002416                   MOV       RCTSIZ,R1               ; Load RCT size
 34 021564    004737  040054                   JSR       PC,ALOCM                ; Allocate error table
 35 021570    010137  002312                   MOV       R1,RCTTBL               ; Save table address
 36 021574    012637  002200                   MOV       (SP)+,FSIZE             ;;POP STACK INTO FSIZE
    021600    012637  002176                   MOV       (SP)+,FFREE             ;;POP STACK INTO FFREE
 37
 38                                            ;           Clear RCT error table
 39
 40 021604    013700  002416                   MOV       RCTSIZ,R0               ; Load count
 41 021610                                     350$:
 42 021610    005021                            CLR       (R1)+                   ; Clear location
 43 021612    005300                            DEC       R0                      ; Decrement
 44 021614    001375                            BNE       350$                    ; Continue until zero
 45
 46                                            ;           Read RCT block 0 to obtain volume info
 47 021616                                     360$:
 48 021616    004737  030626                   JSR       PC,INTCPK               ;INITIALIZE THE COMMAND PACKET
```

```
 49 021622  012737  042472  002340      MOV     #DBUFF,UADR              ;GET BEGINNING ADDRESS OF DATA BUFFER
 50 021630  012737  000000  002476      MOV     #0,RCTBLK                ;INIT THE RCT LOGICAL BLOCK TO 0 AND
 51 021636  004737  030074              JSR     PC,INITRW               ;SOME READ/WRITE PARAMETERS.
 52                                                                     ; Setup to issue MULTRD command
    021642  004737  027224              JSR     PC,MULTRD               ; Issue command to the xDA
    021646  022172                      900$                            ; Here if fatal error,
 53 021650                      370$:
 54 021650  013704  002324              MOV     HCOM,R4                 ; Load host comm area ptr
 55 021654  013705  002304              MOV     CTLRTBL,R5              ; Load controller table ptr
 56 021660  004737  032022              JSR     PC,GETVSN               ; Format volume serial number
 57
 58 021664  016546  000006              MOV     C.DRVN(R5),-(SP)                ;PUSH C.DRVN(R5) ON STACK
    021670  011546                      MOV     (R5),-(SP)              ;PUSH (R5) ON STACK
    021672  012746  014225              MOV     #DEVNAM,-(SP)           ;PUSH #DEVNAM ON STACK
    021676  013746  002074              MOV     L$LUN,-(SP)             ;PUSH L$LUN ON STACK
    021702  012746  005016              MOV     #INF110,-(SP)           ;PUSH #INF110 ON STACK
    021706  004137  033254              JSR     R1,LPNTB                        ;CALL LPNTB PRINT ROUTINE
    021712  000012                      .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
 59
 60 021714                      385$:
 61 021714  013700  003246              MOV     MDLNDX,R0               ; Load model index
 62 021720  013746  002432              MOV     UNSIZ+2,-(SP)           ;PUSH UNSIZ+2 ON STACK
    021724  013746  002430              MOV     UNSIZ,-(SP)             ;PUSH UNSIZ ON STACK
    021730  013746  002616              MOV     VSNPTR,-(SP)            ;PUSH VSNPTR ON STACK
    021734  016046  003262              MOV     MDLTEXT(R0),-(SP)               ;PUSH MDLTEXT(R0) ON STACK
    021740  012746  005165              MOV     #INF111,-(SP)           ;PUSH #INF111 ON STACK
    021744  004137  033254              JSR     R1,LPNTB                        ;CALL LPNTB PRINT ROUTINE
    021750  000012                      .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
 63
```

```
    1                                    ;       Read all RCT blocks
    2
    3 021752                       400$:
    4 021752  013704  002324               MOV     HCOM,R4                    ; Load host comm area address
    5 021756  004737  030626               JSR     PC,INTCPK                  ; Initialize command packet
    6 021762  012737  042472  002340        MOV     #DBUFF,UADR                ; Load buffer address
    7 021770  012737  000000  002476        MOV     #0,RCTBLK                  ; Reset RCT logical block number
    8 021776  012746  007002               MOV     #INF301,-(SP)              ;PUSH #INF301 ON STACK
      022002  004137  033254               JSR     R1,LPNTB                       ;CALL LPNTB PRINT ROUTINE
      022006  000002                       .WORD   ARG.CT                     ;ARGUMENT COUNT * 2
    9 022010                       420$:
   10 022010  004737  030074               JSR     PC,INITRW                  ; Initialize R/W parameters
   11 022014  063764  002476  000140        ADD     RCTBLK,HC.CPK+P.LBN(R4)    ; Add RCT logical block
   12 022022  005564  000142               ADC     HC.CPK+P.LBN+2(R4)         ; To start of RCT area
   13                                                                        ; Setup to issue MULTRD command
      022026  004737  027224               JSR     PC,MULTRD                  ; Issue command to the xDA
      022032  022172                       900$                              ; Here if fatal error,
   14 022034                       430$:
   15 022034  005237  002476               INC     RCTBLK                     ; Increment to next block
   16 022040  023737  002476  002416        CMP     RCTBLK,RCTSIZ              ; Test for all block sread
   17 022046  002760                       BLT     420$                       ; Branch if NO
   18
```

```
    1
    2                                    ;          Call STEP3 to read RCT block 0 and process replacement
    3                                    ;          recovery if necessasy.
    4
    5 022050                             500$:
    6 022050   004737   024260                     JSR    PC,STEP3            ; Call STEP3 to test for crash
    7 022054   100446                              BMI    900$                ; Branch if fatal error
    8
    9 022056   005737   002546                     TST    NXTSTEP             ; Test for recovery
   10 022062   001433                              BEQ    550$                ; Branch if NO
   11
   12 022064   005737   002150                     TST    MANRPL              ; Test for MANUAL mode
   13 022070   001004                              BNE    520$                ; Branch if YES
   14
   15 022072   005737   002152                     TST    AUTORC              ; Test for automatic recovery
   16 022076   001015                              BNE    540$                ; Branch if yes
   17 022100   000434                              BR     900$                ; Else, error exit
   18 022102                             520$:
   19 022102   012737   000001   002654            MOV    #1,REPLY            ; Else, set default reply
   20 022110   104443                              TRAP   C$GMAN
      022112   000404                              BR     10000$
      022114   002654                              .WORD  REPLY
      022116   000130                              .WORD  T$CODE
      022120   015031                              .WORD  TXT006
      022122   177777                              .WORD  -1
      022124                             10000$:
   21 022124   005737   002654                     TST    REPLY               ; Test replay
   22 022130   001420                              BEQ    900$                ; Branch if NO
   23 022132                             540$:
   24 022132   012746   010264                     MOV    #INF601,-(SP)       ;PUSH #INF601 ON STACK
      022136   004137   033254                     JSR    R1,LPNTB               ;CALL LPNTB PRINT ROUTINE
      022142   000002                              .WORD  ARG.CT              ;ARGUMENT COUNT * 2
   25 022144   004737   024156                     JSR    PC,REPLACE          ; Process possible recovery
   26 022150   022172                              900$                       ; Error exit
   27 022152                             550$:
   28 022152   004737   023160                     JSR    PC,INIPARM          ; Initialize program variables
   29
   30                                    ;          Normal exit
   31 022156                             800$:
   32 022156   062716   000002                     ADD    #2,(SP)             ; Bump past error return
   33 022162   012737   000000   002442            MOV    #0,STATUS           ; Load success
   34 022170   000207                              RTS    PC                  ; and return
   35
   36                                    ;          Error exit
   37 022172                             900$:
   38 022172   017616   000000                     MOV    @(SP),(SP)          ; Load error return
   39 022176   012737   177777   002442            MOV    #-1,STATUS          ; Load failure
   40 022204   000207                              RTS    PC                  ; and return
   41
```

```
     1                                     .SBTTL   INISWP   Initialize software parms
     2                                  ;
     3                                  ;INISWP:
     4                                  ;        Initialize the parameters below by prompting for
     5                                  ;        Operator input:
     6                                  ;
     7                                  ;        OPHELP:          Display operator help?
     8                                  ;        MANRPL:          Automatic or Manual replacement?
     9                                  ;        AUTORC:          Automatic crash recovery?
    10                                  ;        DSPRPL:          Display replacements as they occur?
    11                                  ;        DSPRCT:          Display RCT replacement descriptors?
    12                                  ;        ENBRPL:          Enable Replacements?
    13                                  ;        ENBWFE:          Enhibit write with Forced Error flag?
    14                                  ;
    15
    16 022206                          INISWP:
    17 022206  012746  004603                   MOV    #INF102,-(SP)      ;PUSH #INF102 ON STACK
       022212  004137  033254                   JSR    R1,LPNTB                 ;CALL LPNTB PRINT ROUTINE
       022216  000002                           .WORD  ARG.CT             ;ARGUMENT COUNT * 2
    18 022220  012701  022525                   MOV    #SWMSGO,R1         ; Load Question address
    19 022224  012702  002146                   MOV    #OPHELP,R2         ; Load reply address
    20 022230  012712  000000                   MOV    #0,(R2)            ; Default is "NO"
    21 022234  004737  023126                   JSR    PC,GETLOG          ; Ask operator
    22 022240  005737  002146                   TST    OPHELP             ; Test response
    23 022244  001402                           BEQ    10$                ; Branch if "NO"
    24 022246                          5$:
    25 022246  004737  032150                   JSR    PC,DSPHELP         ; Display operator help
    26 022252                          10$:
    27 022252  012737  000000  002150           MOV    #0,MANRPL          ; Default is Auto replacemet
    28 022260  012701  022553                   MOV    #SWMSG1,R1         ; Load Question address
    29 022264  012702  002654                   MOV    #REPLY,R2          ; Load reply address
    30 022270  012712  000101                   MOV    #'A,(R2)           ; Default is "AUTOMATIC"
    31 022274  004737  023070                   JSR    PC,GETCHR          ; Ask operator
    32 022300  122737  000115  002654           CMPB   #'M,REPLY          ; Test for "M"
    33 022306  001004                           BNE    20$                ; Branch if NO
    34 022310  012737  000001  002150           MOV    #1,MANRPL          ; Else, set Manual replacement
    35 022316  000430                           BR     50$                ; and continue
    36 022320                          20$:
    37 022320  012701  022613                   MOV    #SWMSG2,R1         ; Load Question address
    38 022324  012702  002152                   MOV    #AUTORC,R2         ; Load reply address
    39 022330  012712  000001                   MOV    #1,(R2)            ; Default is "YES"
    40 022334  004737  023126                   JSR    PC,GETLOG          ; Ask operator
    41 022340                          30$:
    42 022340  012701  022644                   MOV    #SWMSG3,R1         ; Load Question address
    43 022344  012702  002154                   MOV    #DSPRPL,R2         ; Load reply address
    44 022350  012712  000001                   MOV    #1,(R2)            ; Default is "YES"
    45 022354  004737  023126                   JSR    PC,GETLOG          ; Ask operator
    46 022360                          40$:
    47 022360  012701  022707                   MOV    #SWMSG4,R1         ; Load Question address
    48 022364  012702  002156                   MOV    #DSPRCT,R2         ; Load reply address
    49 022370  012712  000000                   MOV    #0,(R2)            ; Default is "NO"
    50 022374  004737  023126                   JSR    PC,GETLOG          ; Ask operator
    51 022400                          50$:
    52 022400  012701  022753                   MOV    #SWMSG5,R1         ; Load Question address
    53 022404  012702  002160                   MOV    #ENBRPL,R2         ; Load reply address
    54 022410  012712  000001                   MOV    #1,(R2)            ; Default is "YES"
    55 022414  004737  023126                   JSR    PC,GETLOG          ; Ask operator
```

```
 56 022420                              60$:
 57 022420  000410                              BR      70$                     ; Bypass this one
 58 022422  012701  022777                      MOV     #SWMSG6,R1              ; Load Question address
 59 022426  012702  002162                      MOV     #ENBERRL,R2             ; Load reply address
 60 022432  012712  000001                      MOV     #1,(R2)                 ; Default is "YES"
 61 022436  004737  023126                      JSR     PC,GETLOG               ; Ask operator
 62 022442                              70$:
 63 022442  012701  023024                      MOV     #SWMSG7,R1              ; Load Question address
 64 022446  012702  002164                      MOV     #ENBWFE,R2              ; Load reply address
 65 022452  012712  000001                      MOV     #1,(R2)                 ; Default is "YES"
 66 022456  004737  023126                      JSR     PC,GETLOG               ; Ask operator
 67 022462                              100$:
 68 022462  000207                              RTS     PC                      ; Return to caller
 69
 70                                             .NLIST  BEX
 71 022464    110    141    166 BACKMSG: .ASCIZ \Have you backed-up customer data\
 72 022525    104    151    163 SWMSG0:  .ASCIZ \Display operator Help\
 73 022553    101    165    164 SWMSG1:  .ASCIZ \Automatic or Manual replacement\
 74 022613    101    165    164 SWMSG2:  .ASCIZ \Automatic crash recovery\
 75 022644    104    151    163 SWMSG3:  .ASCIZ \Display replacements as they occur\
 76 022707    104    151    163 SWMSG4:  .ASCIZ \Display RCT replacement descriptors\
 77 022753    105    156    141 SWMSG5:  .ASCIZ \Enable replacements\
 78 022777    105    156    141 SWMSG6:  .ASCIZ \Enable error logging\
 79 023024    105    156    141 SWMSG7:  .ASCIZ \Enable write with Forced Error flag\
 80                                             .LIST   BEX
 81                                             .EVEN
 82
```

```
 1                                      .SBTTL  GETCHR  Get character from Operator
 2                                      ;
 3                                      ;GETCHR:
 4                                      ;       Gets an alphanumeric character input from operator.
 5                                      ;Inputs:
 6                                      ;       R1      .ASCIZ Prompt msg address
 7                                      ;       R2      Reply address
 8                                      ;
 9                                      ;Outputs:
10                                      ;       Prints prompt; response is placed in reply address.
11                                      ;
12                                      ;Calling Seq:
13                                      ;
14                                      ;       MOV     #PROMPT,R1
15                                      ;       MOV     #REPLY,R2
16                                      ;       JSR     PC,GETCHR
17                                      ;
18
19
20 023070                              GETCHR:
21 023070  010137  023110                MOV     R1,CHRMSG               ; Save message address
22 023074  010237  023104                MOV     R2,CHRRPLY              ; Save reply address
27 023100                              10$:
28 023100  104443                         TRAP    C$GMAN
   023102  000406                         BR      10001$
   023104  023104                         .WORD   CHRRPLY
   023106  000152                         .WORD   T$CODE
   023110  023110                         .WORD   CHRMSG
   023112  177777                         .WORD   -1
   023114  000001                         .WORD   T$LOLIM
   023116  000001                         .WORD   T$HILIM
   023120                              10001$:
29 023120                              20$:
30 023120  117700  177760                MOVB    @CHRRPLY,R0             ; Set return code
31 023124  000207                         RTS     PC                     ; and return
32
33
```

```
 1                                    .SBTTL  GETLOG  Get Logical from Operator
 2                                  ;
 3                                  ;GETLOG:
 4                                  ;     Gets an logical (Y/N) reply from operator.
 5                                  ;Inputs:
 6                                  ;     R1      .ASCIZ Prompt msg address
 7                                  ;     R2      Reply address
 8                                  ;
 9                                  ;Outputs:
10                                  ;     Prints prompt; response is placed in reply address.
11                                  ;
12                                  ;Calling Seq:
13                                  ;
14                                  ;     MOV     #PROMPT,R1
15                                  ;     MOV     #REPLY,R2
16                                  ;     JSR     PC,GETLOG
17                                  ;
18
19 023126                          GETLOG:
20 023126  010137  023146                  MOV     R1,LOGMSG           ; Save message address
21 023132  010237  023142                  MOV     R2,LOGRPLY          ; Save reply address
26 023136                          10$:
27 023136  104443                          TRAP    C$GMAN
   023140  000404                          BR      10002$
   023142  023142                          .WORD   LOGRPLY
   023144  000130                          .WORD   T$CODE
   023146  023146                          .WORD   LOGMSG
   023150  177777                          .WORD   -1
   023152                          10002$:
28 023152                          20$:
29 023152  117700  177764                  MOVB    @LOGRPLY,R0         ; Set return code
30 023156  000207                          RTS     PC                  ; and return
31
32
```

```
    1                                    .SBTTL   INIPARM Initialize program parameters
    2                                  ;
    3                                  ;INIPARM:
    4                                  ;        Initialize program variables.
    5                                  ;Inputs:
    6                                  ;        None.
    7                                  ;Outputs:
    8                                  ;        Initializes program variables.
    9                                  ;
   10                                  ;Calling Seq:
   11                                  ;
   12                                  ;        JSR      PC,INIPARM
   13                                  ;
   14
   15 023160                          INIPARM:
   16 023160   012737  000000  002574          MOV      #0,ACCESS          ; Initialize
   17 023166   012737  000000  002576          MOV      #0,ACCESS+2        ; ACCESS count
   18 023174   012737  000000  002562          MOV      #0,GLBN            ; Initialize starting
   19 023202   012737  000000  002564          MOV      #0,GLBN+2          ; group LBN
   20 023210   012737  000000  002600          MOV      #0,MINLBN          ; Reset
   21 023216   012737  000000  002602          MOV      #0,MINLBN+2        ; Minimum
   22 023224   013737  002430  002604          MOV      UNSIZ,MAXLBN       ; Reset
   23 023232   013737  002432  002606          MOV      UNSIZ+2,MAXLBN+2   ; Maximum
   24
   25 023240   012737  000000  002566          MOV      #0,GBCNT           ; Group byte count
   26 023246   012737  000001  002570          MOV      #1,GBCNT+2         ; is 128. x 512.
   27
   28 023254   012737  000000  002220          MOV      #0,KW.EL           ; Clear
   29 023262   012737  000000  002222          MOV      #0,KW.EL+2         ; Elapsed time
   30 023270   012737  000000  002466          MOV      #0,RECOVR          ; Reset recovery flag
   31 023276   012737  000000  002434          MOV      #0,PRIMRY          ; Zero PRIMARY RBN count
   32 023304   012737  000000  002436          MOV      #0,SCNDRY          ; Zero SECONDARY RBN count
   33 023312   012737  000000  002440          MOV      #0,UNUSE           ; Zero UNUSABLE RBN count
   34
   35 023320   012737  000000  002530          MOV      #0,FERCNT          ; Clear FORCED ERROR count
   36 023326   012737  000000  002532          MOV      #0,HCECNT          ; Clear HEADER COMP ERROR count
   37 023334   012737  000000  002534          MOV      #0,DSTCNT          ; Clear DATA SYNC TMO count
   38 023342   012737  000000  002536          MOV      #0,ECCCNT          ; Clear ECC ERROR count
   39 023350   012737  000000  002540          MOV      #0,BBRCNT          ; Clear BBR count
   40 023356   012737  000000  002542          MOV      #0,WFECNT          ; Clear WRITE FE count
   41
   42 023364   012737  000000  002516          MOV      #0,LBNSTAT         ; Clear LBN status
   43 023372   012737  000000  002520          MOV      #0,RPLSTAT         ; Clear Replacement status
   44 023400   012737  000000  002342          MOV      #0,CMDREF          ; Zero cmd ref number
   45
   46 023406   000207                          RTS      PC                 ; Return to caller
   47
```

```
    1                                     .SBTTL  GETLBN  Get LBN from operator
    2                                 ;GETLBN
    3                                 ;
    4                                 ;Inputs:
    5                                 ;       R1 =    Address of operator prompt
    6                                 ;       R2 =    Address of receiving long word
    7                                 ;
    8                                 ;Outputs:
    9                                 ;       Operator is prompted for LBN.  Reply is checked
   10                                 ;       against UNSIZ maximum.
   11                                 ;       LBN is converted to a 28-bit value by CNV28.
   12                                 ;
   13                                 ;Calling seq:
   14                                 ;
   15                                 ;       MOV     #PROMPT,R1
   16                                 ;       MOV     #LBN,R4
   17                                 ;       JSR     PC,GETLBN
   18                                 ;
   19
   20 023410                         GETLBN:
   21 023410  010146                         MOV     R1,-(SP)                ;;PUSH R1 ON STACK
      023412  010246                         MOV     R2,-(SP)                ;;PUSH R2 ON STACK
      023414  010346                         MOV     R3,-(SP)                ;;PUSH R3 ON STACK
      023416  010446                         MOV     R4,-(SP)                ;;PUSH R4 ON STACK
      023420  010546                         MOV     R5,-(SP)                ;;PUSH R5 ON STACK
   22 023422  010137  023456                 MOV     R1,MSGADR               ; Save message address
   23 023426                         10$:
   24 023426  012746  014231                 MOV     #CRLF,-(SP)             ;PUSH #CRLF ON STACK
      023432  004137  033254                 JSR     R1,LPNTB                        ;CALL LPNTB PRINT ROUTINE
      023436  000002                         .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
   25 023440  012737  000000  002666         MOV     #0,TTYIN                ; Flag default response
   26                                                                        ;PRINT 'Enter LBN to be replaced(D) 0 ?'
   32 023446                         20$:
   33 023446  104443                         TRAP    C$GMAN
      023450  000406                         BR      10003$
      023452  002666                         .WORD   TTYIN
      023454  000152                         .WORD   T$CODE
      023456  023456                         .WORD   MSGADR
      023460  177777                         .WORD   177777
      023462  000000                         .WORD   T$LOLIM
      023464  000011                         .WORD   T$HILIM
      023466                         10003$:
   34 023466  005737  002666                 TST     TTYIN                   ; Test response
   35 023472  001433                         BEQ     40$                     ; Branch if null
   36 023474  004737  034756                 JSR     PC,CNV28                ;CONVERT ASCIZ TO DECIMAL
   37 023500  103752                         BCS     10$                     ;BRANCH IF ERROR
   38 023502  013700  002430                 MOV     UNSIZ,R0                ;STUFF IN LO ORDER UNIT SIZE AND
   39 023506  013701  002432                 MOV     UNSIZ+2,R1              ;HI ORDER UNIT SIZE TO GET
   40                                                                        ;RCT STARTING LBN.
   41 023512  162700  000001                 SUB     #1,R0                   ;DECREMENT TO GET LAST HOST LBN AND
   42 023516  005601                         SBC     R1                      ;SUBTRACT CARRY BIT.
   43 023520  026401  000002                 CMP     2(R4),R1                ;SEE IF HI ORDER LBN WAS EXCEEDED.
   44 023524  003003                         BGT     30$                     ;BRANCH IF SO, ELSE
   45 023526  002412                         BLT     35$                     ;WITHIN LIMITS.
   46 023530  021400                         CMP     (R4),R0                 ;SEE IF LO ORDER LBN WAS EXCEEDED.
   47 023532  101410                         BLOS    35$                     ;BRANCH IF NOT, ELSE
   48 023534                         30$:                                    ;PRINT 'LIMITS - LO= 0, HI= "D32".'
```

```
    49 023534  010146                       MOV     R1,-(SP)          ;PUSH R1 ON STACK
       023536  010046                       MOV     R0,-(SP)          ;PUSH R0 ON STACK
       023540  012746   014367              MOV     #OLIMIT,-(SP)     ;PUSH #OLIMIT ON STACK
       023544  004137   033244              JSR     R1,LPNTF              ;CALL LPNTF PRINT ROUTINE
       023550  000006                       .WORD   ARG.CT            ;ARGUMENT COUNT * 2
    50 023552  000725                       BR      10$               ;TRY AGAIN
    51 023554                       35$:
    52 023554  012737   000001   002666     MOV     #1,TTYIN          ; Set SUCCESS
    53
    54 023562                       40$:
    55 023562  012605                       MOV     (SP)+,R5          ;;POP STACK INTO R5
       023564  012604                       MOV     (SP)+,R4          ;;POP STACK INTO R4
       023566  012603                       MOV     (SP)+,R3          ;;POP STACK INTO R3
       023570  012602                       MOV     (SP)+,R2          ;;POP STACK INTO R2
       023572  012601                       MOV     (SP)+,R1          ;;POP STACK INTO R1
    56 023574  013737   002666   002442     MOV     TTYIN,STATUS      ; Set condition codes for caller
    57 023602  000207                       RTS     PC                ; Return to caller
    58
```

```
                                        .SBTTL   READLBN Read suspect LBN
  1
  2                                     ;READLBN
  3                                     ;
  4                                     ;Reread suspect LBN 32 times to determine if really bad
  5                                     ;and retrieve best data.
  6                                     ;
  7                                     ;Inputs:
  8                                     ;       HCOM      Host communications area
  9                                     ;
 10                                     ;Outputs:
 11                                     ;       LBNSTAT Status of read:
 12                                     ;
 13                                     ;         TF.FER = FORCED ERROR
 14                                     ;         TF.HCE = HEADER COMP ERROR
 15                                     ;         TF.DST = DATA SYNC TMO
 16                                     ;         TF.ECC = ECC ERROR
 17                                     ;         TF.SEX = SERIOUS EXCEPTION
 18                                     ;         TF.LOG = ERRORLOG SENT
 19                                     ;         TF.BBU = BAD BLOCK UNREPORTED
 20                                     ;         TF.BBR = BAD BLOCK REPORTED
 21                                     ;         TF.BAD = DATA IS BAD
 22                                     ;
 23                                     ;       STATUS  0 or 1: success
 24                                     ;                  -1: Fatal error
 25                                     ;
 26                                     ;Calling seq:
 27                                     ;
 28                                     ;       JSR      PC,READLBN
 29                                     ;       .WORD    <error exit>
 30                                     ;
 31
 32 023604                             READLBN:
 33 023604  013704  002324                MOV     HCOM,R4              ; Load comm ptr
 34 023610  012737  000040  002556        MOV     #32.,RDCNT           ; Initialize READ count
 35 023616                             100$:
 36 023616  012737  000000  002346        MOV     #0,CMDMOD            ; Zero cmd modifiers
 37 023624  012737  000000  002516        MOV     #0,LBNSTAT           ; Zero LBN status
 38 023632                             110$:
 39 023632  004737  024772                JSR     PC,STEP4             ; Read LBN data
 40 023636  100517                         BMI     900$                 ; Branch if fatal error
 41 023640  153737  002356  002516        BISB    FLAGS,LBNSTAT        ; Set end msg flags
 42 023646  005737  002354                TST     EMSTAT               ; Test engmsg status
 43 023652  001464                         BEQ     200$                 ; Branch if SUCCESS
 44 023654                             120$:
 45 023654  013700  002354                MOV     EMSTAT,R0            ; Load endmsg status
 46 023660  052737  100000  002516        BIS     #TF.BAD,LBNSTAT      ; Set LBN data bad
 47
 48 023666  022700  000010                CMP     #ST.DAT,R0           ; Test for Forced Error
 49 023672  001004                         BNE     122$                 ; Branch if NO
 50 023674  052737  000001  002516        BIS     #TF.FER,LBNSTAT      ; Else, set FER status
 51 023702  000450                         BR      200$                 ; And write to RCT
 52 023704                             122$:
 53 023704  022700  000110                CMP     #SB.HCE,R0           ; Test for Header Compare Error
 54 023710  001004                         BNE     124$                 ; Branch if NO
 55 023712  052737  000002  002516        BIS     #TF.HCE,LBNSTAT      ; Else, set HCEstatus
 56 023720  000415                         BR      130$                 ; And continue
 57 023722                             124$:
```

```
 58 023722  022700  000150              CMP    #SB.DST,R0          ; Test for Data Sync Timeout
 59 023726  001004                      BNE    126$                ; Branch if NO
 60 023730  052737  000004  002516      BIS    #TF.DST,LBNSTAT     ; Else, set DST status
 61 023736  000406                      BR     130$                ; And continue
 62 023740                      126$:
 63 023740  022700  000350              CMP    #SB.ECC,R0          ; Test for Uncorrectable ECC
 64 023744  001003                      BNE    130$                ; Branch if NO
 65 023746  052737  000010  002516      BIS    #TF.ECC,LBNSTAT     ; Else, set ECC status
 66 023754                      130$:
 67 023754  005337  002556              DEC    RDCNT               ; Decrement count
 68 023760  001316                      BNE    100$                ; and continue until zero
 69 023762                      150$:
 70 023762  033737  000010  002516      BIT    TF.ECC,LBNSTAT      ; Test for uncorrectable ECC
 71 023770  001403                      BEQ    180$                ; Branch if NO
 72 023772  012737  000100  002346      MOV    #MD.SEC,CMDMOD      ; Else, suppress ECC
 73 024000                      180$:
 74 024000  004737  024772              JSR    PC,STEP4            ; Read LBN
 75 024004  100434                      BMI    900$                ; Branch if fatal error
 76 024006  005737  002160              TST    ENBRPL              ; Test for replacement enabled
 77 024012  001420                      BEQ    300$                ; BRanch if NO
 78 024014  004737  025136              JSR    PC,STEP5            ; Write LBN data
 79 024020  100426                      BMI    900$                ; Branch if fatal error
 80 024022  000414                      BR     300$                ; Exit
 81 024024                      200$:
 82 024024  005737  002160              TST    ENBRPL              ; Test for replacement enabled
 83 024030  001403                      BEQ    210$                ; BRanch if NO
 84 024032  004737  025136              JSR    PC,STEP5            ; Else, record data in RCT
 85 024036  100417                      BMI    900$                ; Branch if fatal error
 86 024040                      210$:
 87 024040  005737  002516              TST    LBNSTAT             ; Test status
 88 024044  001003                      BNE    300$                ; Brnach if non-zero
 89 024046                      220$:
 90 024046  005337  002556              DEC    RDCNT               ; Decrement count
 91 024052  001261                      BNE    100$                ; continue if not zero
 92 024054                      300$:
 93 024054  062716  000002              ADD    #2,(SP)             ; Bump past error return
 94 024060  013737  002516  002442      MOV    LBNSTAT,STATUS      ; Set cond codes
 95 024066  042737  100000  002442      BIC    #TF.BAD,STATUS      ; Clear MSB
 96 024074  000207                      RTS    PC                  ; Return to caller
 97 024076                      900$:
 98 024076  017616  000000              MOV    @(SP),(SP)          ; Load error return
 99 024102  012737  177777  002442      MOV    #-1,STATUS          ; Load fatal error
100 024110  000207                      RTS    PC                  ; Return to caller
101
103
```

```
      1                              .SBTTL   REPLACE Replacement routine
      2                              ;REPLACE
      3                              ;
      4                              ;This is the global bad block replacement routine, used by all tests.
      5                              ;It performs all steps used in the modified replacement algorithm
      6                              ;designed for this diagnostic.  The routine will only execute one step
      7                              ;on each call, with the next step in the algorithm placed in NXTSTEP,
      8                              ;based upon program logic.
      9                              ;
     10                              ;Entries:
     11                              ;        REPLACE - Step to execute in NXTSTEP
     12                              ;        STEPx   - Step to execute in STEP
     13                              ;Exits:
     14                              ;        PC+2    -  Fatal error
     15                              ;
     16                              ;        NXTSTEP =  x:  Continue with STEPx
     17                              ;                =  0:  Replacement complete
     18                              ;                = -1:  Replacement step error
     19                              ;
     20                              ;Steps:
     21                              ;        REPLACE - General purpose entry.
     22                              ;
     23                              ;        STEP3   - Read RCT 0 and test for recovery in progress.
     24                              ;        STEP4   - Read LBN selected for replacement.
     25                              ;        STEP5   - Write LBN image to RCT block 1.
     26                              ;        STEP6   - Update RCT block 0 to indicate PHASE 1.
     27                              ;        STEP9   - Search for available RBN for replacement.
     28                              ;        STEP10  - Update RCT block 0 to indicate PHASE 2.
     29                              ;        STEP11  - Update RCT replacement descriptor.
     30                              ;        STEP12  - Issue REPLACE command and write LBN data.
     31                              ;        STEP13  - Update RCT block 0 to indicate replacement complete.
     32                              ;        STEP15  - Update RCT descriptor to indicate usage.
     33                              ;        STEP16  - Write LBN image to replaced LBN.
     34                              ;        STEP17  - Update RCT block 0 to indicate replacement complete.
     35                              ;
     36                              ;
     37
     38 024112                      STEPTBL:
     39 024112  024250                  .WORD          STEPFTL       ; Step 0 entry
     40 024114  024250                  .WORD          STEPFTL       ; Step 1 entry
     41 024116  024250                  .WORD          STEPFTL       ; Step 2 entry
     42 024120  024260                  .WORD          STEP3         ; Step 3 entry
     43 024122  024772                  .WORD          STEP4         ; Step 4 entry
     44 024124  025136                  .WORD          STEP5         ; Step 5 entry
     45 024126  025204                  .WORD          STEP6         ; Step 6 entry
     46 024130  024250                  .WORD          STEPFTL       ; Step 7 entry
     47 024132  024250                  .WORD          STEPFTL       ; Step 8 entry
     48 024134  025350                  .WORD          STEP9         ; Step 9 entry
     49 024136  025460                  .WORD          STEP10        ; Step 10 entry
     50 024140  025610                  .WORD          STEP11        ; Step 11 entry
     51 024142  026132                  .WORD          STEP12        ; Step 12 entry
     52 024144  026530                  .WORD          STEP13        ; Step 13 entry
     53 024146  024250                  .WORD          STEPFTL       ; Step 14 entry
     54 024150  026640                  .WORD          STEP15        ; Step 15 entry
     55 024152  026760                  .WORD          STEP16        ; Step 16 entry
     56 024154  027132                  .WORD          STEP17        ; Step 17 entry
     57
```

```
 58          000044            STEPSIZ =              .-STEPTBL        ; Number of table entries
 59
 60 024156                     REPLACE:
 61
 62 024156                     10$:
 63 024156  013700  002546         MOV     NXTSTEP,R0           ; Load next STEP
 64 024162  010037  002544         MOV     R0,STEP              ; Save as current STEP
 65 024166  006300                 ASL     R0                   ; STEP x 2 is offset
 66 024170  003416                 BLE     STEPERR              ; Error if STEP <= 0
 67 024172  022700  000044         CMP     #STEPSIZ,R0          ; Compare to table max
 68 024176  003413                 BLE     STEPERR              ; Error if STEP >= STEPSIZ
 69
 70 024200  004770  024112         JSR     PC,@STEPTBL(R0)      ; Else, JSR to STEP entry
 71 024204  005737  002546         TST     NXTSTEP              ; Test return code
 72 024210  003362                 BGT     10$                  ; Branch if next STEP > zero
 73 024212  100405                 BMI     STEPERR              ; Else, error if STEP < zero
 74
 75                             ;   Normal return
 76 024214                     20$:
 77 024214  062716  000002         ADD     #2,(SP)              ; Bump past error return
 78 024220  013700  002546         MOV     NXTSTEP,R0           ; Set cond codes for caller
 79 024224  000207                 RTS     PC                   ; and return
 80
 81                             ;   Fatal error return for replacement STEPx.
 82 024226                     STEPERR:
 83 024226  012737  177777  002546 MOV     #-1,NXTSTEP          ; Load fatal status
 84 024234  017616  000000         MOV     @(SP),(SP)           ; Load error return
 85 024240  013737  002546  002442 MOV     NXTSTEP,STATUS       ; Set cond codes
 86 024246  000207                 RTS     PC                   ; Return to caller
 87
 88                             ;   Illegal STEP
 89 024250                     STEPFTL:
 90 024250  012737  177777  002546 MOV     #-1,NXTSTEP          ; Load fatal error
 91 024256  000207                 RTS     PC                   ; and return
 92
```

```
 1                                      .SBTTL  STEP3    Replacement step 3
 2                                      ;STEP3
 3                                      ;
 4                                      ;Read RCT block 0.  Test for PHASE 1 or PHASE 2 replacement in progress.
 5                                      ;If P1 set, control transferred to STEP9.  If P2 set, control transferred
 6                                      ;to STEP11.
 7                                      ;
 8                                      ;Inputs:
 9                                      ;       R4 - Address of host comm area
10                                      ;
11                                      ;Outputs:
12                                      ;       VIBUFF - Set if LBN data is valid
13                                      ;       TBUFF - Image of RCT block 0
14                                      ;
15                                      ;Externals:
16                                      ;       INTCPK, INITRW, MULTRD, COPY
17                                      ;
18                                      ;Exits:
19                                      ;       EXIT3   - DEFAULT next STEP is 4
20                                      ;                 PHASE 1 recover STEP is 9
21                                      ;                 PHASE 2 recover STEP is 11.
22                                      ;       STEPERR - Fatal error exit
23                                      ;
24
25 024260                              STEP3:
26 024260  013704  002324                      MOV     HCOM,R4                 ; Load host comm area address
27 024264  004737  030626                      JSR     PC,INTCPK               ;INITIALIZE THE COMMAND PACKET
28 024270  012737  042472  002340              MOV     #DBUFF,UADR             ;GET BEGINNING ADDRESS OF DATA BUFFER
29 024276  012737  000000  002476              MOV     #0,RCTBLK               ;INIT THE RCT LOGICAL BLOCK TO 0 AND
30 024304  004737  030074                      JSR     PC,INITRW               ;SOME READ/WRITE PARAMETERS.
31                                             ; Setup to issue MULTRD command
   024310  004737  027224                      JSR     PC,MULTRD               ; Issue command to the xDA
   024314  024250                              STEPFTL                         ; Here if fatal error.
32 024316                              1$:
33 024316  004537  034276                      JSR     R5,COPY                 ;COPY RCT LOGICAL BLOCK 0 DATA FROM
34 024322  042472                              .WORD   DBUFF                   ;DATA BUFFER TO
35 024324  043472                              .WORD   TBUFF                   ;TEMPORARY BUFFER FOR LATER USE.
36 024326  013701  002340                      MOV     UADR,R1                 ;GET THE BEGINNING OF UNIBUS ADDRESS
37
```

```
    1
    2                              ;       Process possible crash recovery
    3
    4 024332  005037  002466              CLR     RECOVR              ; Assume no recovery needed
    5 024336  032761  100000  000010      BIT     #P1,RFLAGS(R1)      ; Test for Phase 1 crash
    6 024344  001404                      BEQ     25$                 ; Branch if NO
    7 024346  012737  000001  002466      MOV     #1,RECOVR           ; Else, set for PHASE 1
    8 024354  000415                      BR      3$                  ; and continue
    9 024356                      25$:
   10 024356  032761  040000  000010      BIT     #P2,RFLAGS(R1)      ; Test for Phase 2 crash
   11 024364  001404                      BEQ     30$                 ; Branch if NO
   12 024366  012737  000002  002466      MOV     #2,RECOVR           ; Else, set for PHASE 2
   13 024374  000405                      BR      3$                  ; and continue
   14 024376                      30$:
   15 024376  012737  000000  002546      MOV     #0,NXTSTEP          ; Set done
   16 024404  000137  024770              JMP     EXIT3               ; and exit
   17 024410                      3$:
   18 024410  012737  000000  002516      MOV     #0,LBNSTAT          ; Zero LBN status
   19 024416  112737  000001  002472      MOVB    #1,VIBUFF           ;ASSUME IMAGE DATA WAS VALID,
   20 024424  016137  000014  002446      MOV     LBNLO(R1),LBN       ;RESTORE LO WORD OF LBN AND
   21 024432  016137  000016  002450      MOV     LBNLO+2(R1),LBN+2   ;HI WORD OF LBN BEING REPLACED.
   22 024440  032761  000200  000010      BIT     #FEF,RFLAGS(R1)     ;SEE IF FORCED ERROR FLAG IS SET,
   23 024446  001405                      BEQ     5$                  ;BRANCH IF NOT, ELSE
   24 024450  105037  002472              CLRB    VIBUFF              ;INDICATE IMAGE DATA IS NOT VALID
   25 024454  052737  100001  002516      BIS     #TF.BAD!TF.FER,LBNSTAT ; Set forced error indicator
   26 024462                      5$:
   27 024462  016137  000020  002452      MOV     RBNLO(R1),RBN       ;RESTORE LO WORD OF RBN AND
   28 024470  016137  000022  002454      MOV     RBNLO+2(R1),RBN+2   ;HI WORD OF THE RBN REPLACEMENT.
   29 024476                      50$:
   30 024476  012737  000004  002370      MOV     #B.PHASE,EXTMSK     ; Print phase
   31 024504  012737  013402  002552      MOV     #ERM605,BASEMSG     ; "CRASH occured during replacement..."
   32 024512  104457                      TRAP    C$ERSOFT
      024514  001135                      .WORD   605
      024516  013173                      .WORD   HDR601
      024520  015720                      .WORD   ERR601
   33
```

```
    1
    2 024522                           55$:
    3 024522  012737  000000  002520        MOV     #0,RPLSTAT              ; Zero replacement status
    4 024530  105037  002514              CLRB    MATFLG                 ;ASSUME NO MATCH FOR LBN
    5 024534  032761  020000  000010        BIT     #BRF,RFLAGS(R1)        ;SEE IF BAD RBN FLAG IS SET,
    6 024542  001450                       BEQ     7$                     ;BRANCH IF NOT, ELSE
    7 024544  112737  000001  002514        MOVB    #1,MATFLG              ;SET THE MATCHED LBN FLAG.
    8 024552  016137  000024  002462        MOV     BADRBN(R1),MATRBN      ;RESTORE MATCHED RBN LO ORDER WORD AND
    9 024560  016137  000026  002464        MOV     BADRBN+2(R1),MATRBN+2  ;RBN HI ORDER WORD.
   10
   11 024566  105037  002515              CLRB    SFLAG                  ;ASSUME PRIMARY EMPTY SEARCH STATUS
   12 024572  010446                      MOV     R4,-(SP)               ;;PUSH R4 ON STACK
      024574  010546                      MOV     R5,-(SP)               ;;PUSH R5 ON STACK
   13 024576  013703  002446              MOV     LBN,R3                 ;GET LO WORD AND
   14 024602  013704  002450              MOV     LBN+2,R4               ;HI WORD OF LBN BEING REPLACED.
   15 024606  013700  002420              MOV     TRKSIZ,RO              ;GET THE TRACK SIZE,
   16 024612  004737  034436              JSR     PC,DIVIDE              ;DIVIDE TO FIND THE TRACK
   17                                                                   ;OF LBN BEING REPLACED
   18 024616  012605                      MOV     (SP)+,R5               ;;POP STACK INTO R5
      024620  012604                      MOV     (SP)+,R4               ;;POP STACK INTO R4
   19 024622  113700  002422              MOVB    RBNS,RO                ;GET NUMBER OF RBNS PER TRACK,
   20 024626  010046                      MOV     RO,-(SP)               ;PUT THE MULTIPLER ON THE STACK
   21 024630  010346                      MOV     R3,-(SP)               ;PUT THE MULTIPLICAND ON THE STACK
   22 024632  004737  034542              JSR     PC,MULT                ;CALL THE MULTPLY ROUTINE
   23 024636  022637  002452              CMP     (SP)+,RBN              ;SEE IF LO ORDER RBN's ARE SAME,
   24 024642  001402                      BEQ     59$                    ; Branch if yes
   25 024644  005726                      TST     (SP)+                  ; Else, adjust stack
   26 024646  000403                      BR      6$                     ; And continue
   27 024650                           59$:
   28 024650  022637  002454              CMP     (SP)+,RBN+2            ;SEE IF HI ORDER RBN's ARE SAME,
   29 024654  001403                      BEQ     7$                     ;BRANCH IF SO, ELSE
   30 024656                           6$:
   31 024656  112737  000001  002515        MOVB    #1,SFLAG               ;SET SECONDARY EMPTY SEARCH STATUS.
```

```
 1
 2 024664                            7$:
 3 024664   012737  044472  002340       MOV    #IBUFF,UADR            ;GET BEGINNING ADDRESS OF IMAGE BUFFER
 4 024672   012737  000001  002476       MOV    #1,RCTBLK             ;INIT THE RCT LOGICAL BLOCK TO 1 AND
 5 024700   004737  030074               JSR    PC,INITRW             ;SOME READ/WRITE PARAMETERS.
 6 024704   063764  002476  000140       ADD    RCTBLK,HC.CPK+P.LBN(R4)  ;ADD RCT LOGICAL BLOCK TO STARTING
 7 024712   005564  000142               ADC    HC.CPK+P.LBN+2(R4)    ;HOST LBN OF THE RCT AREA.
 8                                                                    ; Setup to issue MULTRD command
   024716   004737  027224               JSR    PC,MULTRD            ; Issue command to the xDA
   024722   024250                        STEPFTL                     ; Here if fatal error,
   024724   001405                        BEQ    9$                   ; Branch if command successful.
 9 024726                            8$:
10 024726   105037  002472               CLRB   VIBUFF               ;INDICATE IMAGE DATA IS NOT VALID
11 024732   052737  100000  002516       BIS    #TF.BAD,LBNSTAT      ; Set BAD LBN status
12 024740                            9$:
13 024740   032761  100000  000010       BIT    #P1,RFLAGS(R1)       ;SEE IF CRASH OCCURED IN PHASE 1,
14 024746   001405                        BEQ    10$                  ; BRANCH IF NOT, ELSE
15 024750   012737  000011  002546       MOV    #9.,NXTSTEP          ; Next STEP for PHASE 1 recovery is 9
16 024756   000137  024770               JMP    EXIT3                ; and exit
17 024762                            10$:
18 024762   012737  000013  002546       MOV    #11.,NXTSTEP         ; Next STEP is 11.
19 024770                            EXIT3:
20 024770   000207                        RTS    PC                   ; Return to caller
21
```

```
    1                                    .SBTTL  STEP4    Replacement step 4
    2                                 ;STEP4
    3                                 ;
    4                                 ;Read LBN to be replaced.
    5                                 ;
    6                                 ;Inputs:
    7                                 ;      R4 - Address of host comm area
    8                                 ;      LBN - Address of LBN to read
    9                                 ;
   10                                 ;Outputs:
   11                                 ;      IBUFF - Contains LBN image read
   12                                 ;      VIBUFF - Set if LBN data is valid
   13                                 ;
   14                                 ;Externals:
   15                                 ;      INITRW, CLRBUF, ISSUE
   16                                 ;
   17                                 ;Exits:
   18                                 ;      LBNERR  - LBN error exit
   19                                 ;      STEPERR - Fatal error exit
   20                                 ;
   21
   22 024772                         STEP4:
   23 024772  013704  002324            MOV    HCOM,R4                ; Load host comm area address
   24 024776  112737  000001  002472    MOVB   #1,VIBUFF              ; Assume image data is valid
   25 025004  012737  044472  002340    MOV    #IBUFF,UADR            ;GET BEGINNING ADDRESS OF IMAGE BUFFER
   26 025012  004737  030074            JSR    PC,INITRW              ;INIT SOME READ/WRITE PARAMETERS AND
   27 025016  013764  002446  000140    MOV    LBN,HC.CPK+P.LBN(R4)   ;GET THE REPLACED LBN ADDRESS.
   28 025024  013764  002450  000142    MOV    LBN+2,HC.CPK+P.LBN+2(R4)
   29 025032  004737  034240            JSR    PC,CLRBUF              ;CLEAR THE BUFFER BEFORE READING
   30                                                                 ; Setup to issue OP.RD command
      025036  012700  000041            MOV    #OP.RD,R0              ; Load MSCP opcode
      025042  004737  030204            JSR    PC,ISSUE              ; Issue command to the xDA
      025046  024250                    STEPFTL                      ; Here if fatal error,
      025050  001426                    BEQ    EXIT4                 ; Branch if command successful.
   31 025052                         20$:
   32 025052  112737  000000  002472    MOVB   #0,VIBUFF             ; Set image data not valid
   33 025060  013700  002354            MOV    EMSTAT,R0             ; Reload emdmsg status
   34 025064  042700  177740            BIC    #^CST.MSK,R0         ; Clear any subcodes
   35 025070  022700  000010            CMP    #ST.DAT,R0           ; Test for Data error
   36 025074  001414                    BEQ    EXIT4                ; Branch if YES
   37
   38 025076  012737  012137  002552    MOV    #ERM301,BASEMSG      ; "Failed READ..."
   39 025104  012737  000140  002370    MOV    #B.MSCP,EXTMSK       ; Flag all MSCP fields
   40 025112  104456                    TRAP   C$ERHRD
      025114  000455                    .WORD  301
      025116  012043                    .WORD  HDR301
      025120  015504                    .WORD  ERR301
   41 025122  000137  024250            JMP    STEPFTL              ; Fatal error exit
   42 025126                         EXIT4:
   43 025126  012737  000005  002546    MOV    #5,NXTSTEP           ; Next STEP is 5.
   44 025134  000207                    RTS    PC                   ; Return to caller
   45
```

```
  1                              .SBTTL  STEP5    Replacement step 5
  2                              ;STEP5
  3                              ;
  4                              ;Write LBN image to RCT block 1.
  5                              ;
  6                              ;Inputs:
  7                              ;       R4 - Address of host comm area
  8                              ;       UADR - Contains address of LBN data to write
  9                              ;
 10                              ;Outputs:
 11                              ;       LBN data written to RCT block 1
 12                              ;
 13                              ;Externals:
 14                              ;       INITRW, MULTWR
 15                              ;
 16                              ;Exits:
 17                              ;       EXIT4   - Successful exit; NXTSTEP is 5.
 18                              ;       STEPERR - Fatal Error exit
 19                              ;
 20
 21 025136                      STEP5:
 22 025136  013704  002324          MOV     HCOM,R4                     ; Load host comm area address
 23 025142  012737  000001  002476  MOV     #1,RCTBLK                   ;INIT THE RCT LOGICAL BLOCK TO 1 AND
 24 025150  004737  030074          JSR     PC,INITRW                   ;SOME READ/WRITE PARAMETERS.
 25 025154  063764  002476  000140  ADD     RCTBLK,HC.CPK+P.LBN(R4)     ;ADD RCT LOGICAL BLOCK TO STARTING
 26 025162  005564  000142          ADC     HC.CPK+P.LBN+2(R4)          ;HOST LBN OF THE RCT AREA.
 27                                                                     ; Setup to issue MULTWR command
    025166  004737  027514          JSR     PC,MULTWR                   ; Issue command to the xDA
    025172  024250                  STEPFTL                             ; Here if fatal error.
 28 025174                      EXIT5:
 29 025174  012737  000006  002546  MOV     #6,NXTSTEP                  ; Next step is 6.
 30 025202  000207              !   RTS     PC                          ; Return
 31
```

```
      1                              .SBTTL  STEP6    Replacement step 6
      2                              ;STEP6
      3                              ;
      4                              ;Read RCT block 0 into working buffer.  Update flags to indicate PHASE 1
      5                              ;replacement in progress.  Rewrite RCT block 0.  If MULTIWRITE fails,
      6                              ;control is transferred to STEP17.
      7                              ;
      8                              ;Inputs:
      9                              ;       R4 - Address of host comm area
     10                              ;       VIBUFF - Set if LBN data is valid
     11                              ;
     12                              ;Outputs:
     13                              ;       TBUFF - Contains RCT block 0 image
     14                              ;
     15                              ;Externals:
     16                              ;       INITRW, MULTRD, MULTWR, COPY
     17                              ;
     18                              ;Exits:
     19                              ;       RCT MULTIREAD error exit, next STEP is 17
     20                              ;       STEPERR - Fatal error exit
     21                              ;
     22
     23 025204                       STEP6:
     24 025204  013704  002324          MOV     HCOM,R4              ; Load host comm area address
     25 025210  012737  042472  002340  MOV     #DBUFF,UADR          ;GET BEGINNING ADDRESS OF DATA BUFFER
     26 025216  012737  000000  002476  MOV     #0,RCTBLK            ;INIT THE RCT LOGICAL BLOCK TO 0 AND
     27 025224  004737  030074          JSR     PC,INITRW            ;SOME READ/WRITE PARAMETERS.
     28                                                              ; Setup to issue MULTRD command
        025230  004737  027224          JSR     PC,MULTRD            ; Issue command to the xDA
        025234  024250                  STEPFTL                      ; Here if fatal error.
     29 025236                       1$:
     30 025236  004537  034276          JSR     R5,COPY              ;COPY RCT LOGICAL BLOCK 0 DATA FROM
     31 025242  042472                  .WORD   DBUFF                ;DATA BUFFER TO
     32 025244  043472                  .WORD   TBUFF                ;TEMPORARY BUFFER FOR LATER USE.
     33
     34 025246  013701  002340          MOV     UADR,R1              ;GET THE BEGINNING OF UNIBUS ADDRESS
     35 025252  013761  002446  000014  MOV     LBN,LBNLO(R1)        ;STORE LO WORD OF LBN AND
     36 025260  013761  002450  000016  MOV     LBN+2,LBNLO+2(R1)    ;HI WORD OF LBN BEING REPLACED.
     37 025266  012761  100000  000010  MOV     #P1,RFLAGS(R1)       ;SET P1 FLAG IN REPLACEMENT FLAG WORD
     38 025274  005737  002516          TST     LBNSTAT              ; Test for Valid LBN DATA
     39 025300  100003                  BPL     3$                   ;BRANCH IF SO, ELSE
     40 025302  052761  000200  000010  BIS     #FEF,RFLAGS(R1)      ;SET FORCED ERROR FLAG.
     41 025310                       3$:
     42 025310  012737  000000  002476  MOV     #0,RCTBLK            ;INIT THE RCT LOGICAL BLOCK TO 0 AND
     43 025316  004737  030074          JSR     PC,INITRW            ;SOME READ/WRITE PARAMETERS.
     44                                                              ; Setup to issue MULTWR command
        025322  004737  027514          JSR     PC,MULTWR            ; Issue command to the xDA
        025326  024250                  STEPFTL                      ; Here if fatal error.
     45 025330                       4$:
     46 025330  004537  034276          JSR     R5,COPY              ;COPY RCT LOGICAL BLOCK 0 DATA FROM
     47 025334  042472                  .WORD   DBUFF                ;DATA BUFFER TO
     48 025336  043472                  .WORD   TBUFF                ;TEMPORARY BUFFER FOR LATER USE.
     49 025340                       EXIT6:
     50 025340  012737  000011  002546  MOV     #9.,NXTSTEP          ; Load next STEP is 9.
     51 025346  000207                  RTS     PC                   ; Return
     52
```

```
  1                                          .SBTTL   STEP9    Replacement step 9
  2                                          ;STEP9
  3                                          ;
  4                                          ;Search for available RBN for replacement.
  5                                          ;
  6                                          ;Inputs:
  7                                          ;      R4 - Address of host comm area
  8                                          ;      DBUFF - Scratch buffer
  9                                          ;
 10                                          ;Outputs:
 11                                          ;      RBN - Replacement RBN
 12                                          ;      SFLAG - Search status
 13                                          ;      MATRBN - Match RBN
 14                                          ;      MATFLG - Set if LBNs matched
 15                                          ;
 16                                          ;Externals:
 17                                          ;      SEARCH
 18                                          ;
 19                                          ;Exits:
 20                                          ;      STEP16  - Search error exit
 21                                          ;      STEPERR - Fatal error exit
 22                                          ;
 23
 24 025350                                   STEP9:
 25 025350  013704  002324                          MOV      HCOM,R4          ; Load host comm area address
 26 025354  004737  030626                          JSR      PC,INTCPK        ;INITIALIZE THE COMMAND PACKET
 27 025360  012737  042472  002340                  MOV      #DBUFF,UADR      ;GET BEGINNING ADDRESS OF DATA BUFFER
 28 025366  004737  031400                          JSR      PC,SEARCH        ;CALL SEARCH ROUTINE
 29 025372  100004                                  BPL      1$               ; Branch if unsuccessful
 30 025374  012737  177777  002546                  MOV      #-1,NXTSTEP      ; Set fatal error
 31 025402  000207                                  RTS      PC               ; and return
 32 025404                                   1$:
 33 025404  123727  002515  000002                  CMPB     SFLAG,#2         ;SEE IF RCT TABLE WAS FULL.
 34 025412  001016                                  BNE      EXIT9            ;BRANCH IF NOT, ELSE
 35 025414                                   10$:
 36 025414  012737  013472  002552                  MOV      #ERM606,BASEMSG  ; "Full RCT table detected..."
 37 025422  012737  000000  002370                  MOV      #0,EXTMSK        ; Zero error mask
 38 025430  104456                                  TRAP     C$ERHRD
    025432  001136                                  .WORD    606
    025434  013173                                  .WORD    HDR601
    025436  015720                                  .WORD    ERR601
 39 025440  012737  000020  002546                  MOV      #16.,NXTSTEP     ; Next STEP is 16.
 40 025446  000207                                  RTS      PC               ; Return
 41 025450                                   EXIT9:
 42 025450  012737  000012  002546                  MOV      #10.,NXTSTEP     ; Next STEP is 10.
 43 025456  000207                                  RTS      PC               ; Return
 44
```

```
 1                              .SBTTL   STEP10  Replacement step 10
 2                              ;STEP10
 3                              ;
 4                              ;Update RCT block 0 from working buffer to indicate PHASE 2 of
 5                              ;replacement.
 6                              ;
 7                              ;Inputs:
 8                              ;        R4 - Address of host comm area
 9                              ;        TBUFF - Working copy of RCT block 0
10                              ;
11                              ;        RBN - Replacement RBN
12                              ;        SFLAG - Search status
13                              ;        MATRBN - Match RBN
14                              ;        MATFLG - Set if LBNs matched
15                              ;
16                              ;Outputs:
17                              ;        RCT block 0 RBN - Replacement RBN
18                              ;        RCT block 0 BADRBN - Bad RBN (if any)
19                              ;
20                              ;Externals:
21                              ;        INITRW, MULTWR, COPY
22                              ;
23                              ;Exits:
24                              ;        STEP11  - Successful exit
25                              ;        STEP16  - RCT MULTIWRITE error exit
26                              ;        STEPERR - Fatal error exit
27                              ;
28
29 025460                      STEP10:
30 025460   004537  034276         JSR      R5,COPY                  ;COPY RCT LOGICAL BLOCK 0 DATA FROM
31 025464   043472                 .WORD    TBUFF                    ;TEMPORARY BUFFER TO
32 025466   042472                 .WORD    DBUFF                    ;DATA BUFFER.
33
34 025470   013704  002324         MOV      HCOM,R4                  ; Load host comm area address
35 025474   013701  002340         MOV      UADR,R1                  ;GET THE BEGINNING OF UNIBUS ADDRESS
36 025500   013761  002452  000020 MOV      RBN,RBNLO(R1)            ;STORE LO WORD OF RBN AND
37 025506   013761  002454  000022 MOV      RBN+2,RBNLO+2(R1)        ;HI WORD OF THE RBN REPLACEMENT.
38 025514   052761  040000  000010 BIS      #P2,RFLAGS(R1)           ;SET PHASE 2 FLAG IN REPLACEMENT WORD
39 025522   105737  002514         TSTB     MATFLG                   ;SEE IF LBN'S MATCHED DURING SEARCH.
40 025526   001411                 BEQ      1$                       ;BRANCH IF NOT, ELSE
41 025530   052761  020000  000010 BIS      #BRF,RFLAGS(R1)          ;SET BAD RBN FLAG IN REPLACEMENT WORD.
42 025536   013761  002462  000024 MOV      MATRBN,BADRBN(R1)        ;STORE LO WORD OF RBN AND
43 025544   013761  002464  000026 MOV      MATRBN+2,BADRBN+2(R1)    ;HI WORD OF THE BAD RBN.
44 025552                      1$:
45 025552   042761  100000  000010 BIC      #P1,RFLAGS(R1)           ;CLEAR PHASE 1 FLAG IN REPLACEMENT WORD.
46 025560   012737  000000  002476 MOV      #0,RCTBLK                ;INIT THE RCT LOGICAL BLOCK TO 0 AND
47 025566   004737  030074         JSR      PC,INITRW                ;SOME READ/WRITE PARAMETERS.
48                                                                   ; Setup to issue MULTWR command
   025572   004737  027514         JSR      PC,MULTWR                ; Issue command to the xDA
   025576   024250                 STEPFTL                          ; Here if fatal error.
49 025600                      EXIT10:
50 025600   012737  000013  002546 MOV      #11.,NXTSTEP             ; Next STEP is 11.
51 025606   000207                 RTS      PC                       ; Return
52
```

```
   1                                    .SBTTL  STEP11  Replacement step 11
   2                                   ;STEP11
   3                                   ;
   4                                   ;Compute RCT block number and offset for descriptor.
   5                                   ;Read RCT, update descriptor for new RBN and rewrite.
   6                                   ;Update RCT descriptor for bad RBN, if appropriate.
   7                                   ;
   8                                   ;Inputs:
   9                                   ;       R4 - Address of host comm area
  10                                   ;       SFLAG - search status
  11                                   ;       DBUFF - Scratch buffer
  12                                   ;
  13                                   ;Outputs:
  14                                   ;
  15                                   ;Externals:
  16                                   ;       CALBLK, INITRW, MULTRD, MULTWR, COPY
  17                                   ;
  18                                   ;Exits:
  19                                   ;       STEP12  - Successful exit
  20                                   ;       STEP16  - RCT MULTIREAD error exit
  21                                   ;       STEP15  - RCT MULTIWRITE error exit
  22                                   ;       STEPERR - Fatal error exit
  23                                   ;
  24
  25 025610                           STEP11:
  26 025610  013704  002324              MOV     HCOM,R4                 ; Load host comm area address
  27 025614  105037  002473              CLRB    VBUFF1                  ;ASSUME BUFFER 1 DATA AND
  28 025620  105037  002474              CLRB    VBUFF2                  ;BUFFER 2 DATA ARE NOT VALID.
  29 025624  013746  002454              MOV     RBN+2,-(SP)             ;;PUSH RBN+2 ON STACK
     025630  013746  002452              MOV     RBN,-(SP)               ;;PUSH RBN ON STACK
  30 025634  004737  031332              JSR     PC,CALBLK               ;CALCULATE THE RCT LOGICAL BLOCK AND
  31                                                                     ;THE RBN OFFSET WITHIN THE BLOCK.
  32 025640  011600                      MOV     (SP),R0                 ;GET OFFSET WITHIN RCT LOGICAL BLOCK
  33 025642  012637  002502              MOV     (SP)+,RCTOFF            ;;POP STACK INTO RCTOFF
  34 025646  062716  000002              ADD     #2,(SP)                 ;OFFSET TO THE BEGINNING OF THE RCT
  35                                                                     ;DESCRIPTOR BLOCKS.
  36 025652  012637  002476              MOV     (SP)+,RCTBLK            ;;POP STACK INTO RCTBLK
  37 025656  013701  002340              MOV     UADR,R1                 ;GET BEGINNING ADDRESS OF BUFFER
  38 025662  006300                      ASL     R0                      ;MULTPLY OFFSET
  39 025664  006300                      ASL     R0                      ;BY 4
  40 025666  060001                      ADD     R0,R1                   ;FIND RBN POSITION IN DESCRIPTOR TABLE
  41
  42 025670  004737  030074              JSR     PC,INITRW               ;SOME READ/WRITE PARAMETERS
  43 025674  063764  002476  000140      ADD     RCTBLK,HC.CPK+P.LBN(R4) ;ADD THE RCT LOGICAL BLOCK TO STARTING
  44 025702  005564  000142              ADC     HC.CPK+P.LBN+2(R4)      ;HOST LBN OF THE RCT AREA.
  45                                                                     ; Setup to issue MULTRD command
     025706  004737  027224              JSR     PC,MULTRD               ; Issue command to the xDA
     025712  024250                      STEPFTL                         ; Here if fatal error.
  46 025714                           2$:
  47 025714  112737  000001  002473      MOVB    #1,VBUFF1               ;SET VALID BUFFER 1 DATA FLAG
  48 025722  004537  034276              JSR     R5,COPY                 ;COPY RCT LOGICAL BLOCK DATA FROM
  49 025726  042472                      .WORD   DBUFF                   ;DATA BUFFER TO
  50 025730  045472                      .WORD   BUFF1                   ;BUFFER ONE.
  51
  52 025732  013721  002446              MOV     LBN,(R1)+               ;STORE LO WORD LBN AND
  53 025736  013711  002450              MOV     LBN+2,(R1)              ;HI WORD LBN IN DESCRIPTOR ENTRY.
  54 025742  052711  020000              BIS     #BIT13,(R1)             ;ASSUME PRIMARY RBN IN REPLACEMENT CODE.
```

```
55 025746  105737  002515              TSTB    SFLAG                      ;SEE IF THIS IS A PRIMARY REPLACEMENT,
56 025752  001402                       BEQ     3$                         ;BRANCH IF SO, ELSE
57 025754  052711  010000              BIS     #BIT12,(R1)                ;SET SECONDARY RBN IN REPLACEMENT CODE.
58 025760                      3$:
59                                                                         ; Setup to issue MULTWR command
   025760  004737  027514              JSR     PC,MULTWR                  ; Issue command to the xDA
   025764  024250                       STEPFTL                            ; Here if fatal error,
60 025766                      5$:
61 025766  105737  002514              TSTB    MATFLG                     ;SEE IF LBN'S MATCHED DURING SEARCH,
62 025772  001002                       BNE     55$                        ; Branch if match
63 025774  000137  026122              JMP     EXIT11                     ; Else, exit step
64 026000                      55$:
65 026000  013746  002464              MOV     MATRBN+2,-(SP)             ;;PUSH MATRBN+2 ON STACK
   026004  013746  002462              MOV     MATRBN,-(SP)               ;;PUSH MATRBN ON STACK
66 026010  004737  031332              JSR     PC,CALBLK                  ;CALCULATE THE RCT LOGICAL BLOCK AND
67                                                                         ;THE RBN OFFSET WITHIN THE BLOCK.
68 026014  011600                       MOV     (SP),RO                    ;GET OFFSET WITHIN RCT LOGICAL BLOCK
69 026016  012637  002504              MOV     (SP)+,MATOFF               ;;POP STACK INTO MATOFF
70 026022  062716  000002              ADD     #2,(SP)                    ;OFFSET TO THE BEGINNING OF THE RCT
71                                                                         ;DESCRIPTOR BLOCKS.
72 026026  012637  002500              MOV     (SP)+,MATBLK               ;;POP STACK INTO MATBLK
73 026032  013701  002340              MOV     UADR,R1                    ;GET BEGINNING ADDRESS OF BUFFER
74 026036  006300                       ASL     RO                         ;MULTPLY OFFSET
75 026040  006300                       ASL     RO                         ;BY 4
76 026042  060001                       ADD     RO,R1                      ;FIND RBN POSITION IN DESCRIPTOR TABLE
77                                                                         ;SET UNUSABLE RBN IN REPLACEMENT CODE.
78 026044  004737  030074              JSR     PC,INITRW                  ;SOME READ/WRITE PARAMETERS
79 026050  063764  002500  000140     ADD     MATBLK,HC.CPK+P.LBN(R4)    ;ADD THE RCT LOGICAL BLOCK TO STARTING
80 026056  005564  000142              ADC     HC.CPK+P.LBN+2(R4)         ;HOST LBN OF THE RCT AREA.
81                                                                         ; Setup to issue MULTRD command
   026062  004737  027224              JSR     PC,MULTRD                  ; Issue command to the xDA
   026066  024250                       STEPFTL                            ; Here if fatal error.
82 026070                      7$:
83 025070  112737  000001  002474     MOVB    #1,VBUFF2                  ;SET VALID BUFFER 2 DATA FLAG
84 026076  004537  034276              JSR     R5,COPY                    ;COPY RCT LOGICAL BLOCK DATA FROM
85 026102  042472                       .WORD   DBUFF                      ;DATA BUFFER TO
86 026104  046472                       .WORD   BUFF2                      ;BUFFER TWO.
87
88 026106  005021                       CLR     (R1)+                      ;STORE LO ORDER LBN AND
89 026110  012711  040000              MOV     #BIT14,(R1)                ;HI ORDER LBN AND UNUSABLE CODE IN
90 026114                      8$:                                         ;DESCRIPTOR ENTRY.
91                                                                         ; Setup to issue MULTWR command
   026114  004737  027514              JSR     PC,MULTWR                  ; Issue command to the xDA
   026120  024250                       STEPFTL                            ; Here if fatal error.
92
93 026122                      EXIT11:
94 026122  012737  000014  002546     MOV     #12.,NXTSTEP               ; Next STEP is 12.
95 026130  000207                       RTS     PC                         ; and return
96
```

```
     1                                      .SBTTL  STEP12  Replacement step 12
     2                                      ;STEP12
     3                                      ;
     4                                      ;Issue REPLACE command to controller.  Rewrite LBN data using
     5                                      ;WRITE-COMPARE modifier and FORCED ERROR if appropriate.
     6                                      ;
     7                                      ;Inputs:
     8                                      ;       R4 - Address of host comm area
     9                                      ;       LBN - LBN to replace
    10                                      ;       RBN _ Replacement RBN
    11                                      ;       SFLAG - Search status
    12                                      ;       VIBUFF - Valid LBN daya flag
    13                                      ;       RCT block 1 - LBN image
    14                                      ;
    15                                      ;Outputs:
    16                                      ;
    17                                      ;Externals:
    18                                      ;       INTCPK, INITRW, ISSUE
    19                                      ;
    20                                      ;Exits:
    21                                      ;       STEP15  - REPLACE error exit
    22                                      ;       STEP9   - LBN verification error exit
    23                                      ;       STEP13  - Successful exit
    24                                      ;       STEPERR - Fatal error exit
    25                                      ;
    26
    27 026132                      STEP12:
    28 026132  013704  002324              MOV     HCOM,R4                 ; Load host comm area address
    29 026136  004737  030626              JSR     PC,INTCPK               ;INITIALIZE THE COMMAND PACKET
    30 026142  013764  002452  000120      MOV     RBN,HC.CPK+P.RBN(R4)    ;STUFF IN LO ORDER AND
    31 026150  013764  002454  000122      MOV     RBN+2,HC.CPK+P.RBN+2(R4) ;HI ORDER RBN TO REPLACE LBN.
    32 026156  013764  002446  000140      MOV     LBN,HC.CPK+P.LBN(R4)    ;STUFF IN LO ORDER AND
    33 026164  013764  002450  000142      MOV     LBN+2,HC.CPK+P.LBN+2(R4) ;HI ORDER LBN BEING REPLACED.
    34 026172  105737  002515              TSTB    SFLAG                   ;SEE IF THIS IS A PRIMARY REPLACEMENT.
    35 026176  001003                      BNE     1$                      ;BRANCH IF NOT, ELSE
    36 026200  012737  000001  002346      MOV     #MD.PRI,CMDMOD          ;PRIMARY REPLACEMENT MODIFIER IN PACKET
    37 026206                      1$:
    38                                                                     ; Setup to issue OP.RPL command
       026206  012700  000024              MOV     #OP.RPL,R0              ; Load MSCP opcode
       026212  004737  030204              JSR     PC,ISSUE               ; Issue command to the xDA
       026216  024250                      STEPFTL                        ; Here if fatal error.
       026220  001414                      BEQ     3$                     ; Branch if command successful.
    39
    40 026222  012737  012313  002552      MOV     #ERM304,BASEMSG        ; "Failed REPLACE..."
    41 026230  012737  000140  002370      MOV     #B.MSCP,EXTMSK         ; Flag all MSCP fields
    42 026236  104456                      TRAP    C$ERHRD
       026240  000460                      .WORD   304
       026242  012043                      .WORD   HDR301
       026244  015504                      .WORD   ERR301
    43 026246  000137  024250              JMP     STEPFTL                ; Fatal exit
    44 026252                      3$:
    45 026252  012737  044472  002340      MOV     #IBUFF,UADR            ;GET BEGINNING ADDRESS OF IMAGE BUFFER
    46 026260  012737  040000  002346      MOV     #MD.CMP,CMDMOD         ;STUFF COMPARE MODIFIER IN PACKET
    47
    48 026266  005737  002516              TST     LBNSTAT                ; Test for valid data
    49 026272  100011                      BPL     4$                     ; Branch if no
    50 026274  005737  002164              TST     ENBWFE                 ; Test for WFE enabled
```

```
 51 026300  001406                          BEQ     4$                    ; Branch if NO
 52
 53 026302  052737  010000  002346          BIS     #MD.ERR.CMDMOD        ; Set FORCED ERROR modifier
 54 026310  052737  000400  002516          BIS     #TF.WFE.LBNSTAT       ; and WFE status
 55 C26316                           4$:
 56 026316  004737  030074              JSR     PC,INITRW             ;INIT SOME READ/WRITE PARAMETERS AND
 57 026322  013764  002446  000140      MOV     LBN,HC.CPK+P.LBN(R4)  ;GET THE REPLACED LBN ADDRESS.
 58 026330  013764  002450  000142      MOV     LBN+2,HC.CPK+P.LBN+2(R4)
 59                                                                   ; Setup to issue OP.WR command
    026336  012700  000042              MOV     #OP.WR,R0             ; Load MSCP opcode
    026342  004737  030204              JSR     PC,ISSUE             ; Issue command to the xDA
    026346  024250                      STEPFTL                      ; Here if fatal error,
    026350  001401                      BEQ     5$                   ; Branch if command successful,
    026352  003005                      BGT     50$                  ; Branch if command unsuccessful
 60
 61                              ;      Success; test for BBR
 62 026354                       5$:
 63 026354  132737  000200  002356      BITB    #EF.BBR,FLAGS         ; Test for BBR
 64 026362  001456                      BEQ     EXIT12                ; Branch if not SET
 65 026364  000451                      BR      6$                    ; Else, get new RBN
 66
 67                              ;      Possible error
 68 026366                       50$:
 69 026366  013700  002354          MOV     EMSTAT,R0             ; Reload emdmsg status
 70 026372  042700  177740          BIC     #↑CST.MSK,R0          ; Clear any subcodes
 71 026376  022700  000010          CMP     #ST.DAT,R0            ; Test for Data error
 72 026402  001414                  BEQ     52$                   ; Branch if YES
 73
 74 026404  012737  012202  002552      MOV     #ERM302,BASEMSG       ; "Failed WRITE..."
 75 026412  012737  000140  002370      MOV     #B.MSCP,EXTMSK        ; Flag all MSCP fields
 76 026420  104456                      TRAP    C$ERHRD
    026422  000456                      .WORD   302
    026424  012043                      .WORD   HDR301
    026426  015504                      .WORD   ERR301
 77 026430  000137  024250              JMP     STEPFTL               ; Fatal error exit
 78 026434                       52$:
 79 026434  005737  002516              TST     LBNSTAT               ; Test LBN image data valid
 80 026440  100004                      BPL     55$                   ; Error if valid
 81 026442  022737  000010  002354      CMP     #ST.DAT,EMSTAT        ; Test for FER
 82 026450  001423                      BEQ     EXIT12                ; OK if yes
 83 026452                       55$:                                 ; Report program error
 84 026452  012737  012361  002552      MOV     #ERM305,BASEMSG       ; "Failed verification..."
 85 026460  012737  000140  002370      MOV     #B.MSCP,EXTMSK        ; Flag all MSCP fields
 86 026466  104456                      TRAP    C$ERHRD
    026470  000461                      .WORD   305
    026472  012043                      .WORD   HDR301
    026474  015504                      .WORD   ERR301
 87 026476  000404                      BR      6$                    ; Branch if yes
 88 026500                       60$:
 89 026500  012737  177777  002546      MOV     #-1,NXTSTEP           ; Else, Fatal error
 90 026506  000207                      RTS     PC                    ; Return
 91 026510                       6$:
 92 026510  012737  000011  002546      MOV     #9.,NXTSTEP           ; Next STEP is 9.
 93 026516  000207                      RTS     PC                    ; and return
 94 026520                       EXIT12:
 95 026520  012737  000015  002546      MOV     #13.,NXTSTEP          ; Next STEP is 13.
 96 026526  000207                      RTS     PC                    ; Return
```

97

```
     1                                          .SBTTL   STEP13   Replacement step 13
     2                                          ;STEP13
     3                                          ;
     4                                          ;Update RCT block 0 to indicate that replacement is complete.
     5                                          ;
     6                                          ;Inputs:
     7                                          ;        R4 - Address of host comm area
     8                                          ;        TBUFF - Working copy of RCT block 0
     9                                          ;        RECOVR - Replacement recovery flag
    10                                          ;
    11                                          ;Outputs:
    12                                          ;
    13                                          ;Externals:
    14                                          ;        INTCPK, INITRW, MULTWR
    15                                          ;
    16                                          ;Exits:
    17                                          ;        EXIT13  - Successful exit
    18                                          ;        STEP17  - RCT MULTIWRITE error exit
    19                                          ;        STEPERR - Fatal error exit
    20                                          ;
    21
    22 026530                          STEP13:
    23 026530  013704  002324                  MOV      HCOM,R4                 ; Load host comm area address
    24 026534  004737  030626                  JSR      PC,INTCPK               ;INITIALIZE THE COMMAND PACKET
    25 026540  012701  043472                  MOV      #TBUFF,R1               ;GET ADDRESS OF TEMP buffer and
    26 026544  010137  002340                  MOV      R1,UADR                 ;CHANGE THE UNIBUS ADDRESS FOR WRITE.
    27 026550  062701  000010                  ADD      #RFLAGS,R1              ;SET OFFSET TO REPLACEMENT FLAG WORD
    28 026554  005021                          CLR      (R1)+                   ;CLEAR REPLACEMENT WORD (#4) AND
    29 026556  005721                          TST      (R1)+                   ;SKIP WORD #5.
    30 026560  012700  000006                  MOV      #<CACHID-LBNLO/2>,R0    ;GET NUMBER OF ENTRIES TO CLEAR
    31 026564  005021                 1$:      CLR      (R1)+                   ;CLEAR THAT LOCATION
    32 026566  005300                          DEC      R0                      ;DECR REG. AND SEE IF DONE CLEARING,
    33 026570  001375                          BNE      1$                      ;BRANCH IF NOT, ELSE
    34
    35 026572  012737  000000  002476          MOV      #0,RCTBLK               ;INIT THE RCT LOGICAL BLOCK TO 0 AND
    36 026600  004737  030074                  JSR      PC,INITRW               ;SOME READ/WRITE PARAMETERS.
    37                                                                          ; Setup to issue MULTWR command
       026604  004737  027513                  JSR      PC,MULTWR               ; Issue command to the xDA
       026610  024250                          STEPFTL                          ; Here if fatal error.
    38
    39 026612                          EXIT13:
    40 026612  012737  000000  002546          MOV      #0,NXTSTEP              ; NO next STEP
    41 026620  012737  000001  002520          MOV      #1,RPLSTAT              ; Set replacement true
    42
    43 026626  004737  020002                  JSR      PC,TYPSTAT              ; Type replacement status
    44 026632  105037  002515                  CLRB     SFLAG                   ; Clear search flag
    45 026636  000207                          RTS      PC                      ; Return
    46
```

```
    1                                  .SBTTL  STEP15  Replacement step 15
    2                                  ;STEP15
    3                                  ;
    4                                  ;Rewrite updated RCT descriptor to indicate RBN usage.
    5                                  ;
    6                                  ;Inputs:
    7                                  ;       R4 - Address of host comm area
    8                                  ;       BUFF1 - RCT descriptor block
    9                                  ;       BUFF2 - RCT descriptor block
   10                                  ;       VBUFF2 - BUFF2 data is valid
   11                                  ;       VBUFF1 - BUFF1 data is valid
   12                                  ;
   13                                  ;Outputs:
   14                                  ;
   15                                  ;Externals:
   16                                  ;       INTCPK, INITRW, MULTWR
   17                                  ;
   18                                  ;Exits:
   19                                  ;       STEP16  - Successful exit
   20                                  ;       STEPERR - Fatal error exit
   21                                  ;
   22
   23 026640                          STEP15:
   24 026640 013704 002324                    MOV     HCOM,R4                 ; Load host comm area address
   25 026644 004737 030626                    JSR     PC,INTCPK               ;INITIALIZE THE COMMAND PACKET
   26 026650 105737 002473                    TSTB    VBUFF1                  ;DATA VALID IN BUFFER 1 ?
   27 026654 001415                           BEQ     2$                      ;BRANCH IF NO, ELSE
   28 026656 012737 045472 002340             MOV     #BUFF1,UADR             ;GET BEGINNING ADDRESS OF BUFF1 BUFFER
   29 026664 004737 030074                    JSR     PC,INITRW               ;INIT SOME READ/WRITE PARAMETERS
   30 026670 063764 002476 000140             ADD     RCTBLK,HC.CPK+P.LBN(R4) ;ADD THE RCT LOGICAL BLOCK TO STARTING
   31 026676 005564 000142                    ADC     HC.CPK+P.LBN+2(R4)      ;HOST LBN OF THE RCT AREA.
   32                                                                         ; Setup to issue MULTWR command
      026702 004737 027514                    JSR     PC,MULTWR               ; Issue command to the xDA
      026706 024250                           STEPFTL                         ; Here if fatal error.
   33 026710                          2$:
   34 026710 105737 002474                    TSTB    VBUFF2                  ;DATA VALID IN BUFFER 2 ?
   35 026714 001415                           BEQ     EXIT15                  ;BRANCH IF NO, ELSE
   36 026716 012737 046472 002340             MOV     #BUFF2,UADR             ;GET BEGINNING ADDRESS OF BUFF2 BUFFER
   37 026724 004737 030074                    JSR     PC,INITRW               ;SOME READ/WRITE PARAMETERS
   38 026730 063764 002500 000140             ADD     MATBLK,HC.CPK+P.LBN(R4) ;ADD THE RCT LOGICAL BLOCK TO STARTING
   39 026736 005564 000142                    ADC     HC.CPK+P.LBN+2(R4)      ;HOST LBN OF THE RCT AREA.
   40                                                                         ; Setup to issue MULTWR command
      026742 004737 027514                    JSR     PC,MULTWR               ; Issue command to the xDA
      026746 024250                           STEPFTL                         ; Here if fatal error.
   41 026750                          EXIT15:
   42 026750 012737 000020 002546             MOV     #16.,NXTSTEP            ; Next STEP is 16.
   43 026756 000207                           RTS     PC                      ; Return
   44
```

```
         1                                  .SBTTL  STEP16   Replacement step 16
         2                                  ;STEP16
         3                                  ;
         4                                ⸴;Write the saved LBN data to the original LBN with FORCED ERROR set,
         5                                  ;if appropriate.
         6                                  ;
         7                                  ;Inputs:
         8                                  ;       R4 - Address of host comm area
         9                                  ;       LBN - LBN address
        10                                  ;       IBUFF - LBN image
        11                                  ;       VIBUFF - Image valid flag
        12                                  ;
        13                                  ;Outputs:
        14                                  ;
        15                                  ;Externals:
        16                                  ;       INCPK, INITRW, ISSUE
        17                                  ;
        18                                  ;Exits:
        19                                  ;       NXTSTEP = 0 - Successful exit
        20                                  ;       STEPERR - Fatal error exit
        21                                  ;
        22
        23 026760                           STEP16:
        24 026760  013704  002324                   MOV     HCOM,R4                  ; Load host comm area address
        25 026764  004737  030626                   JSR     PC,INTCPK                ;INITIALIZE THE COMMAND PACKET
        26 026770  012737  044472  002340           MOV     #IBUFF,UADR              ;GET BEGINNING ADDRESS OF IMAGE BUFFER
        27
        28 026776  005737  002516                   TST     LBNSTAT                  ;SEE IF LBN IMAGE DATA WAS VALID,
        29 027002  100006                            BPL     1$                       ;BRANCH IF SO, ELSE
        30 027004  005737  002164                   TST     ENBWFE                   ; Test for WFE enabled
        31 027010  001403                            BEQ     1$                       ; Branch if NO
        32
        33 027012  052737  010000  002346           BIS     #MD.ERR,CMDMOD           ;STUFF FORCED ERROR MODIFIER IN PACKET.
        34 027020                          1$:
        35 027020  004737  030074                   JSR     PC,INITRW                ; INIT SOME READ/WRITE PARAMETERS AND
        36 027024  013764  002446  000140           MOV     LBN,HC.CPK+P.LBN(R4)     ; GET THE REPLACED LBN ADDRESS.
        37 027032  013764  002450  000142           MOV     LBN+2,HC.CPK+P.LBN+2(R4)
        38                                                                           ; Setup to issue OP.WR command
           027040  012700  000042                   MOV     #OP.WR,R0                ; Load MSCP opcode
           027044  004737  030204                   JSR     PC,ISSUE                 ; Issue command to the xDA
           027050  024250                            STEPFTL                          ; Here if fatal error,
           027052  001423                            BEQ     EXIT16                   ; Branch if command successful.
        39
        40 027054  013700  002354                   MOV     EMSTAT,R0                ; Reload emdmsg status
        41 027060  042700  177740                   BIC     #↑CST.MSK,R0             ; Clear any subcodes
        42 027064  022700  000010                   CMP     #ST.DAT,R0               ; Test for Data error
        43 027070  001414                            BEQ     EXIT16                   ; Branch if YES
        44
        45 027072  012737  012202  002552           MOV     #ERM302,BASEMSG          ; "Failed WRITE..."
        46 027100  012737  000140  002370           MOV     #B.MSCP,EXTMSK           ; Flag all MSCP fields
        47 027106  104456                            TRAP    C$ERHRD
           027110  000456                            .WORD   302
           027112  012043                            .WORD   HDR301
           027114  015504                            .WORD   ERR301
        48 027116  000137  024250                   JMP     STEPFTL                  ; Fatal error exit
        49
        50 027122                           EXIT16:
```

STEP16   Replacement step 16

```
51 027122  012737  000021  002546        10V      #17.,NXTSTEP              ; Next STEP is 17.
52 027130  000207                         RTS      PC                       ; Return
53
```

```
  1                                   .SBTTL   STEP17   Replacement step 17
  2                                   ;STEP17
  3                                   ;
  4                                   ;Update RCT block 0 to incidate that replacement is complete.
  5                                   ;
  6                                   ;Inputs:
  7                                   ;       R4 - Address of host comm area
  8                                   ;       TBUFF - Working copy of RCT block 0
  9                                   ;
 10                                   ;Outputs:
 11                                   ;
 12                                   ;Externals:
 13                                   ;       INCPK, INITRW, MULTWR
 14                                   ;
 15                                   ;Exits:
 16                                   ;       EXIT17  - Successful exit
 17                                   ;       STEPERR - Fatal error exit
 18                                   ;
 19
 20 027132                   STEP17:
 21 027132  013704  002324           MOV      HCOM,R4                ; Load host comm area address
 22 027136  004737  030626           JSR      PC,INTCPK              ;INITIALIZE THE COMMAND PACKET
 23 027142  012701  043472           MOV      #TBUFF,R1              ;GET ADDRESS OF TEMP BUFFER AND
 24 027146  010137  002340           MOV      R1,UADR                ;CHANGE THE UNIBUS ADDRESS FOR WRITE.
 25 027152  062701  000010           ADD      #RFLAGS,R1             ;SET OFFSET TO REPLACEMENT FLAG WORD
 26 027156  005021                   CLR      (R1)+                  ;CLEAR REPLACEMENT WORD (#4) AND
 27 027160  005721                   TST      (R1)+                  ;SKIP WORD #5.
 28 027162  012700  000006           MOV      #<CACHID-LBNLO/2>,R0   ;GET NUMBER OF ENTRIES TO CLEAR
 29 027166  005021            1$:     CLR      (R1)+                  ;CLEAR THAT LOCATION
 30 027170  005300                   DEC      R0                     ;DECR REG. AND SEE IF DONE CLEARING,
 31 027172  001375                   BNE      1$                     ;BRANCH IF NOT, ELSE
 32
 33 027174  012737  000000  002476   MOV      #0,RCTBLK              ;INIT THE RCT LOGICAL BLOCK TO 0 AND
 34 027202  004737  030074           JSR      PC,INITRW              ;SOME READ/WRITE PARAMETERS.
 35                                   ; Setup to issue MULTWR command
    027206  004737  027514           JSR      PC,MULTWR              ; Issue command to the xDA
    027212  024250                    STEPFTL                        ; Here if fatal error.
 36 027214                   EXIT17:
 37 027214  012737  177777  002546   MOV      #-1,NXTSTEP            ; Else, load success
 38 027222  000207                   RTS      PC                     ; Return
 39
```

```
     1                              .SBTTL  MULTRD  RCT Multiread routine
     2                              ;MULTRD
     3                              ;
     4                              ;THIS ROUTINE WILL PERFORM MULTIPLE READS, IF A BLOCK IN THE RCT CANNOT
     5                              ;BE READ FROM THE 1ST COPY OF THE RCT.
     6                              ;
     7                              ;INPUTS:
     8                              ;       COPIES - NUMBER OF COPIES OF THE RCT AREA
     9                              ;       RCTSIZ - RCT SIZE (W/ PAD)
    10                              ;       R4 - ADDRESS OF HOST COMMUNICATION AREA
    11                              ;OUTPUTS:
    12                              ;       LOAD COMMAND PACKET WITH;
    13                              ;       COMPARE MODIFIER,
    14                              ;       @PC    - RETURN IF FATAL ERROR,
    15                              ;       PC+2   - RETURN IF COMMAND NOT SUCCESSFUL
    16                              ;
    17                              ;Calling Seq:
    18                              ;
    19                              ;       JSR     PC,MULTRD
    20                              ;       .WORD   <error exit>
    21                              ;
    22
    23 027224                      MULTRD:
    24 027224  010146                      MOV     R1,-(SP)                ;;PUSH R1 ON STACK
    25 027226  112737  000001  002424      MOVB    #1,RCTCOPY              ; Start with copy 1
    26 027234  105037  002475              CLRB    ERRCNT                  ; Clear RCT error count
    27 027240  005037  002444              CLR     TMPSTAT                 ; Clear temp status
    28 027244  013704  002324              MOV     HCOM,R4                 ; Load host comm area address
    29 027250                      10$:
    30 027250  012737  040000  002346      MOV     #MD.CMP,CMDMOD          ; STUFF COMPARE MODIFIER IN PACKET
    31 027256  004737  034240              JSR     PC,CLRBUF               ; CLEAR THE BUFFER BEFORE READING
    32                                                                     ; Setup to issue OP.RD command
       027262  012700  000041              MOV     #OP.RD,R0               ; Load MSCP opcode
       027266  004737  030204              JSR     PC,ISSUE                ; Issue command to the xDA
       027272  027500                      70$                            ; Here if fatal error,
       027274  001450                      BEQ     50$                     ; Branch if command successful,
    33 027276                      20$:
    34 027276  013700  002354              MOV     EMSTAT,R0               ; Reload emdmsg status
    35 027302  042700  177740              BIC     #^CST.MSK,R0            ; Clear any subcodes
    36 027306  022700  000010              CMP     #ST.DAT,R0              ; Test for Data error
    37 027312  001416                      BEQ     30$                     ; Branch if YES
    38
    39 027314  012737  013216  002552      MOV     #ERM601,BASEMSG         ; "Failed RCT READ..."
    40 027322  012737  000140  002370      MOV     #B.MSCP,EXTMSK          ; Flag all MSCP fields
    41 027330  104456                      TRAP    C$ERHRD
       027332  001131                      .WORD   601
       027334  012043                      .WORD   HDR301
       027336  015504                      .WORD   ERR301
    42 027340  012737  177777  002442      MOV     #-1,STATUS              ; Set Fatal status
    43 027346  000454                      BR      70$                     ; Fatal error exit
    44 027350                      30$:
    45 027350  050037  002444              BIS     R0,TMPSTAT              ; Save status
    46 027354  105237  002475              INCB    ERRCNT                  ; Increment error count
    47 027360  004737  016370              JSR     PC,RCTERR               ; Check this block bad before
    48 027364  027446                      60$                            ; Fatal error return
    49 027366                      40$:
    50 027366  105237  002424              INCB    RCTCOPY                 ; Increment copy
```

```
 51 027372  123737  002423  002424        CMPB    COPIES,RCTCOPY           ; Test for last copy
 52 027400  002406                         BLT     50$                      ; Branch if none left
 53 027402  063764  002416  000140        ADD     RCTSIZ,HC.CPK+P.LBN(R4)  ; Else, adjust LBN
 54 027410  005564  000142                ADC     HC.CPK+P.LBN+2(R4)       ; To next copy of RCT
 55 027414  000715                         BR      10$                      ; And write next copy
 56
 57                                     ;  Normal exit
 58 027416                          50$:
 59 027416  012737  000000  002346        MOV     #0,CMDMOD                ; Clear command modifiers
 60 027424  013737  002444  002442        MOV     TMPSTAT,STATUS           ; Reload status
 61 027432  012601                         MOV     (SP)+,R1                 ;;POP STACK INTO R1
 62 027434  062716  000002                ADD     #2,(SP)                  ; Bump past error return
 63 027440  013700  002442                MOV     STATUS,R0                ; Set condition codes for caller
 64 027444  000207                         RTS     PC                       ; and return
 65 027446                          60$:
 66 027446  012737  013267  002552        MOV     #ERM603,BASEMSG          ; "Failed READ of all RCT copies..."
 67 027454  012737  000000  002370        MOV     #0,EXTMSK                ; Flag RCT fields
 68 027462  104455                         TRAP    C$ERDF
    027464  001133                         .WORD   603
    027466  013173                         .WORD   HDR601
    027470  015720                         .WORD   ERR601
 69 027472  012737  177777  002442        MOV     #-1,STATUS               ; Set cond. codes
 70
 71                                     ;  Fatal Error Exit
 72 027500                          70$:
 73 027500  012601                         MOV     (SP)+,R1                 ;;POP STACK INTO R1
 74 027502  017616  000000                MOV     @(SP),(SP)               ; Load error return
 75 027506  013700  002442                MOV     STATUS,R0                ; Set condition codes for caller
 76 027512  000207                         RTS     PC                       ; and return
 77
```

```
    1                                             .SBTTL  MULTWR  RCT Multiwrite routine
    2                                             ;MULTWR
    3                                             ;
    4                                             ;THIS ROUTINE WILL PERFORM MULTIPLE WRITES ON ALL COPIES OF A LOGICAL BLOCK
    5                                             ;IN THE RCT AREA.
    6                                             ;
    7                                             ;INPUTS:
    8                                             ;       COPIES - NUMBER OF COPIES OF THE RCT AREA
    9                                             ;       RCTSIZ - RCT SIZE (W/ PAD)
   10                                             ;       R4 - ADDRESS OF HOST COMMUNICATION AREA
   11                                             ;OUTPUTS:
   12                                             ;       LOAD COMMAND PACKET WITH;
   13                                             ;       COMPARE MODIFIER,
   14                                             ;       @PC     - RETURN IF FATAL ERROR,
   15                                             ;       PC+2- RETURN IF COMMAND NOT SUCCESSFUL
   16                                             ;
   17                                             ;Calling Seq:
   18                                             ;
   19                                             ;       JSR     PC,MULTWR
   20                                             ;       .WORD   <error exit>
   21                                             ;
   22
   23 027514                              MULTWR:
   24 027514 010146                               MOV     R1,-(SP)              ;;PUSH R1 ON STACK
   25 027516 105037 002475                        CLRB    ERRCNT                ;INIT THE ERROR COUNT
   26 027522 005037 002444                        CLR     TMPSTAT               ; Reset temp status
   27 027526 112737 000001 002424                 MOVB    #1,RCTCOPY            ; Start with first copy
   28 027534 013704 002324                        MOV     HCOM,R4               ; Load host comm area address
   29 027540                              10$:
   30 027540 012737 040000 002346                 MOV     #MD.CMP,CMDMOD        ; STUFF COMPARE MODIFIER IN PACKET
   31                                                                           ; Setup to issue OP.WR command
      027546 012700 000042                        MOV     #OP.WR,R0             ; Load MSCP opcode
      027552 004737 030204                        JSR     PC,ISSUE             ; Issue command to the xDA
      027556 030060                               70$                           ; Here if fatal error,
      027560 001472                               BEQ     45$                   ; Branch if command successful.
   32 027562                              20$:
   33 027562 013700 002354                        MOV     EMSTAT,R0             ; Reload emdmsg status
   34 027566 042700 177740                        BIC     #↑CST.MSK,R0          ; Clear any subcodes
   35 027572 022700 000010                        CMP     #ST.DAT,R0            ; Test for Data error
   36 027576 001416                               BEQ     25$                   ; Branch if YES
   37
   38 027600 012737 013242 002552                 MOV     #ERM602,BASEMSG       ; "Failed RCT WRITE..."
   39 027606 012737 000140 002370                 MOV     #B.MSCP,EXTMSK        ; Flag all MSCP fields
   40 027614 104456                               TRAP    C$ERHRD
      027616 001132                               .WORD   602
      027620 012043                               .WORD   HDR301
      027622 015504                               .WORD   ERR301
   41 027624 012737 177777 002442                 MOV     #-1,STATUS            ; Set Fatal status
   42 027632 000512                               BR      70$                   ; Fatal error exit
   43 027634                              25$:
   44 027634 050037 002444                        BIS     R0,TMPSTAT            ; Save status
   45 027640 105237 002475                        INCB    ERRCNT                ; COUNT THE NUMBER ERRORS DETECTED
   46 027644 004737 016370                        JSR     PC,RCTERR             ; Test if new bad block
   47 027650 030026                               60$                           ; Fatal error return
   48 027652                              30$:
   49 027652 052737 010000 002346                 BIS     #MD.ERR,CMDMOD        ; STUFF FORCED ERROR MODIFIER IN PACKET
   50                                                                           ; Setup to issue OP.WR command
```

```
         027660  012700  000042          MOV      #OP.WR,R0            ; Load MSCP opcode
         027664  004737  030204          JSR      PC,ISSUE            ; Issue command to the xDA
         027670  030060                  70$                          ; Here if fatal error.
         027672  001425                  BEQ      45$                 ; Branch if command successful.
51 027674                          35$:
52 027674  013700  002354          MOV      EMSTAT,R0           ; Reload emdmsg status
53 027700  042700  177740          BIC      #†CST.MSK,R0        ; Clear any subcodes
54 027704  022700  000010          CMP      #ST.DAT,R0          ; Test for Data error
55 027710  001416                  BEQ      45$                 ; Branch if YES
56 027712                          40$:
57 027712  012737  013242  002552  MOV      #ERM602,BASEMSG     ; "Failed RCT WRITE..."
58 027720  012737  000140  002370  MOV      #B.MSCP,EXTMSK      ; Flag all MSCP fields
59 027726  104456                  TRAP     C$ERHRD
   027730  001132                  .WORD    602
   027732  012043                  .WORD    HDR301
   027734  015504                  .WORD    ERR301
60 027736  012737  177777  002442  MOV      #-1,STATUS          ; Set Fatal status
61 027744  000445                  B        70$                 ; Fatal error exit
62 027746                          45$:
63 027746  105237  002424          INCB     RCTCOPY             ; Increment copy
64 027752  123737  002423  002424  CMPB     COPIES,RCTCOPY      ; Test for last copy
65 027760  002406                  BLT      50$                 ; Branch if none left
66 027762  063764  002416  000140  ADD      RCTSIZ,HC.CPK+P.LBN(R4) ; Else, adjust LBN
67 027770  005564  000142          ADC      HC.CPK+P.LBN+2(R4)  ; To next copy of RCT
68 027774  000661                  BR       10$                 ; And write next copy
69
70                                 ;        Normal exit
71 027776                          50$:
72 027776  012737  000000  002346  MOV      #0,CMDMOD           ; Clear command modifiers
73 030004  013737  002444  002442  MOV      TMPSTAT,STATUS      ; Reload status
74 030012  012601                  MOV      (SP)+,R1            ;;POP STACK INTO R1
75 030014  062716  000002          ADD      #2,(SP)             ; Bump past error return
76 030020  013700  002442          MOV      STATUS,R0           ; Set condition codes for caller
77 030024  000207                  RTS      PC                  ; and return
78 030026                          60$:
79 030026  012737  013334  002552  MOV      #ERM604,BASEMSG     ; "Failed WRITE to all copies..."
80 030034  012737  000000  002370  MOV      #0,EXTMSK           ; Flag RCT fields
81 030042  104455                  TRAP     C$ERDF
   030044  001134                  .WORD    604
   030046  013173                  .WORD    HDR601
   030050  015720                  .WORD    ERR601
82 030052  012737  177777  002442  MOV      #-1,STATUS          ; Set cond. codes
83
84                                 ;        Fatal Error Exit
85 030060                          70$:
86 030060  012601                  MOV      (SP)+,R1            ;;POP STACK INTO R1
87 030062  017616  000000          MOV      @(SP),(SP)          ; Load error return
88 030066  013700  002442          MOV      STATUS,R0           ; Set condition codes for caller
89 030072  000207                  RTS      PC                  ; and return
90
```

```
 1                                      .SBTTL   INITRW   Initialize command packet for R/W
 2                                      ;INITRW
 3                                      ;
 4                                      ;INITIALIZE SOME GENERAL PARAMETERS IN THE COMMAND PACKET FOR A
 5                                      ;READ OR WRITE COMMAND.
 6                                      ;
 7                                      ;INPUTS:
 8                                      ;     UADR - UNIBUS ADDRESS IN MEMORY,
 9                                      ;     BCNT - BYTE COUNT,
10                                      ;     UNSIZ - LO ORDER WORD OF STARTING RCT LBN,
11                                      ;     UNSIZ+2 - HI ORDER WORD OF STARTING RCT LBN,
12                                      ;     R4 - ADDRESS OF HOST COMMUNICATION AREA
13                                      ;OUTPUTS:
14                                      ;     LOAD COMMAND PACKET WITH;
15                                      ;     UNIBUS ADDRESS,
16                                      ;     BYTE COUNT,
17                                      ;     STARTING RCT LBN
18                                      ;
19                                      ;Calling Seq:
20                                      ;
21                                      ;     JSR     PC,INITRW
22                                      ;
23
24 030074                              INITRW:
25 030074  013704  002324                 MOV     HCOM,R4                      ; Load host comm area address
26 030100  013764  002340  000124         MOV     UADR,HC.CPK+P.UADR(R4)   ;Load UNIBUS buffer address
27 030106  013764  002334  000120         MOV     BCNT,HC.CPK+P.BCNT(R4)   ;STUFF low order BYTE COUNT
28 030114  013764  002336  000122         MOV     BCNT+2,HC.CPK+P.BCNT+2(R4) ;STUFF high order BYTE COUNT
29 030122  013764  002430  000140         MOV     UNSIZ,HC.CPK+P.LBN(R4)      ;STUFF IN LO ORDER UNIT SIZE AND
30 030130  013764  002432  000142         MOV     UNSIZ+2,HC.CPK+P.LBN+2(R4) ;HI ORDER UNIT SIZE TO GET STARTING
31                                                                             ;LBN OF THE RCT AREA.
32 030136  000207                         RTS     PC                           ;RETURN
33
```

SEQ 0148

CZUDLAO BBR Replacement Utility MACRO Y05.03c Friday 19-Apr-85 10:42 Page 87
INITAC   Initialize command packet for ACCESS

```
 1                                      .SBTTL  INITAC  Initialize command packet for ACCESS
 2                                      ;INITAC
 3                                      ;
 4                                      ;Initialize command packet parameters for an ACESS command.
 5                                      ;
 6                                      ;Inputs:
 7                                      ;       R4 - Address of host communications area
 8                                      ;       UADR - Address of buffer
 9                                      ;       GLBN - ACCESS group starting LBN
10                                      ;       GBCNT - ACCESS group byte count
11                                      ;
12                                      ;Outputs:
13                                      ;       Load command packets with:
14                                      ;         Zero buffer descriptor
15                                      ;         Starting LBN
16                                      ;         Byte count
17                                      ;
18                                      ;Calling Seq:
19                                      ;
20                                      ;       JSR     PC,INITAC
21                                      ;
22
23 030140                              INITAC:
24 030140  013704  002324                      MOV     HCOM,R4                    ; Load host comm area address
25 030144  012764  000000  000124              MOV     #0,HC.CPK+P.UADR(R4)       ; Load UNIBUS buffer address
26 030152  013764  002566  000120              MOV     GBCNT,HC.CPK+P.BCNT(R4)    ; Load low order group byte count
27 030160  013764  002570  000122              MOV     GBCNT+2,HC.CPK+P.BCNT+2(R4) ; Load high order group byte count
28 030166  013764  002562  000140              MOV     GLBN,HC.CPK+P.LBN(R4)      ; Load group
29 030174  013764  002564  000142              MOV     GLBN+2,HC.CPK+P.LBN+2(R4)  ; starting LBN address
30
31 030202  000207                              RTS     PC                         ; Return to caller
32
```

```
   1                                        .SBTTL  ISSUE    Issue command
   2                                        ;ISSUE
   3                                        ;
   4                                        ;ISSUE COMMAND TO xDA. CHECK THAT EXECUTION HAPPENED WITHOUT ERROR.
   5                                        ;
   6                                        ;INPUTS:
   7                                        ;       R0       MSCP command code
   8                                        ;       CTLRTBL Controller table address
   9                                        ;
  10                                        ;OUTPUTS:
  11                                        ;       R0 =     End message status
  12                                        ;                   0 = Success
  13                                        ;                  >0 = Error
  14                                        ;                  -1 = Fatal
  15                                        ;
  16                                        ;
  17                                        ;Calling Seq:
  18                                        ;
  19                                        ;       MOV      #OP_CODE,R0
  20                                        ;       JSR      PC,ISSUE
  21                                        ;       .WORD    <error exit>
  22                                        ;
  23
  24 030204                                 ISSUE:
  25 030204  010037  002344                 MOV      R0,OPCODE              ; Save MSCP command
  26 030210  013704  002324                 MOV      HCOM,R4                ; Reload comm area ptr
  27 030214  013705  002304                 MOV      CTLRTBL,R5             ; Load table start address
  28 030220  004737  030632                 JSR      PC,BLDCMD              ; Build command packet
  29 030224  004737  030744                 JSR      PC,SNDCMD              ; Send command to XDA
  30 030230                          5$:
  31 030230  004737  031072                 JSR      PC,WAIT                ; Wait for message response
  32 030234  030474                          42$                            ; Error return
  33 030236  001100                          BNE      41$                   ; Branch if timeout
  34
  35                                        ;       Unload end packet fields
  36 030240                          10$:
  37 030240  013704  002324                 MOV      HCOM,R4                ; Reload comm area ptr
  38 030244  013705  002304                 MOV      CTLRTBL,R5             ; Load table start address
  39 030250  042765  000004  000014         BIC      #CT.CMD,C.FLG(R5)      ; Clear command issued
  40 030256  016437  000020  002350         MOV      HC.MPK+P.CRF(R4),ENDREF ; Save endmsg reference number
  41 030264  016437  000032  002354         MOV      HC.MPK+P.STS(R4),EMSTAT ; Save endmsg status
  42 030272  116437  000031  002356         MOVB     HC.MPK+P.FLGS(R4),FLAGS ; Save endmsg flags
  43 030300  116437  000030  002352         MOVB     HC.MPK+P.OPCD(R4),ECODE ; Save endcode
  44
  45                                        ;       Test command reference and end code
  46 030306                          20$:
  47 030306  132737  000200  002352         BITB     #OP.END,ECODE         ; Test for end msg packet
  48 030314  001004                          BNE      25$                   ; Branch if YES
  49 030316  004737  016634                 JSR      PC,ERRLOG             ; Else, process error log packet
  50 030322  030614                          50$                            ; Error return
  51 030324  000741                          BR       5$                    ; Wait again
  52
  53                                        ;       Process error log packet
  54 030326                          25$:
  55 030326  122737  000207  002352         CMPB     #OP.END+OP.SEX,ECODE  ; Test for serious exception
  56 030334  001512                          BEQ      44$                   ; Branch if YES
  57
```

```
 58                                        ;       Process end message packet
 59 030336                               30$:
 60 030336   023737  002342  002350        CMP     CMDREF,ENDREF              ; Compare reference numbers
 61 030344   001071                         BNE     43$                       ; Branch if different
 62
 63 030346   013700  002344               MOV     OPCODE,RO                 ; Reload MSCP command code
 64 030352   052700  000200               BIS     #GP.END,RO                ; Set end msg bit
 65 030356   120037  002352               CMPB    RO,ECODE                  ; and compare
 66 030362   001010                        BNE     40$                       ; Branch if different
 67
 68                                        ;       SUCCESS exit
 69 030364                               35$:
 70 030364   062716  000002               ADD     #2,(SP)                   ; Else, bump past error return
 71 030370   013737  002354  0C2442       MOV     EMSTAT,STATUS             ; Set success status
 72 030376   013700  002442               MOV     STATUS,RO                 ; Load status
 73 C30402   000207                        RTS     PC                        ; and return
 74
 75                                        ;       Process unrecognized end packet
 76 030404                               40$:
 77 030404   012737  177777  002442       MOV     #-1,STATUS                ; Set fatal status
 78 030412   012737  012617  002552       MOV     #ERM307,BASEMSG           ; "Unrecognized endcode"
 79 030420   012737  000140  002370       MOV     #B.MSCP,EXTMSK            ; Flag all MSCP fields
 80 030426   104455                        TRAP    C$ERDF
    030430   000463                        .WORD   307
    030432   012043                        .WORD   HDR301
    030434   015504                        .WORD   ERR301
 81 030436   000466                        BR      50$                       ; Exit
 82
 83                                        ;       Process timeout
 84 030440                               41$:
 85 030440   012737  177777  002442       MOV     #-1,STATUS                ; Set fatal status
 86 030446   012737  012727  002552       MOV     #ERM401,BASEMSG           ; "No interrupt received"
 87 030454   012737  000003  002370       MOV     #B.SADDR!B.SAVAL,EXTMSK; Flag all address and value
 88 030462   104455                        TRAP    C$ERDF
    030464   000621                        .WORD   401
    030466   012663                        .WORD   HDR401
    030470   015650                        .WORD   ERR401
 89 030472   000450                        BR      50$                       ; Exit
 90
 91                                        ;       Report fatal controller error
 92 030474                               42$:
 93 030474   012737  177777  002442       MOV     #-1,STATUS                ; Set fatal status
 94 030502   012737  013000  002552       MOV     #ERM402,BASEMSG           ; "Controller error reported..."
 95 030510   012737  000003  002370       MOV     #B.SADDR!B.SAVAL,EXTMSK; Flag all address and value
 96 030516   104455                        TRAP    C$ERDF
    030520   000622                        .WORD   402
    030522   012663                        .WORD   HDR401
    030524   015650                        .WORD   ERR401
 97 030526   000432                        BR      50$                       ; Exit
 98
 99                                        ;       Report cmd reference error
100 030530                              43$:
101 030530   012737  177777  002442       MOV     #-1,STATUS                ; Set fatal status
102 030536   012737  012526  002552       MOV     #ERM306,BASEMSG           ; "Command refernce number error"
103 030544   012737  000151  002370       MOV     #B.CMDRF!B.EMREF!B.MSCP,EXTMSK  ; Flag all MSCP fields
104 030552   104455                        TRAP    C$ERDF
    030554   000462                        .WORD   306
```

ISSUE   Issue command

```
        030556  012043                      .WORD   HDR301
        030560  015504                      .WORD   ERR301
105
106                                 ;       Serious exception error
107 030562                          44$:
108 030562  012737  177777  002442  MOV     #-1,STATUS          ; Set fatal status
109 030570  012737  013050  002552  MOV     #ERM403,BASEMSG     ; "Serious exception"
110 030576  012737  000140  002370  MOV     #B.MSCP,EXTMSK      ; Flag all MSCP fields
111 030604  104455                  TRAP    C$ERDF
    030606  000623                  .WORD   403
    030610  012663                  .WORD   HDR401
    030612  015650                  .WORD   ERR401
112
113                                 ;       Error exit
114 030614                          50$:
115 030614  017616  000000          MOV     @(SP),(SP)          ; Load error erturn
116 030620  013700  002442          MOV     STATUS,R0           ; Reload R0
117 030624  000207                  RTS     PC                  ; and return
118
```

```
  1                                    .SBTTL  BLDCMD  Build command
  2                                    ;BLDCMD
  3                                    ;
  4                                    ;BUILD A COMMAND IN COMMAND PACKET
  5                                    ;
  6                                    ;INPUTS:
  7                                    ;       R0 - COMMAND CODE
  8                                    ;       R5 - CONTROLLER TABLE ADDRESS.
  9                                    ;OUTPUTS:
 10                                    ;       R4 - ADDRESS OF HOST COMMUNICATION AREA.
 11                                    ;       SET COMMAND PACKET SIZE.
 12                                    ;       SET CIRCUIT IDENT.
 13                                    ;       IF 3$=1. CLEAR REST OF COMMAND PACKET FIELDS. ELSE
 14                                    ;               DON'T CLEAR REST OF COMMAND PACKET.
 15                                    ;       SET OPCODE.
 16                                    ;       SET UNIT NUMBER.
 17                                    ;       CLR 3$
 18                                    ;
 19                                    ;Calling Seq:
 20                                    ;
 21                                    ;       JSR     PC,INTCPK              ; Initialize packet/build command
 22                                    ;or
 23                                    ;       JSR     PC,BLDCMD              ; Build command
 24                                    ;
 25
 29
 30 030626                    INTCPK:
 31 030626   005237  030742            INC     3$                     ;ENABLE COMMAND PACKET TO BE CLEARED
 32 030632                    BLDCMD:
 33 030632   010146                    MOV     R1,-(SP)               ;;PUSH R1 ON STACK
    030634   010246                    MOV     R2,-(SP)               ;;PUSH R2 ON STACK
 34 030636   013705  002304            MOV     CTLRTBL,R5             ; Load table start address
 35 030642   013704  002324            MOV     HCOM,R4               ; Load host comm area address
 36 030646   010402                    MOV     R4,R2                 ;COPY TO R2
 37 030650   062702  000100            ADD     #HC.CEV,R2            ;COMPUTE ADDRESS OF COMMAND ENVELOPE
 38 030654   012722  000060            MOV     #HC.PSZ,(R2)+         ;LOAD PACKET SIZE
 39 030660   012722  000000            MOV     #MSCP,(R2)+           ;LOAD MSCP CIRCUIT IDENTIFIER
 40 030664   010122                    MOV     R1,(R2)+              ;PUT IDENTIFIER INTO PACKET
 41 030666   012701  000030            MOV     #<HC.PSZ>/2,R1        ;GET WORDS TO CLEAR
 42 030672   013764  002344  000114    MOV     OPCODE,HC.CPK+P.OPCD(R4); Load command code
 43 030700   013764  002346  000116    MOV     CMDMOD,HC.CPK+P.MOD(R4) ; Load command modifiers
 44 030706                    1$:
 45 030706   005737  030742            TST     3$                     ;INITIALIZE REST OF COMMAND PACKET ?
 46 030712   001403                    BEQ     2$                     ;BRANCH IF NO
 47 030714   005022                    CLR     (R2)+                  ;CLEAR PACKET
 48 030716   005301                    DEC     R1
 49 030720   001372                    BNE     1$
 50 030722                    2$:
 51 030722   016564  000006  000110    MOV     C.DRVN(R5),HC.CPK+P.UNIT(R4) ;LOAD UNIT NUMBER
 52 030730   005037  030742            CLR     3$                     ;DISABLE CLEAR FOR NEXT COMMAND
 53 030734   012602                    MOV     (SP)+,R2               ;;POP STACK INTO R2
    030736   012601                    MOV     (SP)+,R1               ;;POP STACK INTO R1
 54 030740   000207                    RTS     PC
 55
 56 030742   000000            3$:     .WORD   0                      ;INIT PACKET FLAG GOES HERE
 57
 61
```

SEQ 0153

```
    1                                    .SBTTL   SNDCMD   Send command
    2                                    ;SNDCMD
    3                                    ;
    4                                    ;SEND A COMMAND TO THE xDA.
    5                                    ;CLEAR THE RESPONSE PACKET. MARK BOTH PACKETS AVAILABLE TO THE
    6                                    ;xDA. 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 COMMUNICATION AREA
   13                                    ;
   14                                    ;Calling Seq:
   15                                    ;
   16                                    ;      JSR      PC,SNDCMD
   17                                    ;
   18
   19
   20 030744                            SNDCMD:
   21 030744  010046                        MOV     R0,-(SP)                    ;;PUSH R0 ON STACK
      030746  010146                        MOV     R1,-(SP)                    ;;PUSH R1 ON STACK
   22 030750  013705  002304                MOV     CTLRTBL,R5                  ; Load table start address
   23 030754  013704  002324                MOV     HCOM,R4                     ; Load host comm area address
   24 030760  005237  002342                INC     CMDREF                      ; Increment command reference
   25 030764  013764  002342  000104        MOV     CMDREF,HC.CPK+P.CRF(R4)     ;PUT REFERENCE NUMBER IN PACKET
   26 030772  012700  000014                MOV     #HC.MEV,R0                  ;GET INDEX TO MESSAGE ENVELOPE
   27 030776  060400                        ADD     R4,R0                       ;IN HOST COMMUNICATION AREA,
   28 031000  010046                        MOV     R0,-(SP)                    ;;PUSH R0 ON STACK
   29 031002  062716  000004                ADD     #HC.ESZ,(SP)                ;AND ADD ENVELOPE SIZE TO INDEX.
   30 031006  012701  000032                MOV     #<HC.PSZ+HC.ESZ>/2,R1       ;SIZE OF MESSAGE PACKET
   31 031012                            1$:
   32 031012  005020                        CLR     (R0)+                       ;CLEAR ENTIRE MESSAGE PACKET
   33 031014  005301                        DEC     R1
   34 031016  001375                        BNE     1$
   35 031020  012600                        MOV     (SP)+,R0                    ;;POP STACK INTO R0
   36 031022  010064  000004                MOV     R0,HC.MSG(R4)               ;INSERT THE MESSAGE PACKET ADDRESS
   37 031026  012764  140000  000006        MOV     #RG.OWN+RG.FLG,HC.MCT(R4)   ;AND MARK THE OWNER/FLAG BITS IN
   38                                                                            ;THE MESSAGE RING.
   39 031034  062700  000064                ADD     #<HC.PSZ+HC.ESZ>,R0         ;INDEX TO THE BEGINNING OF THE
   40 031040  010064  000010                MOV     R0,HC.CMD(R4)               ;COMMAND PACKET. INSERT THE ADDRESS
   41 031044  012764  100000  000012        MOV     #RG.OWN,HC.CCT(R4)          ;AND MARK THE OWNER BIT IN THE
   42                                                                            ;COMMAND RING.
   43 031052  005775  000000                TST     @(R5)                       ;TELL xDA TO START POLLING
   44 031056  052765  000004  000014        BIS     #CT.CMD,C.FLG(R5)           ;MARK COMMAND ISSUED
   45 031064  012601                        MOV     (SP)+,R1                    ;;POP STACK INTO R1
      031066  012600                        MOV     (SP)+,R0                    ;;POP STACK INTO R0
   46 031070  000207                        RTS     PC
   47
```

```
  1                                    .SBTTL  WAIT     Wait for xDA response
  2                                 ;WAIT
  3                                 ;
  4                                 ;WAIT FOR xDA TO RESPOND WITH A MESSAGE PACKET
  5                                 ;
  6                                 ;INPUTS:
  7                                 ;       R5 - ADDRESS OF CONTROLLER TABLE
  8                                 ;OUTPUTS:
  9                                 ;       Z SET    IF NO ERROR,
 10                                 ;       Z CLR    IF timeout
 11
 12                                 ;Calling Seq:
 13                                 ;
 14                                 ;       JSR      PC,WAIT
 15                                 ;       .WORD    <error exit>
 16                                 ;       BNE      <cmd timeout>
 17                                 ;
 18
 19 031072                         WAIT:
 20 031072  010046                          MOV      R0,-(SP)            ;;PUSH R0 ON STACK
    031074  010146                          MOV      R1,-(SP)            ;;PUSH R1 ON STACK
 21 031076  012700  000036                  MOV      #30.,R0             ;SET TIMEOUT VALUE OF 30 SECONDS,
 22 031102  010501                          MOV      R5,R1               ;GET POINTER TO CONTROLLER TABLE AND
 23 031104  062701  000022                  ADD      #C.TO,R1            ;INDEX TO TIMEOUT COUNTER SO THE
 24 031110  004737  034334                  JSR      PC,SETTO            ;TIME CAN BE SAVED IN THE TABLE.
 25 031114  013705  002304                  MOV      CTLRTBL,R5          ; Load table start address
 26 031120                         1$:
 27 031120  011500                          MOV      (R5),R0             ;GET ADDRESS OF xDAIP REGISTER
 28 031122  032765  000010  000014          BIT      #CT.MSG,C.FLG(R5)   ;LOOK IF INTERRUPT OCCURRED
 29 031130  001031                          BNE      3$                  ;BRANCH IF SO
 30 031132  016037  000002  002316          MOV      2(R0),SAVAL         ;LOOK AT xDASA REGISTER
 31 031140  001037                          BNE      4$                  ;BRANCH IF ERROR CODE PRESENT
 32
 33 031142  000401                          BR       10$                 ; No console interrupt
 34                                                                      ;>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<
 35 031144  104422                          TRAP     C$BRK
 36 031146                         10$:
 37 031146  005737  002210                  TST      KW.CSR              ;SEE IF A CLOCK ON SYSTEM
 38 031152  001762                          BEQ      1$
 39 031154  023765  002222  000024          CMP      KW.EL+2,C.TOH(R5)   ;CHECK IF TIMEOUT HAS HAPPENED
 40 031162  101005                          BHI      2$
 41 031164  001355                          BNE      1$
 42 031166  023765  002220  000022          CMP      KW.EL,C.TO(R5)
 43 031174  103751                          BLO      1$
 44
 45                                 ;       Controller timeout
 46 031176                         2$:
 47 031176  012601                          MOV      (SP)+,R1            ;;POP STACK INTO R1
    031200  012600                          MOV      (SP)+,R0            ;;POP STACK INTO R0
 48 031202  062716  000002                  ADD      #2,(SP)             ; Bump past error return
 49 031206  012700  000001                  MOV      #1,R0               ; Set timeout status
 50 031212  000207                          RTS      PC                  ; Return to caller
 51
 52                                 ;       Normal exit
 53 031214                         3$:
 54 031214  042765  000010  000014          BIC      #CT.MSG,C.FLG(R5)   ; Clear message received flag
 55 031222  012601                          MOV      (SP)+,R1            ;;POP STACK INTO R1
```

```
        031224  012600              MOV     (SP)+,R0        ;;POP STACK INTO R0
    56  031226  062716  000002      ADD     #2,(SP)         ; Bump past error return
    57  031232  012700  000000      MOV     #0,R0           ; Set success status
    58  031236  000207              RTS     PC              ; Return to caller
    59
    60                          ;   Controller error
    61  031240              4$:
    62  031240  012601              MOV     (SP)+,R1        ;;POP STACK INTO R1
        031242  012600              MOV     (SP)+,R0        ;;POP STACK INTO R0
    63  031244  017616  000000      MOV     @(SP),(SP)      ; Load error return
    64  031250  012700  177777      MOV     #-1,R0          ; Set Fatal error status
    65  031254  000207              RTS     PC              ; Return to caller
    66
```

```
     1                              .SBTTL   CALRBN   Calculate RBN
     2                              ;CALRBN
     3                              ;
     4                              ;THIS ROUTINE CALCULATES A PHYSICAL RBN VALUE, FROM THE RCT LOGICAL BLOCK AND
     5                              ;OFFSET WITHIN THE RCT LOGICAL BLOCK.
     6                              ;
     7                              ;INPUTS:
     8                              ;       RCTBLK (RCT logical block),
     9                              ;       RCTOFF (RBN offset within RCT logical block)
    10                              ;
    11                              ;OUTPUTS:
    12                              ;       RBN (lo order word of RBN),
    13                              ;       RBN+2 (hi order word of RBN)
    14                              ;
    15                              ;Calling seq:
    16                              ;
    17                              ;       JSR      PC,CALRBN
    18                              ;
    19
    20 031256  013737  002476  002452  CALRBN: MOV     RCTBLK,RBN          ;GET RCT LOGICAL BLOCK NUMBER,
    21 031264  162737  000002  002452          SUB     #2,RBN              ;CALCULATE THE ACTUAL RBN,
    22 031272  012746  000200                  MOV     #ENTRYS,-(SP)       ;PUT THE MULTIPLER ON THE STACK
       031276  013746  002452                  MOV     RBN,-(SP)           ;PUT THE MULTIPLICAND ON THE STACK
       031302  004737  034542                  JSR     PC,MULT             ;CALL THE MULTPLY ROUTINE
       031306  012637  002452                  MOV     (SP)+,RBN           ;GET THE LSB'S OF THE PRODUCT
       031312  012637  002454                  MOV     (SP)+,RBN+2         ;GET THE MSB'S OF THE PRODUCT
    23 031316  063737  002502  002452          ADD     RCTOFF,RBN          ;ADD THE RCT LOGICAL BLOCK OFFSET AND
    24 031324  005537  002454                  ADC     RBN+2               ;ADD THE CARRY BIT.
    25 031330  000207                          RTS     PC                  ;RETURN
```

```
    1                              .SBTTL  CALBLK   Calculate RCT block
    2                              ;CALBLK
    3                              ;
    4                              ;THIS ROUTINE CALCULATES THE RCT LOGICAL BLOCK AND THE OFFSET OF THE RBN
    5                              ;WITHIN THAT LOGICAL BLOCK, FROM THE PHYSICAL RBN VALUE.
    6                              ;
    7                              ;SP = STACK POINTER
    8                              ;
    9                              ;INPUTS:
   10                              ;      SP+2 (Hi order word of RBN),
   11                              ;      SP (Lo order word of RBN)
   12                              ;
   13                              ;OUTPUTS:
   14                              ;      R0 (contents destroyed),
   15                              ;      SP+2 (RCT logical block of RBN),
   16                              ;      SP (RBN offset within the RCT logical block)
   17                              ;
   18                              ;Calling seq:
   19                              ;
   20                              ;      JSR     PC,CALBLK
   21                              ;
   22                              ;
   23 031332                      CALBLK:
       031332  010346                 MOV     R3,-(SP)         ;;PUSH R3 ON STACK
       031334  010446                 MOV     R4,-(SP)         ;;PUSH R4 ON STACK
       031336  010546                 MOV     R5,-(SP)         ;;PUSH R5 ON STACK
   24 031340  016603  000010          MOV     10(SP),R3        ;GET LO ORDER WORD AND
   25 031344  016604  000012          MOV     12(SP),R4        ;HI ORDER WORD OF PRIMARY RBN.
   26 031350  012700  000200          MOV     #ENTRYS,R0       ;GET # OF ENTRIES (128.) PER RCT BLOCK
   27 031354  004737  034436          JSR     PC,DIVIDE        ;DIVIDE TO FIND THE APPROPRIATE
   28 031360  010366  000012          MOV     R3,12(SP)        ;RCT LOGICAL BLOCK AND
   29 031364  010566  000010          MOV     R5,10(SP)        ;THE RBN OFFSET WITHIN THE BLOCK.
   30 031370  012605                  MOV     (SP)+,R5         ;;POP STACK INTO R5
       031372  012604                 MOV     (SP)+,R4         ;;POP STACK INTO R4
       031374  012603                 MOV     (SP)+,R3         ;;POP STACK INTO R3
   31 031376  000207                  RTS     PC               ;RETURN
   32
```

```
     1                                          .SBTTL   SEARCH   RCT search
     2                                          ;SEARCH
     3                                          ;
     4                                          ;THIS ROUTINE IMPLEMENTS THE RCT SEARCH ALGORITHM.   IT SEARCHES THE RCT
     5                                          ;TO FIND AN AVAILABLE RBN FOR REPLACEMENT.
     6                                          ;
     7                                          ;INPUTS:
     8                                          ;        LBN (Lo order word of LBN being replaced),
     9                                          ;        LBN+2 (Hi order word of LBN being replaced),
    10                                          ;        TRKSIZ (number of LBN'S per track),
    11                                          ;        RBNS (number of REPLACEMENT BLOCKS per track),
    12                                          ;
    13                                          ;OUTPUTS:
    14                                          ;        SFLAG (Search status flag),
    15                                          ;        MATFLG (Match status flag),
    16                                          ;        RBN (Lo order word of RBN replacement),
    17                                          ;        RBN+2 (Hi order word of RBN replacement),
    18                                          ;        MATRBN (Lo order word of RBN which matched the LBN being replaced),
    19                                          ;        MATRBN+2 (Hi order word of RBN which matched the LBN being replaced)
    20                                          ;
    21                                          ;Calling seq:
    22                                          ;
    23                                          ;        JSR      PC,SEARCH
    24                                          ;        BMI      <error exit>
    25                                          ;
    26
    27 031400                          SEARCH:
    28 031400  105037  002514                   CLRB     MATFLG                       ;INIT THE MATCH FLAG,
    29 031404  105037  002512                   CLRB     EMPTY                        ;EMPTY STATUS FLAG AND
    30 031410  105037  002513                   CLRB     RESCAN                       ;RE-SCAN FLAG.
    31 031414  010346                           MOV      R3,-(SP)                     ;;PUSH R3 ON STACK
       031416  010446                           MOV      R4,-(SP)                     ;;PUSH R4 ON STACK
       031420  010546                           MOV      R5,-(SP)                     ;;PUSH R5 ON STACK
    32 031422  013703  002446                   MOV      LBN,R3                       ;GET LO WORD AND
    33 031426  013704  002450                   MOV      LBN+2,R4                     ;HI WORD OF LBN BEING REPLACED.
    34 031432  013700  002420                   MOV      TRKSIZ,R0                    ;GET THE TRACK SIZE,
    35 031436  004737  034436                   JSR      PC,DIVIDE                    ; Divide to find LBN track
    36 031442  113700  002422                   MOVB     RBNS,R0                      ;GET NUMBER OF RBNS PER TRACK.
    37 031446  010046                           MOV      R0,-(SP)                     ;PUT THE MULTIPLER ON THE STACK
    38 031450  010346                           MOV      R3,-(SP)                     ;PUT THE MULTIPLICAND ON THE STACK
    39 031452  004737  034542                   JSR      PC,MULT                      ;CALL THE MULTPLY ROUTINE
    40 031456  004737  031332                   JSR      PC,CALBLK                    ;CALCULATE THE RCT LOGICAL BLOCK AND
    41                                                                                ;THE RBN OFFSET WITHIN THE BLOCK.
    42 031462  012637  002506                   MOV      (SP)+,STROFF                 ;;POP STACK INTO STROFF
       031466  012637  002476                   MOV      (SP)+,RCTBLK                 ;;POP STACK INTO RCTBLK
    43 031472  012605                           MOV      (SP)+,R5                     ;;POP STACK INTO R5
       031474  012604                           MOV      (SP)+,R4                     ;;POP STACK INTO R4
       031476  012603                           MOV      (SP)+,R3                     ;;POP STACK INTO R3
    44 031500  062737  000002  002476  1$:      ADD      #2,RCTBLK                    ;OFFSET TO THE BEGINNING OF THE RCT
    45                                                                                ;DESCRIPTOR BLOCKS.
    46 031506  004737  030074          2$:      JSR      PC,INITRW                    ;SOME READ/WRITE PARAMETERS
    47 031512  063764  002476  000140           ADD      RCTBLK,HC.CPK+P.LBN(R4)      ;ADD THE RCT LOGICAL BLOCK TO STARTING
    48 031520  005564  000142                   ADC      HC.CPK+P.LBN+2(R4)           ;HOST LBN OF THE RCT AREA.
    49                                                                                ; Setup to issue MULTRD command
       031524  004737  027224                   JSR      PC,MULTRD                    ; Issue command to the xDA
       031530  032006                           10$:                                 ; Here if fatal error.
    50 031532                          3$:
```

```
 51 031532  005037  002510                      CLR     DELTA              ;INIT OFFSET TO PRIMARY RBN
 52 031536                             4$:
 53 031536  013701  002340              MOV     UADR,R1            ;GET THE BEGINNING OF UNIBUS ADDRESS
 54 031542  013700  002506              MOV     STROFF,R0          ;GET STARTING OFFSET IN RCT BLOCK,
 55 031546  063700  002510              ADD     DELTA,R0           ;SEE IF OFFSET EXCEEDED LO BLOCK LIMIT,
 56 031552  010037  002502              MOV     R0,RCTOFF          ;STORE IT FOR LATER.
 57 031556  002475                       BLT     8$                 ;BRANCH IF SO, ELSE
 58 031560  020027  000200              CMP     R0,#ENTRYS         ;SEE IF OFFSET EXCEEDED HI BLOCK LIMIT,
 59 031564  002072                       BGE     8$                 ;BRANCH IF SO, ELSE
 60 031566  006300                       ASL     R0                 ;MULTPLY OFFSET
 61 031570  006300                       ASL     R0                 ;BY 4
 62 031572  060001                       ADD     R0,R1              ; Find RBN position in descriptor table
 63 031574  016100  000002              MOV     2(R1),R0           ;GET HI ORDER LBN IN RBN DESCRIPTOR,
 64 031600  042700  007777              BIC     #^C<170000>,R0     ;SEE IF THE RBN IS EMPTY OR A NULL,
 65 031604  001006                       BNE     5$                 ;BRANCH IF NOT EMPTY, ELSE
 66 031606  113737  002512  002515       MOVB    EMPTY,SFLAG        ;SET PRIMARY EMPTY SEARCH STATUS AND
 67 031614  004737  031256               JSR     PC,CALRBN          ;CALCULATE THE RBN.
 68 031620  000475                       BR      12$
 69
 70 031622  042761  170000  000002  5$:  BIC     #170000,2(R1)      ;GET RID OF RBN REPLACEMENT CODE AND
 71 031630  112737  000001  002512       MOVB    #1,EMPTY           ;SET NOT PRIMARY EMPTY SEARCH STATUS.
 72 031636  032700  020000               BIT     #BIT13,R0          ;PRIMARY OR SECONDARY REPLACEMENT ?
 73 031642  001422                       BEQ     6$                 ;BRANCH IF NO, ELSE
 74                                                                 ;SEE IF LBN BEING REPLACED MATCHES
 75 031644  021137  002446               CMP     (R1),LBN           ;THE LBN IN THE LO WORD RBN DESCRIPTOR,
 76 031650  001017                       BNE     6$                 ;BRANCH IF NOT, ELSE
 77 031652  026137  000002  002450       CMP     2(R1),LBN+2        ;THE LBN IN THE HI WORD RBN DESCRIPTOR,
 78 031660  001013                       BNE     6$                 ;BRANCH IF NOT, ELSE
 79 031662  112737  000001  002514       MOVB    #1,MATFLG          ;SET THE MATCHED LBN FLAG.
 80 031670  004737  031256               JSR     PC,CALRBN          ;CALCULATE THE PHYSICAL RBN
 81 031674  013737  002452  002462       MOV     RBN,MATRBN         ;STORE MATCHED RBN LO ORDER WORD AND
 82 031702  013737  002454  002464       MOV     RBN+2,MATRBN+2     ;RBN HI ORDER WORD.
 83
 84 031710  005700                  6$:  TST     R0                 ;IS THIS A NULL REPLACEMENT ?
 85 031712  100017                       BPL     8$                 ;BRANCH IF NO, ELSE
 86 031714  105737  002513               TSTB    RESCAN             ;SEE IF THIS IS A RE-SCAN OF THE RCT BLOCKS,
 87 031720  001404                       BEQ     7$                 ;BRANCH IF NOT, ELSE
 88 031722  112737  000002  002515       MOVB    #2,SFLAG           ;SET RCT FULL FLAG IN SEARCH STATUS AND
 89 031730  000431                       BR      12$                ;EXIT.
 90
 91 031732  112737  000001  002513  7$:  MOVB    #1,RESCAN          ;SET SEQUENTIAL RE-SCAN FLAG AND
 92 031740  005037  002506               CLR     STROFF             ;START SEQUENCIAL SEARCH THRU THE
 93 031744  005037  002476               CLR     RCTBLK             ;FIRST RCT LOGICAL BLOCK.
 94 031750  000653                       BR      1$
 95
 96 031752  005437  002510          8$:  NEG     DELTA              ;SEE WHICH DIRECTION SEARCH IS GOING,
 97 031756  100402                       BMI     9$                 ;BRANCH IF REVERSE, ELSE
 98 031760  005237  002510               INC     DELTA              ;INCREMENT TO NEXT RBN
 99                                                                 ;DESCRIPTOR FORWARD.
100 031764  023727  002510  000200  9$:  CMP     DELTA,#ENTRYS      ;SEE IF STILL IN SAME RCT BLOCK,
101 031772  002661                       BLT     4$                 ;BRANCH IF SO, ELSE
102 031774  005037  002506               CLR     STROFF             ;START SEQUENTIAL SEARCH THRU THE
103 032000  005237  002476               INC     RCTBLK             ;NEXT RCT LOGICAL BLOCK.
104 032004  000640                       BR      2$
105 032006                         10$:
106 032006  012700  177777               MOV     #-1,R0             ; Set fatal error
107 032012                         11$:
```

```
108 032012  000207                      RTS     PC                  ; Return to caller
109 032014                     12$:
110 032014  012700  000000             MOV     #0,R0               ; Load success
111 032020  000207                      RTS     PC                  ; and return
```

```
    1                                    .SBTTL  GETVSN  Get volume Serial Number
    2                                 ;GETVSN
    3                                 ;
    4                                 ;Retrieve and format volume serial number
    5                                 ;
    6                                 ;Inputs:
    7                                 ;       UADR:   Address of buffer containing RCT block 0 image
    8                                 ;       VSNBUF: Twenty-four byte ASCII buffer
    9                                 ;
   10                                 ;Outputs:
   11                                 ;       VNSPTR: Address of .ASCIZ string containing VSN
   12                                 ;
   13                                 ;Externals:
   14                                 ;       DIV10:  Divide 4 byte positive biniary number by 10.
   15                                 ;
   16                                 ;Calling seq:
   17                                 ;
   18                                 ;       JSR     PC,GETVSN
   19                                 ;
   20
   21 032022                         GETVSN:
   22 032022  010146                      MOV     R1,-(SP)            ;;PUSH R1 ON STACK
      032024  010246                      MOV     R2,-(SP)            ;;PUSH R2 ON STACK
      032026  010346                      MOV     R3,-(SP)            ;;PUSH R3 ON STACK
      032030  010446                      MOV     R4,-(SP)            ;;PUSH R4 ON STACK
      032032  010546                      MOV     R5,-(SP)            ;;PUSH R5 ON STACK
   23 032034  013700  002340              MOV     UADR,R0             ;GET THE BEGINNING OF UNIBUS ADDRESS
   24 032040  012001                      MOV     (R0)+,R1            ;PUT WORD 0,
   25 032042  012002                      MOV     (R0)+,R2            ;WORD 1,
   26 032044  012003                      MOV     (R0)+,R3            ;WORD 2 AND
   27 032046  012004                      MOV     (R0)+,R4            ;WORD 3 IN REGISTERS.
   28 032050  012700  002620              MOV     #VSNBUF,R0          ; Load address of VSN text
   29 032054  012705  000024              MOV     #20.,R5             ;GET NUMBER OF CHARACTERS
   30 032060                         3$:
   31 032060  112720  000040              MOVB    #' ,(R0)+           ;INTIALIZE STORAGE WITH SPACES
   32 032064  005305                      DEC     R5                  ;DONE AREA YET ?
   33 032066  001374                      BNE     3$                  ;BR IF NO
   34
   35 032070  005010                      CLR     (R0)                ;INSERT NULL AS TERMINATOR
   36 032072                         4$:
   37 032072  004737  034474              JSR     PC,DIV10            ;CONVERT OCTAL TO DECIMAL AND
   38 032076  062705  000060              ADD     #'0,R5              ;MAKE IT ASCIZ.
   39 032102  110540                      MOVB    R5,-(R0)
   40 032104  010146                      MOV     R1,-(SP)
   41 032106  050216                      BIS     R2,(SP)
   42 032110  050316                      BIS     R3,(SP)
   43 032112  050426                      BIS     R4,(SP)+
   44 032114  001366                      BNE     4$                  ;BRANCH IF NOT DONE
   45
   46 032116  010037  002616              MOV     R0,VSNPTR           ; Save VSN pointer
   47 032122  012605                      MOV     (SP)+,R5            ;;POP STACK INTO R5
      032124  012604                      MOV     (SP)+,R4            ;;POP STACK INTO R4
      032126  012603                      MOV     (SP)+,R3            ;;POP STACK INTO R3
      032130  012602                      MOV     (SP)+,R2            ;;POP STACK INTO R2
      032132  012601                      MOV     (SP)+,R1            ;;POP STACK INTO R1
   48 032134  000207                      RTS     PC                  ; Return to caller
   49
```

```
 1                                 .SBTTL  GTDRVT   Get drive table pointer
 2                                 ;GTDRVT
 3                                 ;
 4                                 ;GET DRIVE TABLE POINTER
 5                                 ;
 6                                 ;INPUTS:
 7                                 ;       R1 - DRIVE NUMBER
 8                                 ;       R5 - CONTROLLER TABLE ADDRESS
 9                                 ;OUTPUTS:
10                                 ;       R4 - DRIVE TABLE ADDRESS
11                                 ;
12
13 032136  010504         GTDRVT: MOV     R5,R4                   ;GET CONTROLLER TABLE ADRESS
14 032140  062704  000020         ADD     #C.DRVT,R4              ;ADD OFFSET TO DRIVE TABLE ADDRESS
15 032144  011404                 MOV     (R4),R4                 ;GET ADDRESS OF TABLE
16 032146  000207                 RTS     PC
17
```

SEQ 0163

```
 1                              .SBTTL  DSPHELP Display operator help
 2                              ;
 3                              ;
 4                              ;           SUBROUTINE TO DUMP DOS HELP FILE TO CONSOLE
 5                              ;
 6                              ;           INPUT:          HLPFILE:  .ASCIZ "FILNAM.EXT"
 7                              ;
 8                              ;           OUTPUT:         PRINTS 72 CHARACTER BUFFER TO CONSOLE
 9                              ;
10                              ;           REGISTER USAGE: R1  =    POINTER TO LINE TEXT BUFFER
11                              ;                           R2  =    TAB POSITION COUNTER
12                              ;                           R4  =    CHARACTER FROM FILE
13                              ;
14                              ;           CALLING SEQ:    JSR     PC,DPSHELP
15                              ;
16                              ;           FUNCTION:       DSPHELP WILL OPEN, READ AND CLOSE THE SELETED FILE
17                              ;                           USING THE DIAG SUPERVISOR MACROS.  CHARACTERS
18                              ;                           ARE READ ONE AT A TIME AND TRANSFERRED TO THE
19                              ;                           LINE BUFFER, WHICH IS DUMPED TO THE CONSOLE
20                              ;                           WHEN FULL.
21
22 032150               DSPHELP:
23
24 032150   010146                  MOV     R1,-(SP)                ;;PUSH R1 ON STACK
   032152   010246                  MOV     R2,-(SP)                ;;PUSH R2 ON STACK
   032154   010446                  MOV     R4,-(SP)                ;;PUSH R4 ON STACK
25
26                              ;   OPEN FILE AND PRINT NULL LINE
27 032156               100$:
28 032156   012700   014672        MOV     #HLPFILE,R0
   032162   104434                  TRAP    C$OPNRD
29 032164   012701   002716        MOV     #LINBUF,R1              ;AND LOAD POINTER TO BUFFER
30 032170   112721   000015        MOVB    #CR,(R1)+              ;MOVE CARRAGE RETURN
31 032174   112721   000012        MOVB    #LF,(R1)+              ;AND LINE FEED TO BUFFER
32 032200   112704   000000        MOVB    #0,R4                  ;FOLLOWED BY NULL
33 032204   004737   032270        JSR     PC,600$                ;PRINT BUFFER, RESET R1
34
35                              ;   GET NEXT CHARACTER FROM SUPERVISOR, AND TEST FOR END OF FILE
36 032210               200$:
37 032210   104426                  TRAP    C$GETB
   032212   010004                  MOV     R0,R4
38 032214   103004                  BCC     400$
39 032216   001405                  BEQ     425$                   ;OR IF CHARACTER = ZERO
40
41                              ;   TEST AND PROCESS TAB CHARACTER
42 032220               300$:
43 032220   004737   032270        JSR     PC,600$                ;PRINT SPACES FOR TAB
44 032224   000771                  BR      200$                   ;AND RETRIEVE NEXT CHARACTER
45
46                              ;   LAST CHAR OR END OF FILE
47 032226               400$:
48 032226   112704   000000        MOVB    #0,R4                  ;INSURE NULL CHAR
49 032232               425$:
50 032232   004737   032270        JSR     PC,600$                ;EMPTY
51 032236   112721   000015        MOVB    #CR,(R1)+              ;OUTPUT CARRAGE RETURN
52 032242   112721   000012        MOVB    #LF,(R1)+              ;AND LINE FEED
53 032246   112704   000000        MOVB    #0,R4                  ;FOLLOWED BY NULL
```

```
 54 032252  004737  032270            JSR     PC,600$                 ;PRINT
 55
 56                         ;        CLOSE FILE AND RETURN TO MAIN
 57 032256                  500$:
 58 032256  104435                    TRAP    C$CLOS
 59 032260  012604                    MOV     (SP)+,R4                ;;POP STACK INTO R4
    032262  012602                    MOV     (SP)+,R2                ;;POP STACK INTO R2
    032264  012601                    MOV     (SP)+,R1                ;;POP STACK INTO R1
 60 032266  000207                    RTS     PC                      ;AND RETURN
 61
 62                         ;        MOVE CHAR TO BUFFER, TEST FOR NULL CHAR OR END OF BUFFER
 63 032270                  600$:
 64 032270  001415                    BEQ     800$                    ; Branch if null
 65
 66                         ;        TEST FOR CONTROL CHARS, TAB ACTIVE OR RETURN
 67 032272                  700$:
 68 032272  120427  000040            CMPB    R4,#40                  ;COMPARE CHAR WITH BLANK
 69 032276  100010                    BPL     750$                    ;BRANCH IF CONTROL CHAR
 70 032300  120427  000015            CMPB    R4,#15                  ; Test for CR
 71 032304  001006                    BNE     780$                    ; If not, ignore
 72
 73 032306  112721  000015            MOVB    #15,(R1)+               ; Else, load CR
 74 032312  112721  000012            MOVB    #12,(R1)+               ; and LF
 75 032316  000402                    BR      800$                    ; and print
 76 032320                  750$:
 77 032320  110421                    MOVB    R4,(R1)+                ;MOVE CHAR TO BUFFER
 78 032322                  780$:
 79 032322  000207                    RTS     PC                      ;NO:   JUST RETURN
 80
 81                         ;        RESET TAB COUNTER, TAB ACTIVE FLAG AND RETURN
 82 032324                  800$:
 83 032324  112721  000000            MOVB    #0,(R1)+                ;MOVE CHAR TO BUFFER
 84 032330  012701  002716            MOV     #LINBUF,R1              ;AND RESET POINTER
 85 032334  010146                    MOV     R1,-(SP)
    032336  012746  014220            MOV     #F.TEXT,-(SP)
    032342  012746  000002            MOV     #2,-(SP)
    032346  010600                    MOV     SP,R0
    032350  104414                    TRAP    C$PNTB
    032352  062706  000006            ADD     #6,SP
 86 032356  012701  002716            MOV     #LINBUF,R1              ;AND RESET POINTER
 87 032362  000207                    RTS     PC                      ;AND RETURN
 88
```

```
    1                                  .SBTTL  DSPDSC  Display RCT descriptors
    2                                  ;DSPDSC
    3                                  ;
    4                                  ;Read and display RBN descriptors in RCT.
    5                                  ;
    6                                  ;Inputs:
    7                                  ;       DBUFF   Buffer for RCT blocks.
    8                                  ;Outputs:
    9                                  ;       Outputs replacement descriptors on console.
   10                                  ;
   11                                  ;Calling seq:
   12                                  ;
   13                                  ;       JSR     PC,DEPDSC
   14                                  ;
   15
   16
   17 032364                   DSPDSC:
   18 032364  013705  002304          MOV     CTLRTBL,R5              ; Load table start address
   19 032370  013704  002324          MOV     HCOM,R4                ; Load host comm area address
   20                                                                 ; "RCT descriptor info..."
   21 032374  013746  002074          MOV     L$LUN,-(SP)            ;PUSH L$LUN ON STACK
      032400  012746  006170          MOV     #INF210,-(SP)          ;PUSH #INF210 ON STACK
      032404  004137  033254          JSR     R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
      032410  000004                  .WORD   ARG.CT                 ;ARGUMENT COUNT * 2
   22
   23 032412  004737  030626          JSR     PC,INTCPK              ;INITIALIZE THE COMMAND PACKET,
   24 032416  005037  002434          CLR     PRIMRY                 ;THE PRIMARY RBN COUNT,
   25 032422  005037  002436          CLR     SCNDRY                 ;THE SECONDARY RBN COUNT,
   26 032426  005037  002440          CLR     UNUSE                  ;THE UNUSABLE RBN COUNT,
   27 032432  012737  042472  002340  MOV     #DBUFF,UADR            ;GET BEGINNING ADDRESS OF DATA BUFFER
   28 032440  012737  000001  002476  MOV     #1,RCTBLK              ;THE RCT LOGICAL BLOCK TO 1 AND
   29 032446                   1$:
   30 032446  005037  002502          CLR     RCTOFF                 ;THE OFFSET WITHIN THE RCT BLOCK.
   31 032452  005237  002476          INC     RCTBLK                 ;INCREMENT RCT LOGICAL BLOCK AND
   32 032456  023737  002476  002416  CMP     RCTBLK,RCTSIZ          ;SEE IF THE ENTIRE RCT WAS READ.
   33 032464  002411                  BLT     2$                     ;BRANCH IF NOT, ELSE
   34
   35 032466  012737  013242  002552  MOV     #ERM602,BASEMSG        ; "No null entry..."
   36 032474  104457                  TRAP    C$ERSOFT
      032476  001132                  .WORD   602
      032500  013173                  .WORD   HDR601
      032502  015220                  .WORD   ERRO01
   37 032504  000137  033054          JMP     100$                   ;BRANCH TO EXIT ON ERROR
   38 032510                   2$:
   39 032510  004737  030074          JSR     PC,INITRW              ;INIT SOME READ/WRITE PARAMETERS
   40 032514  063764  002476  000140  ADD     RCTBLK,HC.CPK+P.LBN(R4) ;ADD RCT LOGICAL BLOCK TO STARTING
   41 032522  005564  000142          ADC     HC.CPK+P.LBN+2(R4)     ;HOST LBN OF THE RCT AREA.
   42                                                                 ; Setup to issue MULTRD command
      032526  004737  027224          JSR     PC,MULTRD              ; Issue command to the xDA
      032532  033054                  100$                           ; Here if fatal error.
   43 032534                   4$:
   44 032534  013701  002340          MOV     UADR,R1                ;GET THE BEGINNING OF UNIBUS ADDRESS
   45 032540                   5$:
   46 032540  004737  031256          JSR     PC,CALRBN              ;CALCULATE THE PHYSICAL RBN
   47 032544  016103  000002          MOV     2(R1),R3               ;GET HI ORDER LBN IN RBN DESCRIPTOR.
   48 032550  042703  007777          BIC     #^C<170000>,R3         ;SEE IF THE RBN IS EMPTY OR A NULL.
   49 032554  001521                  BEQ     9$                     ;BRANCH IF EMPTY,
```

```
 50 032556 100033                        BPL    6$                  ;BRANCH IF NOT NULL, ELSE
 51                                                                  ;THIS IS MUST BE THE END, SO
 52                                                                  ; "RCT descriptor summary..."
 53 032560 013701 002434                 MOV    PRIMRY,R1           ; Load primary
 54 032564 063701 002436                 ADD    SCNDRY,R1           ; Add secondary
 55 032570 063701 002440                 ADD    UNUSE,R1            ; Add unusable
 56 032574 012746 006452                 MOV    #INF220,-(SP)       ;PUSH #INF220 ON STACK
    032600 004137 033254                 JSR    R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
    032604 000002                        .WORD  ARG.CT              ;ARGUMENT COUNT * 2
 57 032606 013746 002422                 MOV    RBNS,-(SP)          ;PUSH RBNS ON STACK
    032612 010146                        MOV    R1,-(SP)            ;PUSH R1 ON STACK
    032614 013746 002440                 MOV    UNUSE,-(SP)         ;PUSH UNUSE ON STACK
    032620 013746 002436                 MOV    SCNDRY,-(SP)        ;PUSH SCNDRY ON STACK
    032624 013746 002434                 MOV    PRIMRY,-(SP)        ;PUSH PRIMRY ON STACK
    032630 012746 006523                 MOV    #INF221,-(SP)       ;PUSH #INF221 ON STACK
    032634 004137 033254                 JSR    R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
    032640 000014                        .WORD  ARG.CT              ;ARGUMENT COUNT * 2
 58
 59 032642 000137 033042                 JMP    90$                 ;BRANCH TO EXIT
 60
 61 032646 042761 170000 000002 6$:      BIC    #170000,2(R1)       ;GET RID OF THE REPLACEMENT BLOCK CODE,
 62                                                                  ;THE LBN CAN BE PRINTED LATER.
 63 032654 020327 020000                 CMP    R3,#BIT13           ;IS THIS A PRIMARY REPLACEMENT ?
 64 032660 001017                        BNE    7$                  ;BRANCH IF NO, ELSE
 65 032662 005237 002434                 INC    PRIMRY              ;UPDATE # OF PRIMARY REPLACEMENT BLOCKS
 66                                                                  ; "RBN "D32". Primary replacement..."
 67 032666 016146 000002                 MOV    2(R1),-(SP)         ;PUSH 2(R1) ON STACK
    032672 011146                        MOV    (R1),-(SP)          ;PUSH (R1) ON STACK
    032674 013746 002454                 MOV    RBN+2,-(SP)         ;PUSH RBN+2 ON STACK
    032700 013746 002452                 MOV    RBN,-(SP)           ;PUSH RBN ON STACK
    032704 012746 006244                 MOV    #INF211,-(SP)       ;PUSH #INF211 ON STACK
    032710 004137 033254                 JSR    R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
    032714 000012                        .WORD  ARG.CT              ;ARGUMENT COUNT * 2
 68 032716 000440                        BR     9$
 69
 70 032720 020327 030000          7$:    CMP    R3,#BIT13!BIT12     ;IS THIS A SECONDARY REPLACEMENT ?
 71 032724 001017                        BNE    8$                  ;BRANCH IF NO, ELSE
 72 032726 005237 002436                 INC    SCNDRY              ;UPDATE # OF SECONDARY
 73                                                                  ;REPLACEMENT BLOCKS
 74                                                                  ; "RBN "D32". Secondary replacement..."
 75 032732 016146 000002                 MOV    2(R1),-(SP)         ;PUSH 2(R1) ON STACK
    032736 011146                        MOV    (R1),-(SP)          ;PUSH (R1) ON STACK
    032740 013746 002454                 MOV    RBN+2,-(SP)         ;PUSH RBN+2 ON STACK
    032744 013746 002452                 MOV    RBN,-(SP)           ;PUSH RBN ON STACK
    032750 012746 006332                 MOV    #INF212,-(SP)       ;PUSH #INF212 ON STACK
    032754 004137 033254                 JSR    R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
    032760 000012                        .WORD  ARG.CT              ;ARGUMENT COUNT * 2
 76 032762 000416                        BR     9$
 77
 78 032764 020327 040000          8$:    CMP    R3,#BIT14           ;IS THIS AN UNUSABLE REPLACEMENT ?
 79 032770 103413                        BLO    9$                  ;BRANCH IF NO, ELSE
 80 032772 005237 002440                 INC    UNUSE               ;UPDATE # OF UNUSABLE REPLACEMENT BLOCKS
 81                                                                  ; "RBN "D32". Unusable"
 82 032776 013746 002454                 MOV    RBN+2,-(SP)         ;PUSH RBN+2 ON STACK
    033002 013746 002452                 MOV    RBN,-(SP)           ;PUSH RBN ON STACK
    033006 012746 006420                 MOV    #INF213,-(SP)       ;PUSH #INF213 ON STACK
    033012 004137 033254                 JSR    R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
```

```
        033016  000006                           .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
83
84 033020  022121                    9$:        CMP     (R1)+,(R1)+             ;UPDATE POINTER IN THE TABLE
85 033022  005237  002502                       INC     RCTOFF                  ;INCREMENT TO NEXT RBN DESCRIPTOR AND
86 033026  023727  002502  000200               CMP     RCTOFF,#ENTRYS          ;SEE IF ALL ENTRIES DONE,
87 033034  002641                               BLT     5$                      ;BRANCH IF NOT, ELSE
88 033036  000137  032446                       JMP     1$                      ;CONTINUE TO READ NEXT BLOCK OF
89                                                                               ;REPLACEMENT DESCRIPTORS.
90 033042                             90$:
91 033042  062716  000002                       ADD     #2,(SP)                 ; Bump past error return
92 033046  012700  000000                       MOV     #0,R0                   ; Status is success
93 033052  000207                               RTS     PC                      ; Return to caller
94 033054                             100$:
95 033054  017616  000000                       MOV     @(SP),(SP)              ; Load error return
96 033060  012700  177777                       MOV     #-1,R0                  ; Status is failure
97 033064  000207                               RTS     PC                      ; Return to caller
98
```

```
   1                                   .SBTTL  PRINTC   Print character
   2                                  ;PRINTC
   3                                  ;
   4                                  ;PRINT A CHARACTER
   5                                  ;
   6                                  ;CALL WITH MACRO PRINT
   7
   8 033066  110037  033242   PRINTC: MOVB    RO,TTYOUT           ;SAVE CHARACTER FOR TTY OUTPUT
   9 033072  010146                   MOV     R1,-(SP)            ;;PUSH R1 ON STACK
  10 033074  012701  014220           MOV     #F.TEXT,R1          ;PICKUP FORMATTED ASCIZ STRING STATEMENT
  11 033100  120027  000015           CMPB    RO,#CR              ;IF NOT A CARRIAGE RETURN, THEN
  12 033104  001002                   BNE     1$                  ;PRINT SOME OTHER CHARACTER, ELSE
  13 033106  012701  014215           MOV     #F.CRLF,R1          ;PICKUP FORMATTED ASCIZ STRING STATEMENT
  14                                                              ;GO PRINT CR-LF.
  15 033112  004777  000232   1$:     JSR     PC,@PTYPE           ;PRINT THE ASCIZ STRING,
  16 033116  012601                   MOV     (SP)+,R1            ;;POP STACK INTO R1
  17 033120  000207                   RTS     PC
  18 033122                   PF:
     033122  012746  033242           MOV     #TTYOUT,-(SP)
     033126  010146                   MOV     R1,-(SP)
     033130  012746  000002           MOV     #2,-(SP)
     033134  010600                   MOV     SP,RO
     033136  104417                   TRAP    C$PNTF
     033140  062706  000006           ADD     #6,SP
  19 033144  000207                   RTS     PC
  20 033146                   PB:
     033146  012746  033242           MOV     #TTYOUT,-(SP)
     033152  010146                   MOV     R1,-(SP)
     033154  012746  000002           MOV     #2,-(SP)
     033160  010600                   MOV     SP,RO
     033162  104414                   TRAP    C$PNTB
     033164  062706  000006           ADD     #6,SP
  21 033170  000207                   RTS     PC
  22 033172                   PX:
     033172  012746  033242           MOV     #TTYOUT,-(SP)
     033176  010146                   MOV     R1,-(SP)
     033200  012746  000002           MOV     #2,-(SP)
     033204  010600                   MOV     SP,RO
     033206  104415                   TRAP    C$PNTX
     033210  062706  000006           ADD     #6,SP
  23 033214  000207                   RTS     PC
  24 033216                   PS:
     033216  012746  033242           MOV     #TTYOUT,-(SP)
     033222  010146                   MOV     R1,-(SP)
     033224  012746  000002           MOV     #2,-(SP)
     033230  010600                   MOV     SP,RO
     033232  104416                   TRAP    C$PNTS
     033234  062706  000006           ADD     #6,SP
  25 033240  000207                   RTS     PC
  26
  27 033242  000        TTYOUT: .BYTE   0                   ;TTY OUTPUT BUFFER
  28 033243  000                .BYTE   0                   ;TERMINATOR FOR ASCIZ STRING
  29                            .EVEN
```

```
                                          .SBTTL  LPNTx    Print formatted message
   1                                      ;PRINT FORMATTED MESSAGE
   2                                      ;
   3                                      ;CALL WITH MACRO PNT, PNTF, PNTB, PNTX, OR PNTS
   4
   5
   6 033244  012737  033122  033350  LPNTF:   MOV     #PF,PTYPE
   7 033252  000413                           BR      LPNT
   8
   9 033254  012737  033146  033350  LPNTB:   MOV     #PB,PTYPE
  10 033262  000407                           BR      LPNT
  11
  12 033264  012737  033172  033350  LPNTX:   MOV     #PX,PTYPE
  13 033272  000403                           BR      LPNT
  14
  15 033274  012737  033216  033350  LPNTS:   MOV     #PS,PTYPE
  16
  17 033302                          LPNT:
     033302  010246                           MOV     R2,-(SP)          ;;PUSH R2 ON STACK
     033304  010346                           MOV     R3,-(SP)          ;;PUSH R3 ON STACK
     033306  010446                           MOV     R4,-(SP)          ;;PUSH R4 ON STACK
     033310  010546                           MOV     R5,-(SP)          ;;PUSH R5 ON STACK
  18                                  ;        MOV     (R1)+,R2          ;GET ADDRESS OF ASCIZ STRING
  19 033312  010604                           MOV     SP,R4             ;COMPUTE ADDRESS OF 1ST ARGUMENT AND
  20 033314  062704  000012                   ADD     #12,R4            ;SAVE IT IN R4.
  21 033320  012402                           MOV     (R4)+,R2          ; Load address of ASCII string
  22 033322  010146                           MOV     R1,-(SP)          ;;PUSH R1 ON STACK
  23 033324  004737  033352                   JSR     PC,OSTRNG         ;PRINT THE FORMATTED MESSAGE
  24 033330  012600                           MOV     (SP)+,R0          ;;POP STACK INTO R0
     033332  012605                           MOV     (SP)+,R5          ;;POP STACK INTO R5
     033334  012604                           MOV     (SP)+,R4          ;;POP STACK INTO R4
     033336  012603                           MOV     (SP)+,R3          ;;POP STACK INTO R3
     033340  012602                           MOV     (SP)+,R2          ;;POP STACK INTO R2
     033342  012601                           MOV     (SP)+,R1          ;;POP STACK INTO R1
  25 033344  062006                           ADD     (R0)+,SP          ;ADJUST STACK POINTER OVER ARGUMENTS
  26 033346  000110                           JMP     @R0               ;RETURN
  27
  28 033350  033122                  PTYPE:   .WORD   PF                ;PRINT TYPE
```

```
       1                                       .SBTTL  OSTRNG  Formatted message output
       2                                       ;OSTRNG
       3                                       ;
       4                                       ;OUTPUT A MESSAGE ACCORDING TO A FORMAT STRING
       5                                       ;FORMAT OF THE ASCIZ STRING IS AS FOLLOWS:
       6                                       ;
       7                                       ;CHARACTERS ENCLOSED IN QUOTES ARE TO BE PRINTED AS THEY ARE.
       8                                       ;
       9                                       ;OTHERWISE CODE IS A SINGLE LETTER FOLLOWED BY AN OPTIONAL DECIMAL
      10                                       ;NUMBER:
      11                                       ;  On - PRINT OCTAL NUMBER. n REPRESENTS SIZE OF BINARY NUMBER PASSED
      12                                       ;         IN PARAMETER IN BITS. MAY BE IN RANGE 1 TO 32. IF n>16, TWO PARAMETER
      13                                       ;         WORDS ARE USED, OTHERWISE ONLY ONE WORD. LEADING ZEROS ARE PRINTED.
      14                                       ;         n IS ALWAYS SPECIFIED.
      15                                       ;  Dn - PRINT UNSIGNED DECIMAL NUMBER FROM n BIT PARAMETER. LEADING ZEROS
      16                                       ;         ARE NOT PRINTED. A 16 BIT NUMBER EQUAL TO ZERO WILL PRINT "O".
      17                                       ;  Hn - PRINT HEX NUMBER FROM PARAMETER OF n BITS. IF n>16 TWO PARAMETERS
      18                                       ;         ARE USED, OTHERWISE ONLY ONE PARAMETER. LEADING ZEROS ARE PRINTED.
      19                                       ;  Sn - PRINT n SPACES. n ASSUMED TO BE 1.
      20                                       ;  Nn - START NEW LINE (CR-LF SEQUENCE). n ASSUMED TO BE 1.
      21                                       ;  An - PRINT N ASCII CHARACTERS FROM PARAMETERS, n ASSUMED TO BE 1.
      22                                       ;         n/2 PARAMETER WORDS USED.
      23                                       ;  Rn - EXECUTE ROUTINE #n. n MUST BE GIVEN AND DEFINED IN HOST PROGRAM.
      24                                       ;
      25                                       ;A NULL CHARACTER MEANS END OF MESSAGE. A NULL AS FIRST CHARACTER IN STRING
      26                                       ;MUST BE IGNORED.
      27                                       ;
      28                                       ;INPUTS:
      29                                       ;      R2 - ADDRESS OF START OF FORMAT STRING
      30                                       ;      R4 - ADDRESS OF PARAMETERS
      31                                       ;OUTPUTS:
      32                                       ;      R2 AND R4 UPDATED TO END OF STRING AND PARAMETERS
      36
      37 033352  112201             OSTRNG: MOVB    (R2)+,R1                ;SEE IF TERMINATOR IN ASCIZ STRING,
      38 033354  001422                     BEQ     OSTRE                   ;EXIT
      39 033356  012700  033632             MOV     #ERRC,RO                ;GET POINTER TO CHARACTER TABLE
      40 033362  120110             NCONS:  CMPB    R1,(RO)                 ;COMPARE CHARACTER WITH TABLE ENTRY
      41 033364  001410                     BEQ     NCONF                   ;BRANCH IF MATCH FOUND
      42 033366  105720                     TSTB    (RO)+                   ;INCREMENT POINTER
      43 033370  001374                     BNE     NCONS                   ;CONTINUE SEARCH IF NOT END OF TABLE
      44 033372  012746  014427             MOV     #ERRME1,-(SP)           ;PUSH #ERRME1 ON STACK
         033376  004137  033244             JSR     R1,LPNTF                     ;CALL LPNTF PRINT ROUTINE
         033402  000002                     .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
      45 033404  000406                     BR      OSTRE
      46 033406  162700  033632     NCONF:  SUB     #ERRC,RO                ;GET INCREMENT INTO TABLE
      47 033412  006300                     ASL     RO                      ;DOUBLE TO WORD COUNT
      48 033414  004770  033644             JSR     PC,@ERRD(RO)            ;DISPATCH TO PRINT ROUTINE
      49 033420  000754                     BR      OSTRNG                  ;GET NEXT
      50 033422  000207             OSTRE:  RTS     PC
```

```
 1
 2                                         ;CONTROL CHARACTER WAS A QUOTE, SO  PRINT ALL CHARACTERS TO
 3                                         ;THE NEXT QUOTE.
 4
 5 033424  112200            CON.QU: MOVB    (R2)+,R0              ;GET CHARACTER
 6 033426  120027  000042            CMPB    R0,@'"               ;CHECK IF ENDING QUOTE
 7 033432  001403                    BEQ     CON.QX               ;IF SO, GO GET NEXT CONTROL CHARACTER
 8 033434  004737  033066            JSR     PC,PRINTC            ;PRINT THE CHARACTER.
 9 033440  000771                    BR      CON.QU               ;CONTINUE PRINTING
10 033442  000207            CON.QX: RTS     PC
11
12                                         ;CONTROL CHARACTER WAS AN 'A', SO PRINT ASCII CHARACTERS FROM
13                                         ;PARAMETERS.
14
15 033444  004737  033664    CON.A:  JSR     PC,GETCNT            ;GET COUNT OF CHARACTERS
16 033450  012401                    MOV     (R4)+,R1             ; GET STRING ADDRESS
17 033452  105711            CON.A1: TSTB    (R1)                 ; TEST FOR NULL
18 033454  001404                    BEQ     CON.A2               ; BRANCH IF YES
19 033456  112100                    MOVB    (R1)+,R0                     ;STORE (R1)+ IN R0 AND
   033460  004737  033066            JSR     PC,PRINTC            ;PRINT THE CHARACTER.
20 033464  000772                    BR      CON.A1               ; CONTINUE
21 033466  000207            CON.A2: RTS     PC                   ;NOW GET NEXT CONTROL CHARACTER
22
23                                         ;CONTROL CHARACTER WAS A 'D', SO PRINT A DECIMAL NUMBER.
24
25 033470  012701  000012    CON.D:  MOV     #10.,R1              ;LOAD RADIX
26 033474  004737  033742            JSR     PC,PNTNUM            ;PRINT NUMBER
27 033500  000207                    RTS     PC                   ;NOW GET NEXT CONTROL CHARACTER
28
29                                         ;CONTROL CHARACTER WAS AN 'H', SO PRINT A HEX NUMBER.
30
31 033502  012701  000020    CON.H:  MOV     #16.,R1              ;LOAD RADIX
32 033506  004737  033742            JSR     PC,PNTNUM            ;PRINT NUMBER
33 033512  000207                    RTS     PC                   ;NOW GET NEXT CONTROL CHARACTER
```

```
  1
  2                                          ;CONTROL CHARACTER WAS AN 'O', SO PRINT AN OCTAL NUMBER.
  3
  4 033514  012701  000010        CON.O:   MOV     #8.,R1                ;LOAD RADIX
  5 033520  004737  033742                 JSR     PC,PNTNUM             ;PRINT NUMBER
  6 033524  000207                          RTS     PC                    ;NOW GET NEXT CONTROL CHARACTER
  7
  8                                          ;CONTROL CHARACTER WAS AN 'N', SO PRINT A CARRIAGE RETURN-LINE FEED.
  9
 10 033526  004737  033664        CON.N:   JSR     PC,GETCNT             ;GET COUNT
 11 033532                        CON.N1:
    033532  112700  000015                 MOVB    #CR,RO                ;STORE #CR IN RO AND
    033536  004737  033066                 JSR     PC,PRINTC             ;PRINT THE CHARACTER.
 12 033542  005301                          DEC     R1                    ;COUNT THE SEQUENCES
 13 033544  001372                          BNE     CON.N1
 14 033546  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 033550  004737  033664        CON.R:   JSR     PC,GETCNT             ;GET ROUTINE NUMBER
 20 033554  020127  000010                 CMP     R1,#ERR.SZ            ;CHECK IF DEFINED ROUTINE NUMBER
 21 033560  101004                          BHI     CON.R1
 22 033562  060101                          ADD     R1,R1                 ;DOUBLE COUNT TO GET WORD INDEX
 23 033564  004771  016132                 JSR     PC,@ERR.TB-2(R1)      ;CALL ROUTINE
 24 033570  000207                          RTS     PC                    ;NOW GET NEXT CONTROL CHARACTER
 25
 26 033572                        CON.R1:
    033572  012746  014427                 MOV     #ERRME1,-(SP)         ;PUSH #ERRME1 ON STACK
    033576  004137  033244                 JSR     R1,LPNTF                      ;CALL LPNTF PRINT ROUTINE
    033602  000002                          .WORD   ARG.CT                ;ARGUMENT COUNT * 2
 27 033604  012601                          MOV     (SP)+,R1              ;;POP STACK INTO R1
 28 033606  000207                          RTS     PC
 29
 30                                          ;CONTROL CHARACTER WAS AN 'S', SO PRINT SOME NUMBER OF SPACES.
 31
 32 033610  004737  033664        CON.S:   JSR     PC,GETCNT             ;GET COUNT
 33 033614                        CON.S1:
    033614  112700  000040                 MOVB    #' ,RO                ;STORE #'  IN RO AND
    033620  004737  033066                 JSR     PC,PRINTC             ;PRINT THE CHARACTER.
 34 033624  005301                          DEC     R1                    ;COUNT THE SPACES
 35 033626  001372                          BNE     CON.S1
 36 033630  000207                          RTS     PC                    ;NOW GET NEXT CONTROL CHARACTER
 37
```

```
 1                                      .SBTTL  FAO VECTOR TABLE
 2                                      ;BUILD TWO TABLES
 3                                      ;        FIRST CONTAINING CONTROL CHARACTERS
 4                                      ;        SECOND CONTAINING ROUTINE ADDRESSES
 5
22
23                                      ;       ERRC:   CONTROL CHARACTER TABLE
24
34
35 033632                              ERRC:
   033632        042                            .BYTE   '"
   033633        101                            .BYTE   'A
   033634        104                            .BYTE   'D
   033635        110                            .BYTE   'H
   033636        117                            .BYTE   'O
   033637        116                            .BYTE   'N
   033640        122                            .BYTE   'R
   033641        123                            .BYTE   'S
36 033642        000                            .BYTE   0                    ;FOLLOW WITH A NULL BYTE
37                                             .EVEN
38
39                                      ;       ERRD:   PROCESSING ROUTINE TABLE
40
50
51 033644                              ERRD:
   033644        033424                         .WORD   CON.QU
   033646        033444                         .WORD   CON.A
   033650        033470                         .WORD   CON.D
   033652        033502                         .WORD   CON.H
   033654        033514                         .WORD   CON.O
   033656        033526                         .WORD   CON.N
   033660        033550                         .WORD   CON.R
   033662        033610                         .WORD   CON.S
52
```

```
    1                                        .SBTTL  GETCNT   GET CHARACTER COUNT
    2                                        ;GETCNT
    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 033664                        GETCNT:
      033664  010046                         MOV    RO,-(SP)                 ;;PUSH RO ON STACK
   15 033666  005001                         CLR    R1                       ;START WITH ZERO COUNT
   16 033670  121227  000060       GETCNX: CMPB    (R2),#'0                  ;CHECK IF CHARACTER A DIGIT
   17 033674  103415                         BLO    GETCDN                   ;BRANCH IF LOWER THAN ZERO
   18 033676  121227  000071                 CMPB   (R2),#'9
   19 033702  101012                         BHI    GETCDN                   ;BRANCH IF HIGHER THAN NINE
   20 033704  006301                         ASL    R1                       ;MULTIPLY NUMBER BY 10
   21 033706  010100                         MOV    R1,RO                    ; SAVE 2N
   22 033710  006301                         ASL    R1                       ; COMPUTE 4N
   23 033712  006301                         ASL    R1                       ; COMPUTE 8N
   24 033714  060001                         ADD    RO,R1                    ; 8N + 2N = 10N
   25 033716  112200                         MOVB   (R2)+,RO                 ;GET DIGIT FROM STING
   26 033720  162700  000060                 SUB    #'0,RO                   ;GET RID OF ASCII
   27 033724  060001                         ADD    RO,R1                    ;ADD TO NUMBER
   28 033726  000760                         BR     GETCNX                   ;GO TO NEXT CHARACTER
   29 033730  005701               GETCDN: TST     R1                       ;CHECK IF NUMBER IS ZERO
   30 033732  001001                         BNE    GETCXX                   ;IF ZERO, CHANGE
   31 033734  005201                         INC    R1                       ; TO DEFAULT OF ONE
   32 033736                        GETCXX:
      033736  012600                         MOV    (SP)+,RO                 ;;POP STACK INTO RO
   33 033740  000207                         RTS    PC
```

```
    1                                    .SBTTL  PNTNUM  PRINT NUMBER
    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                                 ;       RO - CONTENTS DESTROYED
   14
   15 033742 010100          PNTNUM: MOV     R1,RO                   ;SAVE RADIX
   16 033744 004737 033664           JSR     PC,GETCNT               ;GET COUNT OF BITS
   17 033750                 PNTNUS:
      033750 010246                  MOV     R2,-(SP)                ;;PUSH R2 ON STACK
      033752 010346                  MOV     R3,-(SP)                ;;PUSH R3 ON STACK
      033754 010546                  MOV     R5,-(SP)                ;;PUSH R5 ON STACK
   18 033756 012403                  MOV     (R4)+,R3                ;GET ONE PARAMETER WORD
   19 033760 005005                  CLR     R5                      ;CLEAR STORAGE FOR OTHER
   20 033762 020127 000020           CMP     R1,#16.                 ;MORE THAN 16 BITS IN NUMBER?
   21 033766 003401                  BLE     1$
   22 033770 012405                  MOV     (R4)+,R5                ;YES, GET SECOND PARAMETER WORD
   23 033772                 1$:
      033772 010446                  MOV     R4,-(SP)                ;;PUSH R4 ON STACK
   24 033774 010504                  MOV     R5,R4                   ;PUT HIGH WORD IN R4
   25 033776 012702 000020           MOV     #16.,R2                 ;COMPUTE BITS NOT WANTED
   26 034002 160102                  SUB     R1,R2                   ;BY SUBTRACTING BITS TO USE
   27 034004 002002                  BGE     2$                      ;FROM 16.
   28 034006 062702 000020           ADD     #16.,R2                 ;IF NEGATIVE, ADD 16 FOR FIRST WORD
   29 034012 001414          2$:     BEQ     6$                      ;IF ZERO, NO BITS NEED BE CLEARED
   30 034014 012705 100000           MOV     #BIT15,R5               ;START MASK WITH SIGN BIT SET
   31 034020 005302          3$:     DEC     R2                      ;COUNT BITS IN MASK
   32 034022 001402                  BEQ     4$
   33 034024 006205                  ASR     R5                      ;SHIFT MORE BITS TO RIGHT
   34 034026 000774                  BR      3$
   35 034030 020127 000020   4$:     CMP     R1,#16.                 ;MORE THAN 16 BITS IN NUMBER?
   36 034034 003402                  BLE     5$
   37 034036 040504                  BIC     R5,R4                   ;YES, CLEAR IN HIGH WORD
   38 034040 000401                  BR      6$
   39 034042 040503          5$:     BIC     R5,R3                   ;NO, CLEAR IN LOW WORD
   40 034044 004737 034436   6$:     JSR     PC,DIVIDE               ;DIVIDE BY RADIX IN RO
   41 034050 010546                  MOV     R5,-(SP)                ;;PUSH R5 ON STACK
   42 034052 005202                  INC     R2                      ;COUNT DIGITS ON STACK
   43 034054 005703                  TST     R3                      ;CHECK IF QUOTIENT IS ZERO
   44 034056 001372                  BNE     6$
   45 034060 005704                  TST     R4
   46 034062 001370                  BNE     6$
```

```
 1 034064   020027   000012           CMP    R0,#10.              ;IF RADIX IS DECIMAL
 2 034070   001424                     BEQ    90$                  ; JUST GO PRINT DIGITS ON STACK
 3 034072   010103                     MOV    R1,R3                ;OTHERWISE COMPUTE NUMBER OF LEADING ZEROS
 4 034074   162700   000014           SUB    #12.,R0              ;DIVIDEND IS BITS IN NUMBER
 5 034100   003002                     BGT    7$                   ;DIVISOR IS BITS PER DIGIT PRINTED
 6 034102   012700   000003           MOV    #3,R0                ;   (3 OR 4)
 7 034106                      7$:
 8 034106   004737   034436           JSR    PC,DIVIDE
 9 034112   005705                     TST    R5                   ;IF REMAINDER NOT ZERO
10 034114   001401                     BEQ    8$                   ;INCREMENT QUOTIENT
11 034116   005203                     INC    R3
12 034120                      8$:
13 034120   160203                     SUB    R2,R3                ;SUBTRACT DIGITS ON STACK
14 034122   001424                     BEQ    10$                  ;NO LEADING ZEROS IF ZERO
15 034124                      9$:
16 034124   112700   000060           MOVB   #'0,R0               ;STORE #'0 IN R0 AND
   034130   004737   033066           JSR    PC,PRINTC            ;PRINT THE CHARACTER.
17 034134   005303                     DEC    R3
18 034136   001372                     BNE    9$                   ;REPEAT UNTIL COUNT REACHES ZERO
19 034140   000415                     BR     10$                  ; NOW PRINT NUMBER
20 034142                      90$:
21 034142   022701   000034           CMP    #28.,R1              ; TEST FOR D28 SPEC (LBN)
22 034146   001012                     BNE    10$                  ; BRANCH IF NO
23 034150   012703   000011           MOV    #9.,R3               ; MAX LBN FIELD SIZE IS 9.
24 034154   160203                     SUB    R2,R3                ; SUBTRACT CHARS ON STACK
25 034156   003406                     BLE    10$                  ; BRANCH IF <= ZERO
26 034160                      99$:
27 034160   112700   000040           MOVB   #' ,R0               ; LOAD BLANK
28 034164   004737   033066           JSR    PC,PRINTC            ; AND PRINT
29 034170   005303                     DEC    R3                   ; DECREMENT
30 034172   001372                     BNE    99$                  ; AND CONTINUE UNTIL ZERO
31 034174                      10$:
32 034174   012605                     MOV    (SP)+,R5             ;;POP STACK INTO R5
33 034176   062705   000060           ADD    #'0,R5               ;CNVERT TO ASCII DIGIT
34 034202   020527   000071           CMP    R5,#'9               ;IF GREATER THAN A 9
35 034206   003402                     BLE    11$                  ; CONVERT TO A OR HIGHER
36 034210   062705   000007           ADD    #<'A-'9-1>,R5        ; FOR HEX DIGIT
37 034214                      11$:
38 034214   110500                     MOVB   R5,R0                ;STORE R5 IN R0 AND
   034216   004737   033066           JSR    PC,PRINTC            ;PRINT THE CHARACTER.
39 034222   005302                     DEC    R2                   ;REPEAT FOR ALL DIGITS
40 034224   001363                     BNE    10$                  ; ON STACK
41 034226   012604                     MOV    (SP)+,R4             ;;POP STACK INTO R4
   034230   012605                     MOV    (SP)+,R5             ;;POP STACK INTO R5
   034232   012603                     MOV    (SP)+,R3             ;;POP STACK INTO R3
   034234   012602                     MOV    (SP)+,R2             ;;POP STACK INTO R2
42 034236   000207                     RTS    PC
43
```

```
     1                                    .SBTTL  CLRBUF  CLEAR BUFFER
     2                                  ;CLRBUF
     3                                  ;
     4                                  ;CLEAR A 512(10) BYTE BUFFER, STARTING AT THE UNIBUS ADDRESS CONTAINED IN
     5                                  ;THE COMMAND PACKET OF THE HOST COMM AREA.
     6                                  ;
     7                                  ;INPUTS:
     8                                  ;     R4 - ADDRESS OF HOST COMM AREA
     9                                  ;OUTPUTS:
    10                                  ;     512(10) BYTE BUFFER CLEARED
    11
    12 034240                          CLRBUF:
       034240   010046                     MOV     R0,-(SP)                    ;;PUSH R0 ON STACK
       034242   010146                     MOV     R1,-(SP)                    ;;PUSH R1 ON STACK
    13 034244   013704   002324            MOV     HCOM,R4                     ; LOAD HOST COMM AREA ADDRESS
    14 034250   016400   000124            MOV     HC.CPK+P.UADR(R4),R0        ;GET STARTING UNIBUS ADDRESS
    15 034254   013701   002334            MOV     BCNT,R1                     ;NUMBER OF BYTES TO CLEAR.
    16 034260   006201                     ASR     R1                          ;MAKE R1 WORDS AND
    17 034262   005020                1$:  CLR     (R0)+                       ;CLEAR THE ENTIRE BUFFER.
    18 034264   005301                     DEC     R1
    19 034266   001375                     BNE     1$
    20 034270   012601                     MOV     (SP)+,R1                    ;;POP STACK INTO R1
       034272   012600                     MOV     (SP)+,R0                    ;;POP STACK INTO R0
    21 034274   000207                     RTS     PC
    22
```

```
 1                              .SBTTL  COPY    COPY BUFFER
 2                              ;COPY
 3                              ;
 4                              ;COPY 512(10) BYTES OF DATA FROM ONE BUFFER TO ANOTHER BUFFER
 5                              ;
 6                              ;INPUTS:
 7                              ;       (R5) - ADDRESS OF SOURCE BUFFER
 8                              ;       2(R5) - ADDRESS OF DESTINATION BUFFER
 9                              ;OUTPUTS:
10                              ;       512(10) BYTES COPIED FROM 'SOURCE' BUFFER TO 'DESTINATION' BUFFER
11
12 034276                      COPY:
   034276  010046                      MOV     R0,-(SP)                ;;PUSH R0 ON STACK
   034300  010146                      MOV     R1,-(SP)                ;;PUSH R1 ON STACK
   034302  010246                      MOV     R2,-(SP)                ;;PUSH R2 ON STACK
13 034304  012500                      MOV     (R5)+,R0                ;GET STARTING ADDRESS OF SOURCE
14 034306  012501                      MOV     (R5)+,R1                ; AND DESTINATION BUFFERS.
15 034310  013702   002334             MOV     BCNT,R2                 ;NUMBER OF BYTES TO COPY
16 034314  006202                      ASR     R2                      ;MAKE R1 WORDS AND
17 034316  012021              1$:     MOV     (R0)+,(R1)+             ;COPY HIS LOCATION TO OTHER BUFFER
18 034320  005302                      DEC     R2
19 034322  001375                      BNE     1$
20 034324  012602                      MOV     (SP)+,R2                ;;POP STACK INTO R2
   034326  012601                      MOV     (SP)+,R1                ;;POP STACK INTO R1
   034330  012600                      MOV     (SP)+,R0                ;;POP STACK INTO R0
21 034332  000205                      RTS     R5
22
```

```
    1                                 .SBTTL  SETTO   SET TIMEOUT
    2                                 ;SETTO
    3                                 ;
    4                                 ;SET TIMEOUT COUNTER TO SOME NUMBER OF SECONDS FROM CURRENT TIME.
    5                                 ;
    6                                 ;       RO - NUMBER OF SECONDS FOR TIMEOUT
    7                                 ;       R1 - ADDRESS WHERE TWO WORD TIME TO BE PUT
    8                                 ;OUTPUTS:
    9                                 ;       RO - CONTENTS DESTROYED
   10                                 ;       R1 - INCREMENTED BY 2
   11                                 ;
   12                                 ;COMPUTE CLOCK TICKS TIL TIMEOUT
   13
   14 034334                         SETTO:
      034334  010246                         MOV     R2,-(SP)                 ;;PUSH R2 ON STACK
      034336  010346                         MOV     R3,-(SP)                 ;;PUSH R3 ON STACK
   15 034340  005002                         CLR     R2                       ;CLEAR PRODUCT
   16 034342  013703  002216                 MOV     KW.HZ,R3                 ;GET MULTIPLICAND
   17 034346  006200                 1$:     ASR     RO                       ;SHIFT MULTIPLIER TO RIGHT
   18 034350  103001                         BCC     2$                       ;IF A ONE BIT SHIFTED OUT
   19 034352  060302                         ADD     R3,R2                    ;ADD MULTIPLICAND TO PRODUCT
   20 034354  006303                 2$:     ASL     R3                       ;DOUBLE THE MULTIPLICAND
   21 034356  005700                         TST     RO
   22 034360  001372                         BNE     1$                       ;CONTINUE UNTIL MULTIPLIER IS ZERO
   23
   24                                         ;GET CURRENT TIME
   25
   26 034362  013700  002220         3$:     MOV     KW.EL,RO                 ;GET TIME
   27 034366  013703  002222                 MOV     KW.EL+2,R3
   28 034372  020037  002220                 CMP     RO,KW.EL                 ;IF CHANGED DURING RETRIEVAL
   29 034376  001371                         BNE     3$                       ;GET IT AGAIN
   30
   31                                         ;ADD TIME TIL TIMEOUT
   32
   33 034400  060200                         ADD     R2,RO                    ;ADD
   34 034402  005503                         ADC     R3
   35
   36                                         ;PUT RESULT IN STORAGE
   37
   38 034404  010021                         MOV     RO,(R1)+
   39 034406  010311                         MOV     R3,(R1)
   40 034410  012603                         MOV     (SP)+,R3                 ;;POP STACK INTO R3
      034412  012602                         MOV     (SP)+,R2                 ;;POP STACK INTO R2
   41 034414  000207                         RTS     PC
```

```
    1                                     .SBTTL  CLOG    COMPUTE BASE 2 LOG
    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 034416                     CLOG:
      034416   010046                     MOV     RO,-(SP)                ;;PUSH RO ON STACK
   12 034420   005001                     CLR     R1                      ;SET UP ZERO START VALUE
   13 034422   000261                     SEC                             ;WITH CARRY READY TO SHIFT IN
   14 034424   006101             1$:     ROL     R1                      ;SHIFT TO LEFT
   15 034426   005300                     DEC     RO                      ;UNTIL RO
   16 034430   100375                     BPL     1$                      ;GOES NEGATIVE
   17 034432   012600                     MOV     (SP)+,RO                ;;POP STACK INTO RO
   18 034434   000207                     RTS     PC
```

```
    1                              .SBTTL  DIVIDE   DIVIDE NUMBER
    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 BY ZERO.
    7                              ;
    8                              ;INPUTS:
    9                              ;       R3 - LOW 16 BITS OF DIVIDEND
   10                              ;       R4 - HIGH 16 BITS OF DIVIDEND
   11                              ;       R0 - DIVISOR
   12                              ;OUTPUTS:
   13                              ;       R3 - LOW 16 BITS OF QUOTIENT
   14                              ;       R4 - HIGH 16 BITS OF QUOTIENT
   15                              ;       R5 - REMAINDER
   16
   17 034436                      DIVIDE:
      034436 010246                       MOV     R2,-(SP)        ;;PUSH R2 ON STACK
   18 034440 012702  000040                MOV     #32.,R2         ;SET UP SHIFT COUNT
   19 034444 005005                       CLR     R5              ;START WITH ZERO REMAINDER
   20 034446 006303               1$:     ASL     R3              ;SHIFT LEFT INTO R5
   21 034450 006104                       ROL     R4
   22 034452 006105                       ROL     R5
   23 034454 020005                       CMP     R0,R5           ;WILL DIVISOR GO INTO REMAINDER
   24 034456 101002                       BHI     2$              ;ONLY SUBTRACT IF IT WILL
   25 034460 160005                       SUB     R0,R5           ;SUBTRACT DIVISOR
   26 034462 005203                       INC     R3              ;PUT A ONE INTO QUOTIENT
   27 034464 005302               2$:     DEC     R2              ;COUNT THE SHIFTS
   28 034466 001367                       BNE     1$
   29 034470 012602                       MOV     (SP)+,R2        ;;POP STACK INTO R2
   30 034472 000207                       RTS     PC
```

```
                                      .SBTTL DIV10    DIVIDE BY TEN
    1                                 ;DIV10
    2
    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 034474                  DIV10:
      034474  010046               MOV    R0,-(SP)         ;;PUSH R0 ON STACK
   21 034476  012700  000100       MOV    #64.,R0          ;SET UP SHIFT COUNT
   22 034502  005005               CLR    R5               ;START WITH ZERO REMAINDER
   23 034504  006301         1$:   ASL    R1
   24 034506  006102               ROL    R2               ;SHIFT LEFT INTO R5
   25 034510  006103               ROL    R3
   26 034512  006104               ROL    R4
   27 034514  006105               ROL    R5
   28 034516  022705  000012       CMP    #10.,R5          ;SILL DIVISOR GO INTO REMAINDER?
   29 034522  101003               BHI    2$               ;ONLY SUBTRACT IF IT WILL
   30 034524  162705  000012       SUB    #10.,R5          ;SUBTRACT DIVISOR
   31 034530  005201               INC    R1               ;PUT A ONE INTO QUOTIENT
   32 034532  005300         2$:   DEC    R0               ;COUNT THE SHIFTS
   33 034534  001363               BNE    1$
   34 034536  012600               MOV    (SP)+,R0         ;;POP STACK INTO R0
   35 034540  000207               RTS    PC               ;  <R4,R3,R2,R1> AND REMAINDER IN R5
```

```
   1                              .SBTL   MULT     INTEGER MULTIPLY ROUTINE
   2                              ;MULT
   3                              ;
   4                              ;CALL
   5                              ;        MOV     MULTIPLER,-(SP)
   6                              ;        MOV     MULTIPLICAND,-(SP)
   7                              ;        JSR     PC,MULT
   8                              ;        RETURN                           ;PRODUCT IS ON THE STACK
   9                              ;
  10                              ;        STACK   PRODUCT
  11                              ;        -----   -------
  12                              ;        TOP     LSB'S
  13                              ;        +2      MSB'S
  14
  15 034542                      MULT:
     034542  010046                      MOV     R0,-(SP)                 ;;PUSH R0 ON STACK
     034544  010146                      MOV     R1,-(SP)                 ;;PUSH R1 ON STACK
     034546  010246                      MOV     R2,-(SP)                 ;;PUSH R2 ON STACK
  16 034550  005046                      CLR     -(SP)                    ;CLEAR THE SIGN KEY
  17 034552  016601  000012              MOV     12(SP),R1                ;GET THE MULTIPLICAND
  18 034556  100002                      BPL     1$                       ;BR IF PLUS
  19 034560  005216                      INC     (SP)                     ;SET THE SIGN KEY
  20 034562  005401                      NEG     R1                       ;MAKE THE MULTIPLICAND POSTIVE
  21 034564  016602  000014      1$:     MOV     14(SP),R2                ;GET THE MULTIPLIER
  22 034570  100002                      BPL     2$                       ;BR IF PLUS
  23 034572  005316                      DEC     (SP)                     ;UPDATE THE SIGN KEY
  24 034574  005402                      NEG     R2                       ;MAKE THE MULTIPLIER POSTIVE
  25 034576  012746  000021      2$:     MOV     #17.,-(SP)               ;SET THE LOOP COUNT
  26 034602  005000                      CLR     R0                       ;SETUP FOR THE MULTIPLY LOOP
  27 034604  103001              3$:     BCC     4$                       ;DON'T ADD IF MULTIPLICAND = 0
  28 034606  060200                      ADD     R2,R0
  29 034610  006000              4$:     ROR     R0                       ;POSITION THE PARITIAL PRODUCT AND
  30 034612  006001                      ROR     R1                       ;THE MULTIPLICAND
  31 034614  005316                      DEC     (SP)                     ;HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
  32 034616  001372                      BNE     3$                       ;BR IF NO
  33 034620  022616                      CMP     (SP)+,(SP)               ;SHOULD PRODUCT BE NEGATIVE?
  34 034622  001403                      BEQ     5$                       ;GO TO EXIT IF NO
  35 034624  005400                      NEG     R0                       ;YES--SO MAKE IT SO
  36 034626  005401                      NEG     R1
  37 034630  005600                      SBC     R0
  38 034632  005726              5$:     TST     (SP)+                    ;CLEAR SIGN INFO. OFF OF STACK
  39 034634  010066  000012              MOV     R0,12(SP)                ;PUT THE PRODUCT ON THE STACK (MSB'S)
  40 034640  010156  000010              MOV     R1,10(SP)                ;LSB'S
  41 034644  012602                      MOV     (SP)+,R2                 ;;POP STACK INTO R2
     034646  012601                      MOV     (SP)+,R1                 ;;POP STACK INTO R1
     034650  012600                      MOV     (SP)+,R0                 ;;POP STACK INTO R0
  42 034652  000207                      RTS     PC
```

```
 1                              .SBTTL  BLD28    BUILD 28-BIT NUMBER
 2                              ;BLD28
 3                              ;
 4                              ;BUILD DEFAULT 28-BIT NUMBER
 5                              ;
 6                              ;INPUT:
 7                              ;       R4 - ADDRESS OF LO ORDER WORD OF DEFAULT NUMBER
 8                              ;OUTPUT:
 9                              ;       TTYIN - ADDRESS OF ASCIZ STRING INPUT BUFFER
10
11 034654                      BLD28:
   034654  010046                      MOV     R0,-(SP)                ;;PUSH R0 ON STACK
   034656  010146                      MOV     R1,-(SP)                ;;PUSH R1 ON STACK
   034660  010346                      MOV     R3,-(SP)                ;;PUSH R3 ON STACK
   034662  010446                      MOV     R4,-(SP)                ;;PUSH R4 ON STACK
   034664  010546                      MOV     R5,-(SP)                ;;PUSH R5 ON STACK
12 034666  011403                      MOV     (R4),R3                 ;GET NUMBER
13 034670  016404  000002              MOV     2(R4),R4
14 034674  012700  000012              MOV     #10.,R0                 ;DIVISOR IS 10.
15 034700  005001                      CLR     R1                      ;CLEAR CHARACTER COUNT
16 034702  004737  034436      1$:     JSR     PC,DIVIDE
17 034706  062705  000060              ADD     #'0,R5                  ;CONVERT REMAINDER TO ASCII CHARACTER
18 034712  010546                      MOV     R5,-(SP)                ;;PUSH R5 ON STACK
19 034714  005201                      INC     R1                      ;COUNT THE CHARACTER
20 034716  010305                      MOV     R3,R5                   ;REPEAT UNTIL QUOTIENT IS ZERO
21 034720  050405                      BIS     R4,R5
22 034722  001367                      BNE     1$
23 034724  012700  002666              MOV     #TTYIN,R0               ;GET POINTER TO STRING
24 034730                      2$:
   034730  012605                      MOV     (SP)+,R5                ;;POP STACK INTO R5
25 034732  110520                      MOVB    R5,(R0)+
26 034734  005301                      DEC     R1
27 034736  001374                      BNE     2$
28 034740  105020                      CLRB    (R0)+                   ;END WITH NULL
29 034742  012605                      MOV     (SP)+,R5                ;;POP STACK INTO R5
   034744  012604                      MOV     (SP)+,R4                ;;POP STACK INTO R4
   034746  012603                      MOV     (SP)+,R3                ;;POP STACK INTO R3
   034750  012601                      MOV     (SP)+,R1                ;;POP STACK INTO R1
   034752  012600                      MOV     (SP)+,R0                ;;POP STACK INTO R0
30 034754  000207                      RETURN
```

```
    1                                 .SBTTL  CNV28    CONVERT ASCIZ TO 28-BIT BINARY
    2                                 ;CNV28
    3                                 ;
    4                                 ;CONVERT ASCIZ STRING TO 28-BIT NUMBER
    5                                 ;
    6                                 ;INPUTS:
    7                                 ;       TTYIN - ASCIZ STRING INPUT UP TO 9. CHARACTERS
    8                                 ;       R4 - ADDRESS OF LO ORDER WORD OF STORAGE LOCATION
    9                                 ;OUTPUTS:
   10                                 ;       IF STRING IS VALID NUMBER
   11                                 ;          TWO WORDS AT R4 LOADED WITH NUMBER
   12                                 ;          CARRY CLEAR
   13                                 ;       IF STRING INVALID
   14                                 ;          ERROR MESSAGE PRINTED
   15                                 ;          CARRY SET
   16
   17 034756                 CNV28:
      034756  010046                  MOV     R0,-(SP)                ;;PUSH R0 ON STACK
      034760  010146                  MOV     R1,-(SP)                ;;PUSH R1 ON STACK
      034762  010246                  MOV     R2,-(SP)                ;;PUSH R2 ON STACK
      034764  010346                  MOV     R3,-(SP)                ;;PUSH R3 ON STACK
   18 034766  005000                  CLR     R0                      ;CLEAR LO ORDER WORD
   19 034770  005001                  CLR     R1                      ;CLEAR HI ORDER WORD
   20 034772  012702  002666          MOV     #TTYIN,R2               ;GET BEGINNING ADDRESS OF TYPED INPUT
   21 034776  112203          1$:     MOVB    (R2)+,R3                ;GET A CHARACTER FROM ASCIZ LINE
   22 035000  001443                  BEQ     4$                      ;IF NULL CHARACTER, ALL DONE
   23 035002  162703  000060          SUB     #'0,R3                  ;SUBTRACT CHARACTER 0
   24 035006  100431                  BMI     2$
   25 035010  022703  000011          CMP     #9.,R3
   26 035014  103426                  BLO     2$
   27 035016  006300                  ASL     R0                      ;*2
   28 035020  006101                  ROL     R1
   29 035022  010146                  MOV     R1,-(SP)                ;;PUSH R1 ON STACK
      035024  010046                  MOV     R0,-(SP)                ;;PUSH R0 ON STACK
   30 035026  006300                  ASL     R0                      ;*4
   31 035030  006101                  ROL     R1
   32 035032  006300                  ASL     R0                      ;*8.
   33 035034  006101                  ROL     R1
   34 035036  062600                  ADD     (SP)+,R0                ;*10.
   35 035040  005501                  ADC     R1
   36 035042  062601                  ADD     (SP)+,R1
   37 035044  060300                  ADD     R3,R0                   ;ADD CURRENT DIGIT
   38 035046  005501                  ADC     R1
   39 035050  032701  170000          BIT     #170000,R1              ;CHECK SIZE OF NUMBER
   40 035054  001750                  BEQ     1$                      ;MUST NOT BE MORE THAN 28 BITS
   41                                                                 ;PRINT ' ?EXCEEDED 28 BIT INPUT LIMIT'
   42 035056  012746  014277          MOV     #LMT28,-(SP)            ;PUSH #LMT28 ON STACK
      035062  004137  033244          JSR     R1,LPNTF                    ;CALL LPNTF PRINT ROUTINE
      035066  000002                  .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
   43 035070  000405                  BR      3$
   44                                                                 ;PRINT ' ?INVALID CHARACTER'
   45 035072                  2$:
      035072  012746  014340          MOV     #INVCHR,-(SP)           ;PUSH #INVCHR ON STACK
      035076  004137  033244          JSR     R1,LPNTF                    ;CALL LPNTF PRINT ROUTINE
      035102  000002                  .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
   46 035104  000261          3$:     SEC                             ;SET CARRY FOR ERROR
   47 035106  000404                  BR      5$
```

```
48
49 035110  010014            4$:      MOV     R0,(R4)             ;MOVE LO ORDER WORD AND
50 035112  010164  000002             MOV     R1,2(R4)            ;HI ORDER WORD TO STORAGE AREA.
51 035116  000241                      CLC                         ;CLEAR CARRY FOR NO ERROR
52 035120                     5$:
   035120  012603                      MOV     (SP)+,R3            ;;POP STACK INTO R3
   035122  012602                      MOV     (SP)+,R2            ;;POP STACK INTO R2
   035124  012601                      MOV     (SP)+,R1            ;;POP STACK INTO R1
   035126  012600                      MOV     (SP)+,R0            ;;POP STACK INTO R0
53 035130  000207                      RETURN
                              .SBTTL   REPORT CODING SECTION
54
55
57                           ;++
58                           ; THE REPORT CODING SECTION CONTAINS THE
59                           ; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
60                           ;--
61
62 035132                    L$RPT::
63
64 035132  013705  002304             MOV     CTLRTBL,R5          ; LOAD TABLE PTR
65 035136  013704  002324             MOV     HCOM,R4             ; LOAD COMM PTR
66 035142                     10$:                                ; "REPLACEMENT INFORMATION..."
67 035142  012746  005271             MOV     #INF200,-(SP)       ;PUSH #INF200 ON STACK
   035146  004137  033254             JSR     R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
   035152  000002                     .WORD   ARG.CT              ;ARGUMENT COUNT * 2
68 035154  013746  002542             MOV     WFECNT,-(SP)        ;PUSH WFECNT ON STACK
   035160  013746  002530             MOV     FERCNT,-(SP)        ;PUSH FERCNT ON STACK
   035164  013746  002536             MOV     ECCCNT,-(SP)        ;PUSH ECCCNT ON STACK
   035170  013746  002534             MOV     DSTCNT,-(SP)        ;PUSH DSTCNT ON STACK
   035174  013746  002532             MOV     HCECNT,-(SP)        ;PUSH HCECNT ON STACK
   035200  013746  002540             MOV     BBRCNT,-(SP)        ;PUSH BBRCNT ON STACK
   035204  012746  005343             MOV     #INF201,-(SP)       ;PUSH #INF201 ON STACK
   035210  004137  033254             JSR     R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
   035214  000016                     .WORD   ARG.CT              ;ARGUMENT COUNT * 2
69
70 035216  013703  002434             MOV     PRIMRY,R3           ; LOAD PRIMARY COUNT
71 035222  063703  002436             ADD     SCNDRY,R3           ; ADD SECONDARY COUNT
72
73 035226  010346                     MOV     R3,-(SP)            ;PUSH R3 ON STACK
   035230  013746  002440             MOV     UNUSE,-(SP)         ;PUSH UNUSE ON STACK
   035234  013746  002436             MOV     SCNDRY,-(SP)        ;PUSH SCNDRY ON STACK
   035240  013746  002434             MOV     PRIMRY,-(SP)        ;PUSH PRIMRY ON STACK
   035244  012746  005725             MOV     #INF202,-(SP)       ;PUSH #INF202 ON STACK
   035250  004137  033254             JSR     R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
   035254  000012                     .WORD   ARG.CT              ;ARGUMENT COUNT * 2
74
75 035256  005737  002650             TST     DEBUG               ; TEST DEBUG FLAG
76 035262  001007                     BNE     100$                ; BRANCH IF YES
77
78 035264  012746  004731             MOV     #INF103,-(SP)       ;PUSH #INF103 ON STACK
   035270  004137  033254             JSR     R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
   035274  000002                     .WORD   ARG.CT              ;ARGUMENT COUNT * 2
79 035276  000167                     .WORD   J$JMP
   035300  000414                     .WORD   L10017-2-.
80
```

```
   1
   2                                        ;       DEBUG REPORTING
   3 035302                                 100$:
   4
   5                                        ;       PRINT RCT ERROR TABLE
   6
   7 035302  013746  002312                         MOV     RCTTBL,-(SP)            ;PUSH RCTTBL ON STACK
     035306  012746  035470                         MOV     #DBM01,-(SP)            ;PUSH #DBM01 ON STACK
     035312  004137  033254                         JSR     R1,LPNTB                     ;CALL LPNTB PRINT ROUTINE
     035316  000004                                 .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
   8
   9 035320  005003                                 CLR     R3
  10 035322  013702  002312                         MOV     RCTTBL,R2
  11 035326                                 110$:
  12 035326  011201                                 MOV     (R2),R1
  13 035330  001411                                 BEQ     120$
  14
  15 035332  010146                                 MOV     R1,-(SP)                ;PUSH R1 ON STACK
     035334  012746  000000                         MOV     #0,-(SP)                ;PUSH #0 ON STACK
     035340  010346                                 MOV     R3,-(SP)                ;PUSH R3 ON STACK
     035342  012746  035556                         MOV     #DBM02,-(SP)            ;PUSH #DBM02 ON STACK
     035346  004137  033254                         JSR     R1,LPNTB                     ;CALL LPNTB PRINT ROUTINE
     035352  000010                                 .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
  16 035354                                 120$:
  17 035354  062702  000002                         ADD     #2,R2
  18 035360  005203                                 INC     R3
  19 035362  023703  002416                         CMP     RCTSIZ,R3
  20 035366  003357                                 BGT     110$
  21
  22                                        ;       PRINT CACHE TABLE CONTENTS
  23
  24 035370  013746  002310                         MOV     CACHTBL,-(SP)           ;PUSH CACHTBL ON STACK
     035374  012746  035601                         MOV     #DBM03,-(SP)            ;PUSH #DBM03 ON STACK
     035400  004137  033254                         JSR     R1,LPNTB                     ;CALL LPNTB PRINT ROUTINE
     035404  000004                                 .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
  25
  26 035406  005003                                 CLR     R3
  27 035410  013704  002310                         MOV     CACHTBL,R4
  28 035414                                 130$:
  29 035414  016402  000002                         MOV     2(R4),R2
  30 035420  100413                                 BMI     140$
  31
  32 035422  011401                                 MOV     (R4),R1
  33 035424  010246                                 MOV     R2,-(SP)                ;PUSH R2 ON STACK
     035426  010146                                 MOV     R1,-(SP)                ;PUSH R1 ON STACK
     035430  012746  000000                         MOV     #0,-(SP)                ;PUSH #0 ON STACK
     035434  010346                                 MOV     R3,-(SP)                ;PUSH R3 ON STACK
     035436  012746  035667                         MOV     #DBM04,-(SP)            ;PUSH #DBM04 ON STACK
     035442  004137  033254                         JSR     R1,LPNTB                     ;CALL LPNTB PRINT ROUTINE
     035446  000012                                 .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
  34 035450                                 140$:
  35 035450  062704  000004                         ADD     #4,R4
  36 035454  005203                                 INC     R3
  37 035456  022703  000620                         CMP     #CACHESIZE,R3
  38 035462  003354                                 BGT     130$
  39
  40 035464  000167                                 .WORD   J$JMP
```

```
    035466  000226                     .WORD   L10017-2-.
 41                                     .NLIST  BEX
 42
 43 035470      116     062     042 DBM01:  .ASCII  /N2"RCT ERROR TABLE:   "016/
 44 035521      116     062     042         .ASCIZ  /N2"  RCT BLOCK.     CONTENTS"/
 45 035556      116     042     040 DBM02:  .ASCIZ  /N"  "D28".      "016/
 46
 47 035601      116     062     042 DBM03:  .ASCII  /N2"CACHING TABLE:   "016/
 48 035630      116     062     042         .ASCIZ  /N2"   ENTRY NO.      BLOCK NO."/
 49 035667      116     042     040 DBM04:  .ASCIZ  /N"  "D28".     "D28"."/
 50                                     .EVEN
 51
 52 035716                         L10017:
    035716  104425                     TRAP    C$RPT
```

```
  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 035720                            L$PROT::
 10
 11 035720  177777                            -1                    ;OFFSET INTO P-TABLE FOR CSR ADDRESS
 12 035722  177777                            -1                    ;OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
 13 035724  177777                            -1                    ;OFFSET INTO P-TABLE FOR DRIVE NUMBER
 14
 16
```

```
  1                                 .SBTTL   INITIALIZATION SECTION
  2
  3                                 ;++
  4                                 ; THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
  5                                 ; AT THE BEGINNING OF EACH PASS.
  6                                 ;--
  7
  8                                 ;;****************************************************************
  9                                 ;    RESET THE WORLD
 10                                 ;    IF HERE FROM START, RESTART, OR NEW PASS COMMAND
 11                                 ;      THEN
 12                                 ;           INITIALIZE LOGICAL UNIT NUMBER
 13                                 ;      ENDIF
 14                                 ;    SELECT A LOGICAL UNIT FOR TEST
 15                                 ;    ESTABLISH FREE MEMORY
 16                                 ;    INITIALIZE TABLES
 17                                 ;    INITIALIZE KW11 CLOCK
 18                                 ;    PRINT DRIVE ON-LINE
 19                                 ;    PRINT NUMBER LOGICAL BLOCKS IN DRIVE
 20                                 ;    END
 21                                 ;;****************************************************************
 24        000001                 I$SECTION            = 1
 25
 26 035726                        L$INIT::                                   ; RESET SYSTEM
 27
 28 035726 104433                          TRAP    C$RESET
 29 035730                        10$:
 30 035730 005737 002202                   TST     LOADFLG            ; TEST FOR FIRST TIME THROUGH
 31 035734 001020                          BNE     20$                ; BRANCH IF NO
 32 035736 005237 002202                   INC     LOADFLG            ; ELSE, SET FLAG
 33 035742                        15$:                                ; RESET START OF FREE MEMORY
 34 035742 104431                          TRAP    C$MEM
    035744 010037 002176                   MOV     R0,FFREE
 35 035750 017737 144222 002200            MOV     @FFREE,FSIZE       ; RESET SIZE OF FREE MEMORY
 36 035756                        18$:                                ; READ BUS TYPE
 37 035756 104407                          TRAP    C$RDBU
 38 035760 103006                          BCC     20$
 39
 40 035762 012737 000001 002614            MOV     #1,BUSFLAG         ; SET FLAG FOR QBUS
 41 035770 112737 000113 014225            MOVB    #'K,DEVNAME        ; SET CONTROLLER "KDA"
 42
```

```
     1                                    .SBTTL  INITIALIZE THE KW11
     2
     3 035776                             20$:
     5 035776  005037  002220                   CLR     KW.EL               ;CLEAR ELAPSED TIME
     6 036002  005037  002222                   CLR     KW.EL+2
     7 036006  012700  000114                   MOV     #'L,R0
       036012  104462                           TRAP    C$CLCK
     8 036014  103414                           BCS     25$
     9 036016  012700  000120                   MOV     #'P,R0
       036022  104462                           TRAP    C$CLCK
    10 036024  103410                           BCS     25$
    11 036026  005037  002210                   CLR     KW.CSR              ;IF NEITHER, CLEAR CSR STORAGE WORD
    12 036032  012746  004546                   MOV     #INF101,-(SP)       ;PUSH #INF101 ON STACK
       036036  004137  033244                   JSR     R1,LPNTF                    ;CALL LPNTF PRINT ROUTINE
       036042  000002                           .WORD   ARG.CT              ;ARGUMENT COUNT * 2
    13 036044  000426                           BR      30$
    14 036046                             25$:
    15 036046  012037  002210                   MOV     (R0)+,KW.CSR        ;STORE DATA RETURNED
    16 036052  012037  002212                   MOV     (R0)+,KW.BRL
    17 036056  012037  002214                   MOV     (R0)+,KW.VEC
    18 036062  012037  002216                   MOV     (R0)+,KW.HZ
    19 036066  012746  000340                   MOV     #PRIO7,-(SP)
       036072  012746  016174                   MOV     #KW11I,-(SP)
       036076  013746  002214                   MOV     KW.VEC,-(SP)
       036102  012746  000003                   MOV     #3,-(SP)
       036106  104437                           TRAP    C$SVEC
       036110  062706  000010                   ADD     #10,SP
    20 036114  012777  000105  144066           MOV     #KW.OUT,@KW.CSR     ;START THE CLOCK
    21
    22 036122                             30$:
    23
```

```
    1                                  .SBTTL  DETERMINE INITIALIZATION ENTRY
    2
    3                                  ;       TEST FOR OPERATOR "START"
    4 036122                           50$:
    5 036122  012700  000040                   MOV     #EF.STA,R0
      036126  104447                           TRAP    C$REFG
    6 036130  103412                            BCS     100$
    7
    8                                  ;       TEST FOR OPERATOR "RESTART"
    9 036132                           60$:
   10 036132  012700  000037                   MOV     #EF.RES,R0
      036136  104447                           TRAP    C$REFG
   11 036140  103455                            BCS     120$
   12
   13                                  ;       TEST FOR NEW PASS
   14 036142                           70$:
   15 036142  012700  000035                   MOV     #EF.NEW,R0
      036146  104447                           TRAP    C$REFG
   16 036150  103455                            BCS     124$
   17
   18 036152  000137  036646                   JMP     300$                    ; ELSE, EXIT
   19
```

```
        1                               .SBTTL  OPERATOR "START"
        2
        3                               ;OPERATOR "START" COMMAND
        4                               ;
        5                               ;       INITIALIZE FREE MEMORY (FFREE,FSIZE)
        6
        7 036156                        100$:
        8 036156  012746  004362                MOV     #INF100,-(SP)        ;PUSH #INF100 ON STACK
          036162  004137  033244                JSR     R1,LPNTF                    ;CALL LPNTF PRINT ROUTINE
          036166  000002                        .WORD   ARG.CT               ;ARGUMENT COUNT * 2
        9 036170  012701  022464                MOV     #BACKMSG,R1          ; LOAD QUESTION ADDRESS
       10 036174  012702  002652                MOV     #BACKUP,R2           ; LOAD REPLY ADDRESS
       11 036200  012712  000000                MOV     #0,(R2)              ; DEFAULT IS "NO"
       12 036204  004737  023126                JSR     PC,GETLOG            ; ASK OPERATOR
       13 036210  005737  002652                TST     BACKUP               ; TEST FOR BACKUP
       14 036214  001001                        BNE     105$                 ; BRANCH IF YES
       15
       16 036216  104444                        TRAP    C$DCLN
       17 036220                        105$:
       18 036220  104431                        TRAP    C$MEM
          036222  010037  002176                MOV     R0,FFREE
       19 036226  017737  143744  002200         MOV     @FFREE,FSIZE        ; RESET SIZE OF FREE MEMORY
       20
       21 036234  012700  000010                MOV     #8.,R0               ; LOAD COUNTER
       22 036240  012701  002230                MOV     #CTLRLST,R1          ; LOAD CONTROLLER TABLE LIST
       23 036244  012702  002254                MOV     #CACHLST,R2          ; LOAD CACHING TABLE LIST
       24 036250                        110$:
       25 036250  005021                        CLR     (R1)+                ; CLEAR CONTROLLER TABLE ENTRY
       26 036252  005022                        CLR     (R2)+                ; CLEAR CACHE TABLE ENTRY
       27 036254  005300                        DEC     R0                   ; DECREMENT
       28 036256  003374                        BGT     110$                 ; AND CONTINUE
       29
       30 036260  004737  040004                JSR     PC,HCOMM             ; ALLOCATE SPACE FOR HOST COMM AREA
       31 036264  012701  000010                MOV     #8.,R1               ; LOAD SIZE
       32 036270  004737  040054                JSR     PC,ALOCM             ; AND ALLOCATE
       33
```

```
    1                              .SBTTL   OPERATOR "RESTART"
    2                           ;
    3                           ;OPERATOR "RESTART"COMMAND
    4                           ;
    5
    6 036274                    120$:
    7 036274  104450                    TRAP     C$MANI
    8 036276  103002                    BCC      124$
    9 036300  004737  022206            JSR      PC,INISWP          ; INITIALIZE SW PARMS
   10 036304                    124$:
   11 036304  005737  002150            TST      MANRPL            ; TEST FOR MANUAL MODE
   12 036310  001403                    BEQ      125$              ; BRANCH IF NO
   13 036312  012737  000001  002012    MOV      #1,L$UNIT         ; ELSE, SET LAST UNIT TO ZERO
   14 036320                    125$:
   15 036320  012737  177777  002074    MOV      #-1,L$LUN         ; INITIALIZE LOGICAL UNIT NUMBER
   16 036326  012737  177776  002300    MOV      #-2,TBLNDX        ; INITIALIZE ADDRESS TABLE INDEX
   17 036334                    130$:
   18 036334  005237  002074            INC      L$LUN             ; INCREMENT TO NEXT LOGICAL UNIT
   19 036340  062737  000002  002300    ADD      #2,TBLNDX         ; INCREMENT ADDRESS TABLE INDEX
   20
   21 036346  023737  002074  002012    CMP      L$LUN,L$UNIT      ; TEST IF LOGICAL UNITS ARE EXHAUSTED
   22 036354  002134                    BGE      300$              ; BRANCH IF YES
   23                                                             ; ELSE, GET PTABLE POINTER
   24 036356  013700  002074            MOV      L$LUN,RO
      036362  104442                    TRAP     C$GPHRD
      036364  010037  002302            MOV      RO,PTABLE
   25 036370  103361                    BCC      130$
   26
```

```
  1
  2                                  ;       ALLOCATE CONTROLLER TABLE
  3 036372                    150$:
  4 036372  013702  002300           MOV     TBLNDX,R2              ; LOAD TABLE INDEX
  5 036376  016201  002230           MOV     CTLRLST(R2),R1        ; LOAD TABLE ADDRESS
  6 036402  001402                    BEQ     155$                  ; BRANCH IF NO ADDRESS
  7 036404  000137  036334            JMP     130$                  ; ELSE, NEXT UNIT
  8 036410                    155$:   .
  9 036410  012701  000016           MOV     #<C.SIZE+4>/2,R1      ; LOAD TABLE SIZE + 2
 10 036414  004737  040054           JSR.    PC,ALOCM              ; AND ALLOCATE
 11 036420  010162  002230           MOV     R1,CTLRLST(R2)        ; SAVE TABLE ADDRESS
 12 036424  013761  002324  000016   MOV     HCOM,C.HCOM(R1)       ; SAVE COMM AREA ADDRESS
 13 036432  012761  000000  000020   MOV     #0,C.DRVT(R1)         ; ZERO DRIVE TABLE PTR
 14
 15                                  ;       INITIALIZE CONTROLLER TABLE
 16 036440                    160$:
 17 036440  010137  002304           MOV     R1,CTLRTBL            ; SAVE CONTROLLER TABLE ADDRESS
 18 036444  013703  002302           MOV     PTABLE,R3            ; LOAD PTABLE ADDRESS
 19 036450  011321                    MOV     (R3),(R1)+           ; STORE XDA IP ADDRESS
 20 036452  016304  000004           MOV     H.BRL(R3),R4         ; GET THE BR LEVEL
 21 036456  000304                    SWAB    R4                   ; SWAP TO HIGH BYTE,
 22 036460  006104                    ROL     R4                   ; SHIFT ONE MORE TO LEFT,
 23 036462  056304  000002           BIS     H.VEC(R3),R4         ; ADD VECTOR ADDRESS AND
 24 036466  010421                    MOV     R4,(R1)+             ; STORE IT IN THE CONTROLLER TABLE.
 25 036470  016321  000006           MOV     H.BST(R3),(R1)+      ; STORE BURST RATE AND
 26 036474  016321  000010           MOV     H.DRV(R3),(R1)+      ; DRIVE NUMBER TO THE CONTROLLER TABLE.
 27 036500  012721  004037           MOV     #4037,(R1)+          ; STORE THE 'JSR R0' INSTRUCION AND
 28 036504  012721  016164           MOV     #XDASRV,(R1)+        ; STORE ISR ADDRESS
 29
 30 036510  013701  002304           MOV     CTLRTBL,R1           ; RELOAD TABLE START
 31 036514  005061  000014           CLR     C.FLG(R1)            ; CLEAR FLAGS
 32 036520  005061  000022           CLR     C.TO(R1)             ; CLEAR TIMEOUT (LOW)
 33 036524  005061  000024           CLR     C.TOH(R1)            ; CLEAR TIMEOUT (HIGH)
 34 036530  005061  000026           CLR     C.REF(R1)            ; CLEAR CMD REF
 35
```

```
 1                              ;       ALLOCATE DRIVE TABLE
 2 036534                       170$:
 3 036534  013705  002304               MOV     CTLRTBL,R5              ; LOAD CONTROLLER TABLE ADDRESS
 4 036540  016501  000020               MOV     C.DRVT(R5),R1           ; LOAD DRIVE TABLE ADDRESS
 5 036544  001006                       BNE     180$                   ; BRANCH IF PRESENT
 6
 7 036546  012701  000074               MOV     #<D.SIZE+4>/2,R1       ; LOAD DRIVE TABLE SIZE + 2
 8 036552  004737  040054               JSR     PC,ALOCM               ; ALLOCATE
 9 036556  010165  000020               MOV     R1,C.DRVT(R5)          ; AND STORE OIN CONTROLLER TABLE
10
11                              ;       INITIALIZE DRIVE TABLE
12 036562                       180$:
13 036562  012700  000072               MOV     #<D.SIZE>/2,R0        ; LOAD DRIVE TABLE SIZE
14 036566                       185$:
15 036566  005021                       CLR     (R1)+                  ; CLEAR LOCATION
16 036570  005300                       DEC     R0                     ; AND CONTINUE
17 036572  003375                       BGT     185$                   ; UNTIL TABLE END
18
```

```
     1
     2                              ;       ALLOCATE CACHING TABLE
     3 036574                       200$:
     4 036574  013702  002300       MOV     TBLNDX,R2               ; LOAD TABLE INDEX
     5 036600  016201  002254       MOV     CACHLST(R2),R1          ; LOAD TABLE ADDRESS
     6 036604  001006               BNE     220$                    ; BRANCH IF ADDRESS PRESENT
     7
     8 036606  012701  001442       MOV     #<CACHESIZE+1>*2,R1     ; LOAD TABLE SIZE + 2
     9 036612  004737  040054       JSR     PC,ALOCM                ; AND ALLOCATE
    10 036616  010162  002254       MOV     R1,CACHLST(R2)          ; SAVE TABLE ADDRESS
    11
    12                              ;       INITIALIZE CACHE TABLE
    13 036622                       220$:
    14 036622  010137  002310       MOV     R1,CACHTBL              ; SAVE CONTROLLER TABLE ADDRESS
    15 036626  012700  001440       MOV     #<CACHESIZE>*2,R0       ; LOAD TABLE SIZE
    16 036632                       225$:
    17 036632  012721  177777       MOV     #-1,(R1)+               ; INITIALIZE ENTRY
    18 036636  005300               DEC     R0                      ; DECREMENT
    19 036640  001374               BNE     225$                    ; CONTINUE UNTIL ZERO
    20
    21 036642  000137  036334       JMP     130$                    ; CONTINUE WITH NEXT UUT
    22
    23                              ;       EXIT INITIALIZATION
    24 036646                       300$:
    25 036646  005737  002652       TST     BACKUP                  ; TEST FLAG
    26 036652  001001               BNE     305$                    ; BRANCH IF SET
    27
    28 036654  104444               TRAP    C$DCLN
    29 036656                       305$:                           ; SET PRIORITY TO ZERO
    30 036656  012700  000000       MOV     #PRI00,R0
       036662  104441               TRAP    C$SPRI
    31
    32         000000               I$SECTION            = 0
    33
    34 036664  104432               TRAP    C$EXIT
       036666  001232               .WORD   L10021-.
    35
```

```
  1                                      .SBTTL  XDAINT  INITIALIZE XDA
  2                                      ;XDAINT
  3                                      ;
  4                                      ;FUNCTIONAL DESCRIPTION:
  5                                      ;      SUBROUTINE TO INITIALIZE A XDA AND BRING IT ON-LINE.
  6                                      ;      ALL STEPS ARE CHECKED. AN ERROR MESSAGE IS REPORTED IF ANY ERROR
  7                                      ;      DETECTED.
  8                                      ;
  9                                      ;INPUTS:
 10                                      ;      R5 - ADDRESS OF CONTROLLER TABLE.
 11                                      ;      R4 - LENGTH, INTERRUPT AND VECTOR FIELDS TO SEND TO XDA
 12                                      ;IMPLICIT INPUTS:
 13                                      ;      HCOM    - HOST COMMUNICATIONS AREA
 14                                      ;      FFREE - FIRST FREE ADDRESS OF MEMORY. THIS ADDRESS IS GIVEN TO XDA
 15                                      ;              AS START OF RING BUFFER.
 16                                      ;      FSIZE - SIZE OF FREE MEMORY AVAILABLE IN WORDS.
 17                                      ;OUTPUTS:
 18                                      ;      R1 - SIZE OF RING BUFFER IN WORDS IF NO ERROR,
 19                                      ;      R4 - ADDRESS OF XDAIP REGISTER IN XDA,
 20                                      ;      R5 - UNCHANGED,
 21                                      ;
 22                                      ;      Z CLR   IF NO ERROR,
 23                                      ;      Z SET   IF ANY ERROR REPORTED
 24                                      ;
 25                                      ;CHECK IF ENOUGH FREE MEMORY FOR RING BUFFER
 26
 27 036670                              XDAINT:
 28 036670  010346                          MOV     R3,-(SP)                ;;PUSH R3 ON STACK
 29 036672  010400                          MOV     R4,R0                   ;GET MESSAGE LENGTH
 30 036674  000300                          SWAB    R0
 31 036676  042700  177770                  BIC     #177770,R0
 32 036702  004737  034416                  JSR     PC,CLOG                 ;COMPUTE LOGARITHMIC VALUE
 33 036706  010102                          MOV     R1,R2                   ;SAVE RESULT IN R2
 34 036710  010400                          MOV     R4,R0                   ;GET COMMAND LENGTH
 35 036712  000300                          SWAB    R0
 36 036714  006000                          ROR     R0
 37 036716  006000                          ROR     R0
 38 036720  006000                          ROR     R0
 39 036722  042700  177770                  BIC     #177770,R0
 40 036726  004737  034416                  JSR     PC,CLOG                 ;COMPUTE LOGARITHMIC VALUE
 41 036732  060201                          ADD     R2,R1                   ;ADD THE TWO RESULTS
 42 036734  006301                          ASL     R1                      ;MULTIPLY BY 2 WORDS PER RING
 43 036736  062701  000002                  ADD     #<HC.ISZ>/2,R1          ;ADD SPACE FOR INTERRUPT INDICATORS
 44 036742  020137  002200                  CMP     R1,FSIZE                ;COMPARE WITH SIZE OF FREE MEMORY
 45 036746  101402                          BLOS    1$
 46 036750  000137  040100                  JMP     FMERR                   ;FATAL ERROR IF NOT ENOUGH MEMORY
 47
 48                                      ;FILL HOST COMMUNICATION AREA WITH ALL ONES
 49
 50 036754  013702  002324      1$:         MOV     HCOM,R2                 ; LOAD COMM AREA ADDRESS
 51 036760  010103                          MOV     R1,R3                   ;GET SIZE OF RING BUFFER
 52 036762  012722  177777      2$:         MOV     #-1,(R2)+               ;WRITE ONES TO BUFFER
 53 036766  005303                          DEC     R3                      ;COUNT THE WORDS IN BUFFER
 54 036770  003374                          BGT     2$                      ;LOOP UNTIL ENTIRE BUFFER WRITTEN
 55
 56                                      ;DO THE INITIALIZATION
 57
```

```
 58 036772  004737  037200            JSR     PC,XDAIST              ;DO FIRST THREE STEPS
 59 036776  103475                    BCS     9$                    ;GET OUT IF MICROCODE REPORTED FAILURE
 60 037000  012364  000002            MOV     (R3)+,2(R4)           ;WRITE NEXT WORD TO XDASA REGISTER
 61 037004  012700  000310            MOV     #200.,R0              ;GET TRY COUNTER
 62 037010                     3$:
 63 037010  016402  000002            MOV     2(R4),R2              ;LOOK AT XDASA
 64 037014  001420                    BEQ     5$
 65 037016  100015                    BPL     4$
 66
 67 037020  010237  002316            MOV     R2,SAVAL              ; SAVE CONTENTS
 68 037024  012737  011061  002552    MOV     #ERM102,BASEMSG       ; "SA ERROR DURING STEP 3"
 69 037032  012737  000003  002370    MOV     #B.SADDR!B.SAVAL,EXTMSK ; FLAG ADDRESS AND VALUE
 70 037040  104455                    TRAP    C$ERDF
    037042  000146                    .WORD   102
    037044  010667                    .WORD   HDR101
    037046  015266                    .WORD   ERR101
 71 037050  000450                    BR      9$
 72
 73 037052  005300             4$:    DEC     R0
 74 037054  001355                    BNE     3$
 75 037056  010264  000002     5$:    MOV     R2,2(R4)              ;WRITE 0 TO XDASA (PURGE)
 76 037062  011402                    MOV     (R4),R2               ;READ FROM XDAIP (POLL)
 77 037064  004737  037606            JSR     PC,XDARSP             ;WAIT FOR STEP OR ERROR BIT
 78 037070  103440                    BCS     9$                    ;GET OUT IF MICROCODE REPORTED FAILURE
 79 037072  010146                    MOV     R1,-(SP)              ;;PUSH R1 ON STACK
 80 037074  004733                    JSR     PC,@(R3)+             ;CALL LAST ROUTINE
 81 037076  012601                    MOV     (SP)+,R1              ;;POP STACK INTO R1
 82
 83                                   ;CHECK HOST COMMUNICATION AREA FOR ALL ZEROS
 84
 85 037100  013702  002324            MOV     HCOM,R2               ; LOAD COMM AREA ADDRESS
 86 037104  010103                    MOV     R1,R3                 ;GET SIZE OF RING BUFFER
 87 037106  005722             6$:    TST     (R2)+                 ;CHECK WORD IN BUFFER
 88 037110  001003                    BNE     7$                    ;GO TO ERROR REPORTER IF NOT ZERO
 89 037112  005303                    DEC     R3                    ;COUNT THE WORDS IN BUFFER
 90 037114  003374                    BGT     6$                    ;LOOP UNTIL ALL WORDS CHECKED
 91 037116  000412                    BR      8$
 92 037120                     7$:
 93 037120  016402  000002            MOV     2(R4),R2
 94 037124  012737  011145  002552    MOV     #ERM103,BASEMSG       ; "DID NOT CLEAR MEMORY"
 95 037132  104455                    TRAP    C$ERDF
    037134  000147                    .WORD   103
    037136  010667                    .WORD   HDR101
    037140  015266                    .WORD   ERR101
 96 037142  000413                    BR      9$
 97
 98                                   ;SEND GO BIT TO XDASA REGISTER TO END INITIALIZATION
 99
100 037144  016500  000004     8$:    MOV     C.BST(R5),R0          ;GET BURST VALUE
101 037150  006300                    ASL     R0                    ;SHIFT TO POSITION
102 037152  006300                    ASL     R0
103 037154  052700  000001            BIS     #SA.GO,R0             ;SET THE GO BIT
104 037160  010064  000002            MOV     R0,2(R4)              ;SEND TO XDA
105 037164  012603                    MOV     (SP)+,R3              ;;POP STACK INTO R3
106 037166  000244                    CLZ                           ;CLEAR Z AS NO ERROR INDICATION
107 037170  000207                    RTS     PC
108
```

```
109                                    ;ERROR RETURN
110
111 037172                   9$:
    037172  012603            MOV     (SP)+,R3           ;;POP STACK INTO R3
112 037174  000264            SEZ                        ;SET Z TO INDICATE ERROR OCCURRED
113 037176  000207            RTS     PC
```

```
  1                                    .SBTTL  XDAIST  START XDA INITIALIZE SEQUENCE
  2                                    ;XDAIST
  3                                    ;
  4                                    ;START THE INITIALIZATION PROCESS ON THE SELECTED XDA.
  5                                    ;STOP BEFORE WRITING THE THIRD WORD SO XDA 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 XDA
 11                                    ;
 12                                    ;LOAD TABLE OF DATA TO SEND TO XDASA REGISTER
 13
 14 037200                    XDAIST:                            ;>>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<
 15 037200  104422                    TRAP    C$BRK
 16 037202  010146                    MOV     R1,-(SP)           ;;PUSH R1 ON STACK
 17 037204  052704  100000            BIS     #SA.STP,R4         ;SET STEP BIT IN DATA WORD
 18 037210  010437  037424            MOV     R4,SND.S1          ;LOAD SEND DATA FOR STEP 1 OF XDA INIT
 19 037214  013737  002324  037430    MOV     HCOM,SND.S2        ; LOAD COMM AREA ADDRESS
 20 037222  062737  000004  037430    ADD     #HC.MSG,SND.S2     ;LOAD SEND DATA FOR STEP 2 OF XDA INIT
 21 037230  012737  100000  037434    MOV     #SA.TST,SND.S3     ;LOAD SEND DATA FOR STEP 3 OF XDA INIT
 22
 23                                    ;START THE INITIALIZATION BY WRITING TO XDAIP REGISTER
 24
 25 037236  016504  000000            MOV     C.UADR(R5),R4      ;GET ADDRESS OF XDAIP REGISTER
 26 037242  005037  002224            CLR     NXMAD              ;CLEAR MEMORY ERROR FLAG
 27 037246  012746  000340            MOV     #PRIO7,-(SP)
    037252  012746  016154            MOV     #NXMI,-(SP)
    037256  012746  000004            MOV     #4,-(SP)
    037262  012746  000003            MOV     #3,-(SP)
    037266  104437                    TRAP    C$SVEC
    037270  062706  000010            ADD     #10,SP
 28 037274  005764  000002            TST     2(R4)              ;ACCESS XDASA REGISTER
 29 037300  005014                    CLR     (R4)               ;WRITE TO XDAIP
 30 037302  012700  000004            MOV     #4,R0
    037306  104436                    TRAP    C$CVEC
 31 037310  005737  002224            TST     NXMAD              ;SEE IF A MEMORY ERROR OCCURRED
 32 037314  001414                    BEQ     1$
 33
 34 037316  012737  010727  002552    MOV     #ERM101,BASEMSG    ; "MEMORY ERROR"
 35 037324  012737  000001  002370    MOV     #B.SADDR,EXTMSK    ; FLAG ADDRESS
 36 037332  104455                    TRAP    C$ERDF
    037334  000145                    .WORD   101
    037336  010667                    .WORD   HDR101
    037340  015266                    .WORD   ERR101
 37 037342  000261                    SEC
 38 037344  000424                    BR      4$
 39
 40                                    ;SET UP LOOP PARAMETERS TO EXECUTE THE FOUR STEPS OF INITIALIZATION
 41
 42 037346  012737  004000  040002 1$: MOV    #SA.S1,XDARSD      ;STORE RESPONSE MASK
 43 037354  012703  037422            MOV     #INITBL,R3         ;GET INDEX TO XDA
 44                                                              ; SEND/REPOND INITIALIZE TABLE
 45
 46                                    ;WAIT FOR AND CHECK RESPONSE DATA
 47
 48 037360  004737  037606        2$: JSR     PC,XDARSP          ;WAIT FOR STEP OR ERROR BITS
```

```
 49 037364  103414                        BCS      4$                    ;EXIT IF ERROR
 50 037366  004733                        JSR      PC,@(R3)+             ;CALL RESPONSE CHECKER FOR STEP
 51 037370  103412                        BCS      4$                    ;GET OUT IF ERROR
 52 037372  006337  040002                ASL      XDARSD                ;SHIFT TO NEXT STEP BIT
 53 037376  032737  040000  040002        BIT      #SA.S4,XDARSD         ;CHECK IF NOW AT STEP 4
 54 037404  001003                        BNE      3$                    ;GET OUT IF SO
 55 037406  012364  000002                MOV      (R3)+,2(R4)           ;WRITE DATA TO XDASA REGISTER
 56 037412  000762                        BR       2$                    ;STAY IN LOOP
 57
 58 037414  000241           3$:          CLC                            ;CLEAR CARRY FOR NO ERROR INDICATION
 59 037416                    4$:
    037416  012601                         MOV      (SP)+,R1             ;;POP STACK INTO R1
 60 037420  000207                         RTS      PC
 61
 62                           ;DATA TO BE SENT AND RECEIVED BY XDA INITIALIZATION
 63
 64 037422  037440           INITBL: .WORD  RSP.S1                       ;1ST WORD RESPONSE CHECK ROUTINE
 65 037424  000000           SND.S1: .WORD  0                            ;1ST WORD TO SEND TO XDASA
 66 037426  037452                   .WORD  RSP.S2                       ;2ND WORD RESPONSE CHECK ROUTINE
 67 037430  000000           SND.S2: .WORD  0                            ;2ND WORD TO SEND TO XDASA
 68 037432  037472                   .WORD  RSP.S3                       ;3RD WORD RESPONSE CHECK ROUTINE
 69 037434  000000           SND.S3: .WORD  0                            ;3RD WORD TO SEND TO XDASA
 70 037436  037510                   .WORD  RSP.S4                       ;4TH WORD RESPONSE CHECK ROUTINE
 71
 72                           ;      RESPONSE CHECK FOR FIRST WORD (STEP 1) FROM XDASA
 73                           ;      CHECK FOR PROPER CONTROLLER TYPE
 74 037440                   RSP.S1:
 75 037440  012701  004400            MOV      #SA.S1+SA.DI,R1           ;SET STEP 1 & ENHANCED DIAGNOSTIC BITS
 76 037444  042702  001140            BIC      #SA.QB+SA.MP+40,R2        ;IGNORE 22 BIT ADRS & MAPPING BITS
 77 037450                   10$:
 78 037450  000434                     BR       RSP.CK                   ;NOW DO A RESPONSE CHECK
 79
 80                           ;      RESPONSE CHECK FOR SECOND WORD (STEP 2) FROM XDASA
 81                           ;      CHECK FOR ECHO OF INTI AND VECTOR
 82 037452                   RSP.S2:
 83 037452  013701  037424            MOV      SND.S1,R1                ;GET WORD SENT TO XDASA
 84 037456  000301                     SWAB     R1                       ;GET HIGH 8 BITS
 85 037460  042701  177400            BIC      #177400,R1
 86 037464  052701  010000            BIS      #SA.S2,R1                ;SET STEP 2 BIT
 87 037470  000424                     BR       RSP.CK                   ;NOW DO A RESPONSE CHECK
 88
 89                           ;      RESPONSE CHECK FOR THIRD WORD (STEP 3) FROM XDASA
 90                           ;      CHECK FOR ECHO OF MESSAGE AND COMMAND RING LENGTHS
 91 037472                   RSP.S3:
 92 037472  013701  037424            MOV      SND.S1,R1                ;GET WORD SENT TO XDASA
 93 037476  042701  177400            BIC      #177400,R1               ;JUST LOW 8 BITS
 94 037502  052701  020000            BIS      #SA.S3,R1                ;SET STEP 3 BIT
 95 037506  000415                     BR       RSP.CK                   ;NOW DO A RESPONSE CHECK
 96
 97                           ;      RESPONSE CHECK FOR FOURTH WORD (STEP 4) FROM XDASA
 98                           ;      CHECK FOR ECHO OF PURGE AND LFAIL BITS
 99 037510                   RSP.S4:
100 037510  010201                     MOV      R2,R1                    ;GET RESPONSE FROM XDA
101 037512  042701  137400            BIC      #^C<SA.S4+SA.CNT+SA.MCV>,R1 ;KEEP MICROCODE VERSION AND STEP 4
102 037516  032701  000360            BIT      #SA.CNT,R1               ;CHECK WITH CONTROLLER MODEL
103 037522  001404                     BEQ      1$                       ;IF ZERO, XDA50(M7161)
104 037524  042765  000040  000014    BIC      #CT.U50,C.FLG(R5)        ;ELSE, XDA52(M7485)
```

```
105 037532  000403                            BR      2$                        ;AND BRANCH
106 037534                          1$:
107 037534  052765  000040  000014  BIS     #CT.U50,C.FLG(R5)
108 037542                          2$:
109
110                                 ;       RESPONSE CHECK, COMPARE EXPECTED DATA IN R1 WITH ACTUAL DATA IN R2
111 037542                          RSP.CK:
112 037542  020102                  CMP     R1,R2                     ;COMPARE THE DATA
113 037544  001417                  BEQ     1$                        ;EXIT IF COMPARED CORRECTLY
114
115 037546  010137  002320          MOV     R1,SAEXP                  ; SAVE EXPECTED VALUE
116 037552  010237  002316          MOV     R2,SAVAL                  ; SAVE ACTUAL CONTENTS
117 037556  012737  011061  002552  MOV     #ERM102,BASEMSG           ; "SA REGISTER ERROR"
118 037564  012737  000007  002370  MOV     #B.SADDR!B.SAVAL!B.SAEXP,EXTMSK; FLAG ALL FIELDS
119 037572  104455                  TRAP    C$ERDF
    037574  000146                  .WORD   102
    037576  010667                  .WORD   HDR101
    037600  015266                  .WORD   ERR101
120 037602  000261                  SEC
121 037604                          1$:
122 037604  000207                  RTS     PC
```

```
     1                                .SBTTL  XDARSP  WAIT FOR XDA INITIALIZE RESPONSE
     2                                ;XDARSP
     3                                ;
     4                                ;WAIT FOR XDA TO RESPOND WITH DATA IN XDASA REGISTER.
     5                                ;EITHER STEP BIT FROM MASK IN LOCATION XDARSD OR ERROR BIT
     6                                ;WILL CAUSE A TERMINATION.
     7                                ;AN ERROR MESSAGE WILL BE PRINTED IF THE XDA DOES NOT RESPOND
     8                                ;IN 10 SECONDS OR IF ERROR SETS.
     9                                ;
    10                                ;INPUTS:
    11                                ;        XDASRD - MASK OF STEP BIT TO LOOK FOR
    12                                ;        R5 - ADDRESS OF CONTROLLER TABLE
    13                                ;        R4 - ADDRESS OF XDAIP REGISTER
    14                                ;OUTPUTS:
    15                                ;        ERROR MESSAGE IF TIME OUT ON RESPONSE OR ERROR BIT SETS
    16                                ;        R2 - DATA FROM XDASA REGISTER
    17                                ;        CARRY SET IF ERROR BIT SETS OR TIME OUT
    18
    19 037606                        XDARSP:
       037606  010146                        MOV     R1,-(SP)                ;;PUSH R1 ON STACK
    20 037610  013737  040002  002320        MOV     XDARSD,SAEXP            ; SAVE EXPECTED VALUE
    21 037616  052737  100000  040002        BIS     #SA.ERR,XDARSD         ;SET ERROR BIT IN MASK WORD
    22 037624  012700  000012                MOV     #10.,R0                 ;SET UP FOR 10 SECOND TIMEOUT
    23 037630  010501                        MOV     R5,R1                   ;POINT TO COUNTER IN CONTROLLER TABLE
    24 037632  062701  000022                ADD     #C.TO,R1
    25 037636  004737  034334                JSR     PC,SETTO
    26 037642  012601                        MOV     (SP)+,R1                ;;POP STACK INTO R1
    27 037644                        1$:
    28 037644  016402  000002                MOV     2(R4),R2                ; READ SA REGISTER
    29 037650  010237  002316                MOV     R2,SAVAL                ; AND SAVE CONTENTS
    30 037654  033702  040002                BIT     XDARSD,R2               ; LOOK AT ERROR AND STEP BIT
    31 037660  001030                        BNE     3$                      ;BRANCH IF EITHER SET
    32                                                                       ;>>>>>>>>BREAK BACK TO MONITOR<<<<<<<<<
    33 037662  104422                        TRAP    C$BRK
    34 037664  005737  002210                TST     KW.CSR                  ;SEE IF CLOCK ON SYSTEM
    35 037670  001765                        BEQ     1$
    36 037672  023765  002222  000024        CMP     KW.EL+2,C.TOH(R5)      ;CHECK IF TIME OUT OCCURRED
    37 037700  101005                        BHI     2$
    38 037702  001360                        BNE     1$
    39 037704  023765  002220  000022        CMP     KW.EL,C.TO(R5)
    40 037712  103754                        BLO     1$
    41 037714                        2$:                                     ; REPORT FATAL ERROR
    42 037714  012737  011061  002552        MOV     #ERM102,BASEMSG        ; "STEP BIT ERROR"
    43 037722  012737  000007  002370        MOV     #B.SADDR!B.SAVAL!B.SAEXP,EXTMSK; FLAG ALL FIELDS
    44 037730  104455                        TRAP    C$ERDF
       037732  000146                        .WORD   102
       037734  010667                        .WORD   HDR101
       037736  015266                        .WORD   ERR101
    45 037740  000414                        BR      4$
    46
    47                                ;       CHECK IF ERROR BIT SET
    48 037742                        3$:
    49 037742  005702                        TST     R2                      ; READ TEST CONTENTS
    50 037744  100014                        BPL     5$                      ; EXIT IF ERROR NOT SET
    51
    52 037746  012737  011061  002552        MOV     #ERM102,BASEMSG        ; "RESIDENT DIAG FAILURE"
    53 037754  012737  000003  002370        MOV     #B.SADDR!B.SAVAL,EXTMSK ; FLAG ADDRESS AND VALUE
```

```
54 037762  104455                    TRAP    C$ERDF
   037764  000146                    .WORD   102
   037766  010667                    .WORD   HDR101
   037770  015266                    .WORD   ERR101
55 037772                  4$:
56 037772  000261                    SEC
57 037774  000207                    RTS     PC
58
59                                   ;NORMAL EXIT
60
61 037776  000241        5$:         CLC             ;CLEAR CARRY AS NO ERROR INDICATION
62 040000  000207                    RTS     PC
63
64                                   ;LOCATION FOR STEP BIT MASK
65
66 040002  000000        XDARSD: .WORD   0            ;LOAD BY CALLING ROUTINE
67
```

```
 1                               .SBTTL  HCOMM    INITIALIZE HOST COMM AREA
 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 040004  012701  000074  HCOMM:  MOV    #<HC.SIZ+4>/2,R1      ; LOAD COMM AREA SIZE + 2
16 040010  004737  040054          JSR    PC,ALOCM             ; ALLOCATE
17 040014  010104                  MOV    R1,R4                ; SAVE ADDRESS
18 040016  010437  002324          MOV    R4,HCOM              ; LOCALLY
19 040022  062701  000020          ADD    #HC.MPK,R1           ; COMPUTE START OF MESSAGE PACKET
20 040026  010164  000004          MOV    R1,HC.MSG(R4)        ; PLACE IN RING
21 040032  010137  002330          MOV    R1,MSGPCK            ; AND SAVE LOCALLY
22 040036  062701  000064          ADD    #<HC.CPK-HC.MPK>,R1  ; COMPUTE START OF COMMAND PACKET
23 040042  010164  000010          MOV    R1,HC.CMD(R4)        ; PLACE IN RING
24 040046  010137  002326          MOV    R1,CMDPCK            ; AND SAVE LOCALLY
25 040052  000207                  RTS    PC
26
```

```
                                    .SBTTL   MEMORY ALLOCATION ROUTINES
    1                               ;ALOCM
    2                               ;
    3                               ;ALLOCATE A BLOCK OF FREE MEMORY.  REPORT ERROR IF MEMORY EXHAUSTED.
    4                               ;
    5                               ;INPUTS:
    6                               ;      R1 - NUMBER OF WORDS TO ALLOCATE
    7                               ;      FFREE - FIRST FREE WORD IN MEMORY
    8                               ;      FSIZE - SIZE OF FREE MEMORY AVAILABLE IN WORDS
    9                               ;OUTPUTS:
   10                               ;      R1 - ADDRESS OF FIRST WORD OF ALLOCATED MEMORY
   11                               ;      FFREE - NEW FIRST FREE WORD IN MEMORY
   12                               ;      FSIZE - SIZE OF FREE MEMORY LEFT AFTER ALLOCATION
   13                               ;SYSTEM FATAL ERROR WILL BE REPORTED IF NOT ENOUGH MEMORY AVAILABLE
   14                               ;AND ENTIRE PROGRAM WILL BE STOPPED.
   15
   16
   17 040054                        ALOCM:
   18 040054   013746  002176          MOV     FFREE,-(SP)            ;;PUSH FFREE ON STACK
   19 040060   160137  002200          SUB     R1,FSIZE               ;REDUCE SIZE OF FREE MEMORY
   20 040064   002405                  BLT     FMERR                  ;REPORT ERROR IF NOT ENOUGH MEMORY
   21 040066   060101                  ADD     R1,R1                  ;CHANGE WORDS TO BYTES
   22 040070   060137  002176          ADD     R1,FFREE               ;CALCULATE NEW START OF FREE MEMORY
   23 040074   012601                  MOV     (SP)+,R1               ;;POP STACK INTO R1
   24 040076   000207                  RTS     PC
   25
   26                               ;FMERR
   27                               ;
   28                               ;MEMORY ALLOCATION ERROR
   29                               ;
   30                               ;THIS ROUTINE PRINTS A SYSTEM FATAL ERROR AND EXITS THE TEST
   31
   32 040100                        FMERR:
   33 040100   012737  010527  002552   MOV    #ERM001,BASEMSG        ; "MEMORY ALLOCATION ERROR"
   34 040106   104454                  TRAP    C$ERSF
      040110   000001                  .WORD   1
      040112   010472                  .WORD   HDR001
      040114   015220                  .WORD   ERR001
   35 040116   104444                  TRAP    C$DCLN
   36
   37                                  .EVEN
   38
   39 040120                        L10021:
      040120   104411                  TRAP    C$INIT
```

```
         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 040122                         L$AUTO::
        11
        12 040122                         L10022:
           040122  104461                         TRAP    C$AUTO
```

```
    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 040124              L$CLEAN::
    9
   10 040124  112700  000015         MOVB    #CR,R0              ;STORE #CR IN R0 AND
      040130  004737  033066         JSR     PC,PRINTC           ;PRINT THE CHARACTER.
   11 040134  004737  016224         JSR     PC,RESET            ;RESET ALL UNITS
   12
   13 040140  104432                 TRAP    C$EXIT
      040142  000002                 .WORD   L10023-.
   14
   15                                 .EVEN
   16
   17 040144              L10023:
      040144  104412                 TRAP    C$CLEAN
   18
```

```
  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 040146                      L$DU::
  9
 10 040146  005737  002442              TST     STATUS                  ; TEST EXIT STATUS
 11 040152  100410                      BMI     20$                     ; BRANCH IF FATAL
 12 040154                      10$:                                    ; "TEST COMPLETE."
 13 040154  013746  002074              MOV     L$LUN,-(SP)             ;PUSH L$LUN ON STACK
    040160  012746  040216              MOV     #CMPUUT,-(SP)           ;PUSH #CMPUUT ON STACK
    040164  004137  033244              JSR     R1,LPNTF                        ;CALL LPNTF PRINT ROUTINE
    040170  000004                      .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
 14 040172  000407                      BR      30$
 15 040174                      20$:                                    ; "DROPPED."
 16 040174  013746  002074              MOV     L$LUN,-(SP)             ;PUSH L$LUN ON STACK
    040200  012746  040254              MOV     #DRPUUT,-(SP)           ;PUSH #DRPUUT ON STACK
    040204  004137  033244              JSR     R1,LPNTF                        ;CALL LPNTF PRINT ROUTINE
    040210  000004                      .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
 17 040212                      30$:
 18 040212  000167                      .WORD   J$JMP
    040214  000066                      .WORD   L10024-2-.
 19
 20                                     .NLIST  BEX
 21 040216    116    062    042 CMPUUT: .ASCIZ  /N2"UNIT "D8" TEST COMPLETE."N/
 22 040254    116    062    042 DRPUUT: .ASCIZ  /N2"UNIT "D8" DROPPED."N/
 23                                     .LIST   BEX
 24                                     .EVEN
 25
 26 040304                      L10024:
    040304  104453                      TRAP    C$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 040306                     L$AU::
    10
    11 040306  000167                      .WORD    J$JMP
       040310  000000                      .WORD    L10025-2-.
    12                                      .EVEN
    13
    14
    15 040312                     L10025:
       040312  104452                       TRAP     C$AU
    18
    19
```

```
  1                                     .SBTTL   TEST1:   MANUAL BAD BLOCK REPLACEMENT
  2
  3 040314                      T1::
  4
  5 040314  005737  002150              TST      MANRPL               ; TEST FOR MANUAL REPLACEMENT
  6 040320  001002                      BNE      10$                  ; BRANCH IF YES
  7 040322  104432                      TRAP     C$EXIT
    040324  000276                      .WORD    L10026-.
  8 040326                      10$:
  9 040326  104450                      TRAP     C$MANI
 10 040330  103402                      BCS      15$
 11 040332  104432                      TRAP     C$EXIT
    040334  000266                      .WORD    L10026-.
 12 040336                      15$:
 13 040336  012746  007061              MOV      #INF310,-(SP)        ;PUSH #INF310 ON STACK
    040342  004137  033244              JSR      R1,LPNTF                 ;CALL LPNTF PRINT ROUTINE
    040346  000002                      .WORD    ARG.CT               ;ARGUMENT COUNT * 2
 14
 15 040350  012737  000000  002074      MOV      #0,L$LUN             ; ZERO LUN
 16 040356  013737  002230  002304      MOV      CTLRLST,CTLRTBL      ; LOAD TABLE ADDRESS
 17 040364                      20$:
 18 040364  004737  021050              JSR      PC,UNITINI           ; INITIALIZE UNIT UNDER TEST
 19 040370  040602                      1200$                        ; ERROR EXIT
 20 040372                      50$:
 21 040372  012701  022707              MOV      #SWMSG4,R1           ; LOAD QUESTION ADDRESS
 22 040376  012702  002654              MOV      #REPLY,R2            ; LOAD ANSWER ADDRESS
 23 040402  012712  000000              MOV      #0,(R2)              ; DEFAULT IS "NO"
 24 040406  004737  023126              JSR      PC,GETLOG            ; ASK OPERATOR
 25 040412  105737  002654              TSTB     REPLY                ; TEST REPLY
 26 040416  001403                      BEQ      100$                 ; BRANCH IF "NO"
 27
 28 040420  004737  032364              JSR      PC,DSPDSC            ; ELSE, DISPLAY DESCRIPTORS
 29 040424  040602                      1200$                        ; ERROR EXIT
 30
 31                             ;        MANUAL BAD BLOCK REPLACEMENT.
 32                             ;        FETCH LBN FROM OPERATOR; CHECK LIMITS.
 33 040426                      100$:
 34 040426  012701  014705              MOV      #TXT000,R1           ; LOAD MESSAGE ADDRESS
 35 040432  012704  002446              MOV      #LBN,R4              ; LOAD LBN ADDRESS
 36 040436  004737  023410              JSR      PC,GETLBN            ; GET LBN FROM OPERATOR
 37 040442  001425                      BEQ      200$                 ; EXIT IF TTYIN = 0
 38 040444                      110$:
 39 040444  004737  023604              JSR      PC,READLBN           ; READ SUSPECT LBN
 40 040450  040602                      1200$                        ; ERROR EXIT
 41 040452  004737  017562              JSR      PC,CACHTST           ; UPDATE CACHING TABLE
 42 040456  040602                      1200$                        ; ERROR EXIT
 43
 44                             ;        REPLACE LBN IF REPLACEMENT ENABLED.
 45 040460                      130$:
 46 040460  005737  002160              TST      ENBRPL               ; TEST FOR REPLACEMENT ENABLED
 47 040464  001004                      BNE      150$                 ; BRANCH IF YES
 48 040466  004737  020002              JSR      PC,TYPSTAT           ; TYPE REPLACEMENT INFO
 49 040472  000137  040426              JMP      100$
 50 040476                      150$:
 51 040476  012737  000006  002546      MOV      #6,NXTSTEP           ; ENTRY IS STEP6
 52 040504  004737  024156              JSR      PC,REPLACE           ; REPLACE LBN
 53 040510  040602                      1200$                        ; ERROR RETURN
```

```
 54 040512  000137  040426            JMP     100$                    ; CONTINUE
 55 040516                    200$:
 56 040516  012746  007135            MOV     #INF311,-(SP)           ;PUSH #INF311 ON STACK
    040522  004137  033244            JSR     R1,LPNTF                      ;CALL LPNTF PRINT ROUTINE
    040526  000002                    .WORD   ARG.CT                  ;ARGUMENT COUNT * 2
 57 040530  104424                    TRAP    C$DRPT
 58
 59 040532  012701  022707            MOV     #SWMSG4,R1              ; LOAD QUESTION ADDRESS
 60 040536  012702  002654            MOV     #REPLY,R2              ; LOAD ANSWER ADDRESS
 61 040542  012712  000000            MOV     #0,(R2)                ; DEFAULT IS "NO"
 62 040546  004737  023126            JSR     PC,GETLOG              ; ASK OPERATOR
 63 040552  001403                    BEQ     400$                  ; BRANCH IF NO
 64
 65 040554  004737  032364            JSR     PC,DSPDSC             ; ELSE, DISPLAY DESCRIPTORS
 66 040560  040602                    1200$                        ; ERROR EXIT
 67
 68                           ;       NORMAL EXIT
 69 040562                    400$:
 70 040562  012737  000000  002442    MOV     #0,STATUS            ; SET SUCCESS
 71 040570  013700  002074            MOV     L$LUN,R0
    040574  104451                    TRAP    C$DODU
 72 040576  104432                    TRAP    C$EXIT
    040600  000022                    .WORD   L10026-.
 73
 74                           ;       ERROR EXIT
 75 040602                    1200$:
 76 040602  012737  177777  002442    MOV     #-1,STATUS           ; SET FATAL STATUS
 77 040610  013700  002074            MOV     L$LUN,R0
    040614  104451                    TRAP    C$DODU
 78 040616  104432                    TRAP    C$EXIT
    040620  000002                    .WORD   L10026-.
 79
 80                                   .EVEN
 81
 82 040622                    L10026:
    040622  104401                    TRAP    C$ETST
 83
```

```
    1                                      .SBTTL   TEST2:   AUTOMATIC BAD BLOCK REPLACEMENT
    2
    3 040624                      T2::
    4
    5 040624  005737  002150              TST      MANRPL              ; TEST FOR MANUAL REPLACEMENT
    6 040630  001402                      BEQ      10$                 ; BRANCH IF NO
    7 040632  104432                      TRAP     C$EXIT
      040634  001306                      .WORD    L10027-.
    8 040636                      10$:
    9 040636  012746  007164              MOV      #INF320,-(SP)       ;PUSH #INF320 ON STACK
      040642  004137  033244              JSR      R1,LPNTF                    ;CALL LPNTF PRINT ROUTINE
      040646  000002                      .WORD    ARG.CT              ;ARGUMENT COUNT * 2
   10 040650  012746  007242              MOV      #INF321,-(SP)       ;PUSH #INF321 ON STACK
      040654  004137  033254              JSR      R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
      040660  000002                      .WORD    ARG.CT              ;ARGUMENT COUNT * 2
   11 040662  012746  007630              MOV      #INF322,-(SP)       ;PUSH #INF322 ON STACK
      040666  004137  033254              JSR      R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
      040672  000002                      .WORD    ARG.CT              ;ARGUMENT COUNT * 2
   12
   13 040674  012737  177777  002074      MOV      #-1,L$LUN           ; INITIALIZE LUN
   14 040702  012737  177776  002300      MOV      #-2,TBLNDX          ; INITIALIZE TABLE INDEX
   15 040710                      100$:
   16 040710  005237  002074              INC      L$LUN               ; INCREMENT LUN
   17 040714  023737  002074  002012      CMP      L$LUN,L$UNIT        ; TEST IF ALL UNITS DONE
   18 040722  002402                      BLT      105$                ; BRANCH IF NO
   19 040724  000137  042136              JMP      1200$               ; ELSE, FINISHED
   20 040730                      105$:
   21 040730  062737  000002  002300      ADD      #2,TBLNDX           ; INCREMENT TABLE INDEX
   22 040736  013702  002300              MOV      TBLNDX,R2           ; LOAD INDEX
   23 040742  016237  002230  002304      MOV      CTLRLST(R2),CTLRTBL ; LOAD ADDRESS
   24 040750  001757                      BEQ      100$                ; BRANCH IF NONE
   25 040752                      120$:
   26 040752  004737  021050              JSR      PC,UNITINI          ; INITIALIZE UNIT UNDER TEST
   27 040756  042104                      1000$                       ; ERROR EXIT
   28
   29                              ;        AUTOMATIC BAD BLOCK REPLACEMENT.
   30                              ;        INITIALIZE TEST VARIABLES
   31 040760                      300$:                               ; HEADER
   32 040760  012746  007027              MOV      #INF302,-(SP)       ;PUSH #INF302 ON STACK
      040764  004137  033254              JSR      R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
      040770  000002                      .WORD    ARG.CT              ;ARGUMENT COUNT * 2
   33
   34 040772  104450                      TRAP     C$MANI
   35 040774  103033                      BCC      305$
   36
   37 040776  005737  002650              TST      DEBUG               ; TEST DEBUG FLAG
   38 041002  001430                      BEQ      305$                ; BRANCH IF NOT SET
   39
   40                              ;        GET MINIMUM AND MAXIMUM LBN'S FOR DEBUG
   41 041004                      301$:
   42 041004  012737  000000  002600      MOV      #0,MINLBN
   43 041012  012737  000000  002602      MOV      #0,MINLBN+2
   44 041020  012701  015060              MOV      #TX.LB1,R1
   45 041024  012704  002600              MOV      #MINLBN,R4
   46 041030  004737  023410              JSR      PC,GETLBN
   47 041034                      302$:
   48 041034  013737  002430  002604      MOV      UNSIZ,MAXLBN
```

```
49 041042  013737  002432  002606        MOV     UNSIZ+2,MAXLBN+2
50 041050  012701  015100                MOV     #TX.LB2,R1
51 041054  012704  002604                MOV     #MAXLBN,R4
52 041060  004737  023410                JSR     PC,GETLBN
53
54                                  ;     INITIALIZE STARTING GLBN
55 041064                        305$:
56 041064  013737  002600  002562        MOV     MINLBN,GLBN          ; LOAD INITIAL LBN (LOW)
57 041072  013737  002602  002564        MOV     MINLBN+2,GLBN+2      ; LOAD INITIAL LBN (HIGH)
58 041100  012737  000200  001000        MOV     #GRPSIZ,BLKSIZ       ; INITIALIZE FOR 1:8 BLOCK GROUPS
59 041106  000137  041630                JMP     600$                 ; TEST FOR EOM BEFORE ACCESS
60
```

```
   1
   2                                   ;        ACCESS NEXT GROUP OF BLOCKS
   3 041112                           310$:
   4 041112   104422                           TRAP     C$BRK
   5 041114   005037   002446                  CLR      LBN                    ; RESET TEST LBN (LOW)
   6 041120   005037   002450                  CLR      LBN+2                  ; RESET TEST LBN (HIGH)
   7 041124   005037   002516                  CLR      LBNSTAT                ; RESET LBN STATUS
   8 041130   005037   002520                  CLR      RPLSTAT                ; RESET REPLACEMENT STATUS
   9 041134   105037   002515                  CLRB     SFLAG                  ; RESET SEARCH FLAG
  10
  11 041140   012737   044472   002340         MOV      #IBUFF,UADR            ; LOAD IMAGE BUFFER ADDRESS
  12 041146   004737   030140                  JSR      PC,INITAC              ; INITIALIZE ACCESS PARAMETERS
  13
  14 041152   012737   000000   002346         MOV      #0,CMDMOD              ; CLEAR MODIFIERS
  15                                                                           ; SETUP TO ISSUE OP.ACC COMMAND
     041160   012700   000020                  MOV      #OP.ACC,R0             ; LOAD MSCP OPCODE
     041164   004737   030204                  JSR      PC,ISSUE               ; ISSUE COMMAND TO THE XDA
     041170   042104                           1000$                          ; HERE IF FATAL ERROR,
     041172   001520                           BEQ      500$                   ; BRANCH IF COMMAND SUCCESSFUL,
  16
  17                                   ;        ERROR RETURN:   CALCULATE ERROR LBN.
  18 041174                           320$:
  19 041174   013701   002324                  MOV      HCOM,R1                ; LOAD COMM AREA ADDRESS
  20 041200   016103   000034                  MOV      HC.MPK+P.BCNT(R1),R3   ; UNLOAD ACCESS
  21 041204   016104   000036                  MOV      HC.MPK+P.BCNT+2(R1),R4 ;   BYTE COUNT
  22 041210   012700   001000                  MOV      #512.,R0               ; DIVISOR IS BYTES PER BLOCK
  23 041214   004737   034436                  JSR      PC,DIVIDE              ; DIVIDE
  24 041220   060337   002562                  ADD      R3,GLBN                ; ADVANCE LBN
  25 041224   005537   002564                  ADC      GLBN+2                 ;   TO BAD BLOCK
  26 041230   013737   002562   002446         MOV      GLBN,LBN               ; TRANSFER GROUP START
  27 041236   013737   002564   002450         MOV      GLBN+2,LBN+2           ;   TO LBN
  28
  29                                   ;        CHECK FOR DATA ERROR.
  30 041244                           350$:
  31 041244   013700   002354                  MOV      EMSTAT,R0              ; LOAD STATUS
  32 041250   042700   177740                  BIC      @#CST.MSK,R0           ; CLEAR ANY SUBCODES
  33 041254   022700   000010                  CMP      #ST.DAT,R0             ; TEST FOR DATA ERROR
  34 041260   001414                           BEQ      400$                   ; BRANCH IF YES
  35
  36                                   ;        UNRECOGNIZED ERROR CODE
  37 041262                           380$:
  38 041262   012737   012246   002552         MOV      #ERM303,BASEMSG        ; "FAILED ACCESS..."
  39 041270   012737   000140   002370         MOV      #B.MSCP,EXTMSK         ; FLAG ALL FIELDS
  40 041276   104455                           TRAP     C$ERDF
     041300   000457                           .WORD    303
     041302   012043                           .WORD    HDR301
     041304   015570                           .WORD    ERR302
  41 041306   000137   042104                  JMP      1000$                  ; EXIT TEST
  42
```

```
    1
    2                                  ;        PROCESS DATA ERROR.
    3 041312                    400$:
    4 041312  004737  023604            JSR     PC,READLBN          ; READ SUSPECT LBN
    5 041316  042104                    1000$                       ; ERROR RETURN
    6 041320  001004                    BNE     420$                ; BRANCH IF REPLACE
    7
    8 041322  004737  017562            JSR     PC,CACHTST          ; UPDATE CACHING TABLE
    9 041326  042104                    1000$                       ; ERROR EXIT
   10 041330  001407                    BEQ     440$                ; BRANCH IF NO REPLACE
   11 041332                    420$:
   12 041332  032737  000200  002516    BIT     #TF.BBR,LBNSTAT     ; TEST FOR BBR
   13 041340  001403                    BEQ     440$                ; BRANCH IF NO
   14 041342                    425$:
   15 041342  005737  002160            TST     ENBRPL              ; TEST FOR REPLACEMENT ENABLED
   16 041346  001011                    BNE     450$                ; BRANCH IF YES
   17 041350                    440$:
   18 041350  004737  020002            JSR     PC,TYPSTAT          ; ELSE, TYPE INFO
   19 041354  062737  000001  002562    ADD     #1,GLBN             ; BUMP TO NEXT LBN
   20 041362  005537  002564            ADC     GLBN+2              ; ADD CARRY
   21 041366  000137  041630            JMP     600$                ; CONTINUE WITH NEXT LBN
   22
   23                                  ;        REPLACE LBN.
   24 041372                    450$:
   25 041372  012737  000006  002546    MOV     #6,NXTSTEP          ; ENTRY IS STEP6
   26 041400  004737  024156            JSR     PC,REPLACE          ; CALL REPLACEMENT ALGORITHM
   27 041404  042104                    1000$                       ; ERROR RETURN
   28 041406  032737  000400  002516    BIT     #TF.WFE,LBNSTAT     ; TEST FOR WFE
   29 041414  001405                    BEQ     455$                ; BRANCH IF NO
   30 041416  062737  000001  002562    ADD     #1,GLBN             ; ELSE, BUMP LBN
   31 041424  005537  002564            ADC     GLBN+2              ; PAST FER
   32 041430                    455$:
   33 041430  000137  041630            JMP     600$                ; CONTINUE
   34
```

```
  1
  2                                           ;        NORMAL RETURN; CHECK FLAGS
  3 041434                             500$:
  4 041434  132737  000200  002356         BITB    #EF.BBR,FLAGS          ; TEST FLAGS FOR BBR
  5 041442  001007                          BNE     510$                  ; BRANCH IF YES
  6 041444  063737  002572  002562         ADD     BLKCNT,GLBN           ; ELSE, MOVE TO NEXT GROUP
  7 041452  005537  002564                 ADC     GLBN+2                ; ADD CARRY
  8 041456  000137  041630                 JMP     600$                  ; CONTINUE WITH NEXT LBN
  9
 10                                         ;        PROCESS BBR.
 11 041462                             510$:
 12 041462  013704  002324                 MOV     HCOM,R4               ; LOAD HOST COMM AREA ADDRESS
 13 041466  016437  000054  002446         MOV     HC.MPK+P.FBBK(R4),LBN ; UNLOAD FIRST BAD BLOCK
 14 041474  016437  000056  002450         MOV     HC.MPK+P.FBBK+2(R4),LBN+2
 15 041502  013737  002446  002562         MOV     LBN,GLBN
 16 041510  013737  002450  002564         MOV     LBN+2,GLBN+2
 17
 18                                         ;        REREAD LBN.
 19 041516                             520$:
 20 041516  004737  023604                 JSR     PC,READLBN            ; READ SUSPECT LBN
 21 041522  042104                          1000$                        ; ERROR RETURN
 22 041524  001004                          BNE     530$                 ; BRANCH IF REPLACE
 23
 24 041526  004737  017562                 JSR     PC,CACHTST            ; UPDATE CACHING TABLE
 25 041532  042104                          1000$                        ; ERROR RETURN
 26 041534  001403                          BEQ     540$                 ; BRANCH IF NO REPLACE
 27 041536                             530$:
 28 041536  005737  002160                 TST     ENBRPL               ; TEST FOR REPLACEMENT ENABLED
 29 041542  001011                          BNE     550$                 ; BRANCH IF YES
 30 041544                             540$:
 31 041544  004737  020002                 JSR     PC,TYPSTAT           ; ELSE, TYPE INFO
 32 041550  062737  000001  002562         ADD     #1,GLBN              ; BUMP TO NEXT LBN
 33 041556  005537  002564                 ADC     GLBN+2               ; ADD CARRY
 34 041562  000137  041630                 JMP     600$                 ; CONTINUE WITH NEXT LBN
 35
 36                                         ;        REPLACE LBN.
 37 041566                             550$:
 38 041566  012737  000006  002546         MOV     #6,NXTSTEP           ; ENTRY IS STEP6
 39 041574  004737  024156                 JSR     PC,REPLACE           ; CALL REPLACEMENT ALGORITHM
 40 041600  042104                          1000$                        ; ERROR RETURN
 41 041602  032737  000400  002516         BIT     #TF.WFE,LBNSTAT      ; TEST FOR WFE
 42 041610  001405                          BEQ     555$                 ; BRANCH IF NO
 43 041612  062737  000001  002562         ADD     #1,GLBN              ; ELSE, BUMP LBN
 44 041620  005537  002564                 ADC     GLBN+2               ; PAST FER
 45 041624                             555$:
 46 041624  000137  041630                 JMP     600$                 ; CONTINUE
 47
```

```
  1
  2                                      ;       TEST FOR END OF MEDIA; RESET BYTE COUNT
  3 041630                          600$:
  4 041630  013701  002604                 MOV     MAXLBN,R1              ; LOAD TOTAL LBN'S (LOW)
  5 041634  013702  002606                 MOV     MAXLBN+2,R2           ; LOAD TOTAL LBN'S (HIGH)
  6 041640                          605$:
  7 041640  163702  002564                 SUB     GLBN+2,R2             ; SUBTRACT START LBN (HIGH)
  8 041644  103466                          BLO     800$                  ; DONE IF GLBN+2 > UNSIZ+2
  9 041646  101030                          BHI     630$                  ; CONTINUE IF GLBN+2 < UNSIZ+2
 10
 11 041650  163701  002562                 SUB     GLBN,R1               ; ELSE, SUBTRACT START LBN (LOW)
 12 041654  101462                          BLOS    800$                  ; DONE IF GLBN >= UNSIZ
 13 041656                          610$:
 14 041656  000240                          NOP
 15 041660  022701  000200                 CMP     #GRPSIZ,R1            ; COMPARE REMAINDER WITH GROUP SIZE
 16 041664  101421                          BLOS    630$                  ; BRANCH IF REMAINING >= GRPSIZ
 17 041666  012737  000000  002566          MOV     #0,GBCNT             ; EI.SE, RESET
 18 041674  012737  000000  002570          MOV     #0,GBCNT+2           ; GROUP BYTE COUNT
 19 041702  010137  002572                 MOV     R1,BLKCNT             ; USE REMAINDER FOR BLOCK COUNT
 20 041706                          620$:
 21 041706  062737  001000  002566          ADD     #BLKSIZ,GBCNT        ; INCREMENT BYTE COUNT (LOW)
 22 041714  005537  002570                 ADC     GBCNT+2              ; ADD CARRY
 23 041720  005301                          DEC     R1                   ; DECREMENT BLOCKS LEFT
 24 041722  001371                          BNE     620$                 ; CONTINUE IF NOT ZERO
 25 041724  000240                          NOP
 26 041726  000411                          BR      640$                 ;
 27 041730                          630$:
 28 041730  012737  000000  002566          MOV     #0,GBCNT             ; ELSE, RESET
 29 041736  012737  000001  002570          MOV     #1,GBCNT+2           ; BYTE COUNT
 30 041744  012737  000200  002572          MOV     #GRPSIZ,BLKCNT       ; AND BLOCK COUNT
 31 041752                          640$:
 32 041752  062737  000001  002574          ADD     #1,ACCESS            ; BUMP ACCESS COUNT
 33 041760  032737  000777  002574          BIT     #777,ACCESS          ; IS IT MOD 512.?
 34 041766  001013                          BNE     650$                 ; BRANCH IF NO
 35                                                                       ; ELSE, PRINT CURRENT LBN
 36 041770  013746  002564                 MOV     GLBN+2,-(SP)         ;PUSH GLBN+2 ON STACK
    041774  013746  002562                 MOV     GLBN,-(SP)           ;PUSH GLBN ON STACK
    042000  012746  010166                 MOV     #INF405,-(SP)        ;PUSH #INF405 ON STACK
    042004  004137  033254                 JSR     R1,LPNTB                    ;CALL LPNTB PRINT ROUTINE
    042010  000006                          .WORD   ARG.CT               ;ARGUMENT COUNT + 2
 37 042012  004737  020650                 JSR     PC,RNTIME            ; AND RUNTIME
 38 042016                          650$:
 39 042016  000137  041112                 JMP     310$                 ; ACCESS NEXT GROUP
```

```
    1
    2 042022                          800$:
    3 042022   013746   002564                MOV    GLBN+2,-(SP)         ;PUSH GLBN+2 ON STACK
      042026   013746   002562                MOV    GLBN,-(SP)           ;PUSH GLBN ON STACK
      042032   012746   010166                MOV    #INF405,-(SP)        ;PUSH #INF405 ON STACK
      042036   004137   033254                JSR    R1,LPNTB                   ;CALL LPNTB PRINT ROUTINE
      042042   000006                         .WORD  ARG.CT               ;ARGUMENT COUNT * 2
    4 042044   004737   020650                JSR    PC,RNTIME            ; DISPLAY RUNTIME
    5
    6 042050   104424                         TRAP   C$DRPT
    7 042052   005737   002650                TST    DEBUG                ; TEST FOR DEBUG
    8 042056   001402                         BEQ    810$                 ; BRAANCH IF NO
    9 042060   000137   041064                JMP    305$                 ; ELSE LOOP
   10 042064                          810$:
   11 042064   005737   002156                TST    DSPRCT               ; TEST FOR DISPLAY
   12 042070   001403                         BEQ    820$                 ; BRANCH IF NO
   13 042072   004737   032364                JSR    PC,DSPDSC            ; ELSE, DISPLAY DESCRIPTORS
   14 042076   042100                         820$                        ; ERROR EXIT
   15 042100                          820$:
   16 042100   000137   040710                JMP    100$                 ; CONTINUE
   17
   18                                 ;      FATAL EXIT; DROP UUT
   19 042104                          1000$:
   20 042104   012737   177777 002442         MOV    #-1,STATUS           ; SET FATAL STATUS
   21 042112   013702   002300                MOV    TBLNDX,R2            ; LOAD UNIT INDEX
   22 042116   012762   000000 002230         MOV    #0,CTLRLST(R2)       ; ZERO CONTROLLER TABLE ADDRESS
   23 042124   013700   002074                MOV    L$LUN,RO
      042130   104451                         TRAP   C$DODU
   24 042132   000137   040710                JMP    100$                 ; AND CONTINUE
   25
   26                                 ;      TEST EXIT
   27 042136                          1200$:
   28 042136   104432                         TRAP   C$EXIT
      042140   000002                         .WORD  L10027-.
   29
   30                                         .EVEN
   31
   32 042142                          L10027:
      042142   104401                         TRAP   C$ETST
   33
   35
```

```
  1                                    .SBTTL   HARDWARE PARAMETER CODING SECTION
  3                                 ;++
  4                                 ; THE HARDWARE 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 042144   000027                    .WORD L10030-L$HARD/2
    042146                         L$HARD::
 14          000000                H.UBA   = 0                    ;UNIBUS ADDRESS
 15          000002                H.VEC   = 2                    ;XDA VECTOR
 16          000004                H.BRL   = 4                    ;BR LEVEL
 17          000006                H.BST   = 6                    ;BURST RATE
 18          000010                H.DRV   = 10                   ;DRIVE NUMBER
 20                                                               ;PRINT 'UNIBUS ADDRESS OF XDA?'
 21 042146   000031                    .WORD   T$CODE
    042150   042224                    .WORD   MSGUBA
    042152   160000                    .WORD   T$LOLIM
    042154   177774                    .WORD   T$HILIM
 22                                                               ;PRINT 'VECTOR?'
 23 042156   001031                    .WORD   T$CODE
    042160   042252                    .WORD   MSGVEC
    042162   000004                    .WORD   T$LOLIM
    042164   000774                    .WORD   T$HILIM
 24                                                               ;PRINT 'BR LEVEL?'
 25 042166   002052                    .WORD   T$CODE
    042170   042261                    .WORD   MSGBRL
    042172   177777                    .WORD   -1
    042174   000004                    .WORD   T$LOLIM
    042176   000007                    .WORD   T$HILIM
 26                                                               ;PRINT 'UNIBUS BURST RATE?'
 27 042200   003052                    .WORD   T$CODE
    042202   042272                    .WORD   MSGBST
    042204   177777                    .WORD   -1
    042206   000000                    .WORD   T$LOLIM
    042210   000077                    .WORD   T$HILIM
 28                                                               ;PRINT 'DRIVE #?'
 29 042212   004052                    .WORD   T$CODE
    042214   042314                    .WORD   MSGLDR
    042216   177777                    .WORD   -1
    042220   000000                    .WORD   T$LOLIM
    042222   000377                    .WORD   T$HILIM
 30                                    .EVEN
    042224                         L10030:
 31
 35 042224    125    116    111    MSGUBA: .ASCIZ   \UNIBUS ADDRESS OF XDA\
 36 042252    126    105    103    MSGVEC: .ASCIZ   \VECTOR\
 37 042261    102    122    040    MSGBRL: .ASCIZ   \BR LEVEL\
 38 042272    125    116    111    MSGBST: .ASCIZ   \UNIBUS BURST RATE\
 39 042314    104    122    111    MSGLDR: .ASCIZ   \DRIVE #\
 40                                    .EVEN
 44
```

```
  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 042324  000000                  .WORD  L10031-L$SOFT/2
    042326             L$SOFT::
 13
 14                                 .EVEN
    042326             L10031:
 15
 16
 17 042326             $PATCH::.BLKW   50.                          ;PATCH AREA (50. WORDS)
 18
 19 042472             DBUFF:: .BLKB   512.    ;BYTES               ;DATA BUFFER
 20 043472             TBUFF:: .BLKB   512.    ;BYTES               ;TEMPORARY BUFFER FOR RCT BLOCK 0
 21 044472             IBUFF:: .BLKB   512.    ;BYTES               ;IMAGE BUFFER FOR LBN BEING REPLACED
 22 045472             BUFF1:: .BLKB   512.    ;BYTES               ;BUFFER 1
 23 046472             BUFF2:: .BLKB   512.    ;BYTES               ;BUFFER 2
 24
 25                                 .EVEN
    047472  047514              .WORD  T$FREE
    047474  000007              .WORD  T$SIZE
    047476             L$LAST::
```

```
     1
     3  047476  000000                      .WORD    0
        047500  000005                      .WORD    L10034-./2-1
        047502                     L10032:
     4  047502  172150                      .WORD    172150           ; UNIBUS ADDRESS
     5  047504  000154                      .WORD    154              ; VECTOR ADDRESS
     6  047506  000005                      .WORD    5.               ; BR LEVEL
     7  047510  000077                      .WORD    63.              ; UNIBUS BURST RATE
     8  047512  000000                      .WORD    0.               ; DRIVE NUMBER
     9  047514                     L10034:
    11          000001            .END
```

```
ACCESS  002574 G     B.ESTA= 000100      COPY    034276       C$PNTF= 000017      DU.QUE= 010000
ADR   = 000020 G     B.EVNT= 000001      CR    = 000015 G     C$PNTS= 000016      DU.SPC= 060000
ALOCM   040054       B.FLAG= 000040      CRLF    014231       C$PNTX= 000015      DU.TER= 040000
ARG.CT= 000006       B.HADD= 0C0002      CTLRLS  002230 G     C$PUTB= 000072      D.BB  = 000002
ASSEMB= 000010       B.MOD = 000004      CTLRTB  002304 G     C$PUTW= 000073      D.BB01= 000004
AUTORC  002152 G     B.MSCP= 000140      CT.AVL= 100000       C$QIO = 000377      D.BB02= 000010
BACKMS  022464       B.PHAS= 000004      CT.BRL= 007000       C$RDBU= 000007      D.BB03= 000014
BACKUP  002652 G     B.RCT = 000003      CT.CMD= 000004       C$REFG= 000047      D.BB04= 000020
BADRBN= 000024 G     B.SADD= 000001      CT.MSG= 000010       C$REL = 000077      D.BB05= 000024
BASEMS  002552 G     B.SAEX= 000004      CT.REQ= 000020       C$RESE= 000033      D.BB06= 000030
BBRCNT  002540 G     B.SAVA= 000002      CT.UNT= 000077       C$REVI= 000003      D.BB07= 000034
BCNT    002334 G     CACHES= 000620 G    CT.U50= 000040       C$RFLA= 000021      D.BB08= 000040
BELL  = 000007 G     CACHID= 000030 G    CT.VEC= 000777       C$RPT = 000025      D.BB09= 000044
BITTBL  020612       CACHLS  002254 G    C$AU  = 000052       C$SEFG= 000046      D.BB10= 000050
BIT0  = 000001 G     CACHTB  002310 G    C$AUTO= 000061       C$SPRI= 000041      D.BB11= 000054
BIT00 = 000001 G     CACHTS  017562      C$BRK = 000022       C$SVEC= 000037      D.BB12= 000060
BIT01 = 000002 G     CALBLK  031332      C$BSEG= 000004       C$TOME= 000076      D.BB13= 000064
BIT02 = 000004 G     CALRBN  031256      C$BSUB= 000002       C.BST = 000004      D.BB14= 000070
BIT03 = 000010 G     CALR1   016042      C$CLCK= 000062       C.DRVN= 000006      D.BB15= 000074
BIT04 = 000020 G     CALR2   016122      C$CLEA= 000012       C.DRVT= 000020      D.BB16= 000100
BIT05 = 000040 G     CALR3   016124      C$CLOS= 000035       C.FLG = 000014      D.BCYL= 000146
BIT06 = 000100 G     CALR4   016126      C$CLP1= 000006       C.HCOM= 000016      D.BE  = 000040
BIT07 = 000200 G     CALR5   016130      C$CPBF= 000074       C.JAD = 000012      D.BEC = 000104
BIT08 = 0C0400 G     CALR6   016132      C$CPME= 000075       C.JSR = 000010      D.BGN1= 000106
BIT09 = 001000 G     CF.ATN= 000200      C$CVEC= 000036       C.REF = 000026      D.BGN2= 000116
BIT1  = 000002 G     CF.MSC= 000100      C$DCLN= 000044       C.SIZE= 000030      D.BGN3= 000126
BIT10 = 002000 G     CF.OTH= 000040      C$DODU= 000051       C.TO  = 000022      D.BGN4= 000136
BIT11 = 004000 G     CF.SHD= 000002      C$DRPT= 000024       C.TOH = 000024      D.CYL = 000400
BIT12 = 010000 G     CF.THS= 000020      C$DU  = 000053       C.UADR= 000000      D.DC  = 000002
BIT13 = 020000 G     CF.576= 000001      C$EDIT= 000003       C.VEC = 000002      D.DCA = 000001
BIT14 = 040000 G     CHRMSG  023110      C$ERDF= 000055       DBM01   035470      D.DCY = 020000
BIT15 = 100000 G     CHRRPL  023104      C$ERHR= 000056       DBM02   035556      D.ECC = 010000
BIT2  = 000004 G     CLKFLG  002206 G    C$ERRO= 000060       DBM03   035601      D.ECYL= 000152
BIT3  = 000010 G     CLOG    034416      C$ERSF= 000054       DBM04   035667      D.END1= 000112
BIT4  = 000020 G     CLRBUF  034240      C$ERSO= 000057       DBUFF   042472 G    D.END2= 000122
BIT5  = 000040 G     CMDIN   002712 G    C$ESCA= 000010       DCODE   002404 G    D.END3= 000132
BIT6  = 000100 G     CMDMOD  002346 G    C$ESEG= 000005       DEBUG   002650 G    D.END4= 000142
BIT7  = 000200 G     CMDPCK  002326 G    C$ESUB= 000003       DELTA   002510 G    D.IW  = 040000
BIT8  = 000400 G     CMDREF  002342 G    C$ETST= 000001       DEVNAM  014225      D.PAT = 000000
BIT9  = 001000 G     CMPUUT  040216      C$EXIT= 000032       DFPTBL  002132 G    D.RET = 001000
BLDCMD  030632       CNTTBL  020636      C$FREQ= 000101       DIAG  = 177777      D.RO  = 004000
BLD28   034654       CNV28   034756      C$FRME= 000100       DIAGMC= 000000      D.SEQ = 000100
BLKCNT  002572 G     CON.A   033444      C$GETB= 000026       DIVIDE  034436      D.SERN= 000156
BLKSTZ= 001000 G     CON.A1  033452      C$GETW= 000027       DIV10   034474      D.SIZE= 000164
BOE   = 000400 G     CON.A2  033466      C$GMAN= 000043       DRIVTB  002306 G    D.TR  = 000020
BRF   = 020000 G     CON.D   033470      C$GPHR= 000042       DRPUUT  040254      D.WC  = 000010
BUFF1   045472 G     CON.H   033502      C$GPRI= 000040       DSPDSC  032364      D.WCA = 000004
BUFF2   046472 G     CON.N   033526      C$INIT= 000011       DSPHEL  032150      D.WO  = 002000
BUSFLA  002614 G     CON.N1  033532      C$INLP= 000020       DSPRCT  002156 G    ECCCNT  002536 G
B.BLOC= 000001       CON.O   033514      C$MANI= 000050       DSPRPL  002154 G    ECODE   002352 G
B.CMDR= 000001       CON.QU  033424      C$MAP = 000102       DSTCNT  002534 G    EF.BBR= 000200
B.CODE= 000002       CON.QX  033442      C$MEM = 000031       DT.AVL= 100000      EF.BBU= 000100
B.COPY= 000002       CON.R   033550      C$MMU = 000103       DT.UNT= 000077      EF.CON= 000036 G
B.DCOD= 000004       CON.R1  033572      C$MSG = 000023       DUP   = 001000      EF.LOG= 000040
B.DERR= 000010       CON.S   033610      C$OPNR= 000034       DU.DFL= 020000      EF.NEW= 000035 G
B.ECCD= 000020       CON.S1  033614      C$OPNW= 000104       DU.FTL= 050000      EF.PWR= 000034 G
B.EMRE= 000010       COPIES  002423 G    C$PNTB= 000014       DU.INF= 030000      EF.RES= 000037 G
```

CZUDLAO BBR Replacement Utility MACRO Y05.03c Friday 19-Apr-85 10:42   Page 146-2
Symbol table

EF.SEX=	000020	ERRLOG	016634	F$MOD =	000000	HC.MPK=	000020	INF602	010320
EF.STA=	000040 G	ERRLO	016734	F$MSG =	000011	HC.MSG=	000004	INF603	010405
EMPTY	002512 G	ERRL1	016772	F$PROT=	000021	HC.PSZ=	000060	INIPAR	023160
EMSTAT	002354 G	ERRL2	017066	F$PWR =	000017	HC.RSZ=	000004	INISWP	022206
ENBERR	002162 G	ERRL3	017224	F$RPT =	000012	HC.SIZ=	000164	INITAC	030140
ENBRPL	002160 G	ERRL4	017462	F$SEG =	000003	HDR001	010472	INITBL	037422
ENBWFE	002164 G	ERRME1	014427	F$SOFT=	000005	HDR101	010667	INITRW	030074
ENDREF	002350 G	ERRMSG	002172 G	F$SRV =	000010	HDR201	011314	INTCPK	030626
ENTRYS=	000200 G	ERRNBR	002170 G	F$SUB =	000002	HDR301	012043	INTRCV	002332 G
ERFLAG	002364 G	ERRREF	002360 G	F$SW =	000014	HDR401	012663	INTSRV	016216 G
ERLAST	016630	ERRSTE	002550 G	F$TEST=	000001	HDR501	013107	INVCHR	014340
ERLDS	002406 G	ERRTYP	002166 G	F.CRLF	014215	HDR601	013173	IPVAL	002322 G
ERLEXI	017514	ERR.SZ=	000010	F.TEXT	014220	HDR701	013731	ISR =	000100 G
ERLFMT	002372 G	ERR.TB	016134	GBCNT	002566 G	HELP =	000000	ISSUE	030204
ERLLBN	002374 G	ERR001	015220 G	GETCDN	033730	HLPFIL	014672	IXE =	004000 G
ERLMSG	004020 G	ERR101	015266 G	GETCHR	023070	HOE =	100000 G	I$AU =	000041
ERLPSI=	000005	ERR201	015336 G	GETCNT	033664	H.BRL =	000004	I$AUTO=	000041
ERLPTB	016616	ERR300	015406 G	GETCNX	033670	H.BST =	000006	I$CLN =	000041
ERLVAL	004026 G	ERR301	015504 G	GETCXX	033736	H.DRV =	000010	I$DU =	000041
ERM001	010527	ERR302	015570 G	GETLBN	023410	H.UBA =	000000	I$HRD =	000041
ERM101	010727	ERR401	015650 G	GETLOG	023126	H.VEC =	000002	I$INIT=	000041
ERM102	011061	ERR601	015720 G	GETVSN	032022	IBE =	010000 G	I$MOD =	000041
ERM103	011145	ERR701	015770 G	GLBN	002562 G	IBUFF	044472 G	I$MSG =	000041
ERM104	011240	EVENT	002362 G	GRPSIZ=	000200 G	IDU =	000040 G	I$PROT=	000040
ERM201	011346	EVL =	000004 G	GTDRVT	032136	IER =	020000 G	I$PTAB=	000041
ERM202	011453	EXIT10	025600	G$CNTO=	000200	INCARN=	000040 G	I$PWR =	000041
ERM203	011711	EXIT11	026122	G$DELM=	000372	INCRLB	002554 G	I$RPT =	000041
ERM204	011743	EXIT12	026520	G$DISP=	000003	INF100	004362	I$SECT=	000000
ERM300	012076	EXIT13	026612	G$EXCP=	000400	INF101	004546	I$SEG =	000041
ERM301	012137	EXIT15	026750	G$HILI=	000002	INF102	004603	I$SETU=	000041
ERM302	012202	EXIT16	027122	G$LOLI=	000001	INF103	004731	I$SFT =	000041
ERM303	012246	EXIT17	027214	G$NO =	000000	INF110	005016	I$SRV =	000041
ERM304	012313	EXIT3	024770	G$OFFS=	000400	INF111	005165	I$SUB =	000041
ERM305	012361	EXIT4	025126	G$OFSI=	000376	INF200	005271	I$TST =	000041
ERM306	012526	EXIT5	025174	G$PRMA=	000001	INF201	005343	J$JMP =	000167
ERM307	012617	EXIT6	025340	G$PRMD=	000002	INF202	005725	KTMEM	002226
ERM401	012727	EXIT9	025450	G$PRML=	000000	INF210	006170	KW.BRL	002212
ERM402	013000	EXTMSK	002370 G	G$RADA=	000140	INF211	006244	KW.CSR	002210
ERM403	013050	E$END =	002100	G$RADB=	000000	INF212	006332	KW.EL	002220
ERM501	013125	E$LOAD=	000035	G$RADD=	000040	INF213	006420	KW.HZ	002216
ERM601	013216	FEF =	000200 G	G$RADL=	000120	INF220	006452	KW.OUT=	000105 G
ERM602	013242	FERCNT	002530 G	G$RADO=	000020	INF221	006523	KW.VEC	002214
ERM603	013257	FFREE	002176 G	G$XFER=	000004	INF301	007002	KW11I	016174 G
ERM604	013334	FLAGS	002356 G	G$YES =	000010	INF302	007027	LBN	002446 G
ERM605	013402	FMERR	040100	HADDR	002400 G	INF310	007061	LBNDX	002522 G
ERM606	013472	FMT.AS	014223	HCECNT	002532 G	INF311	007135	LBNLO =	000014 G
ERM607	013621	FSIZE	002200 G	HCOM	002324 G	INF320	007164	LBNSTA	002516 G
ERM700	013763	F$AU =	000015	HCOMM	040004	INF321	007242	LF =	000012 G
ERM701	014024	F$AUTO=	000020	HC.CCT=	000012	INF322	007630	LINBUF	002716
ERM702	014062	F$BGN =	000040	HC.CEV=	000100	INF323	010043	LMT28	014277
ERM703	014130	F$CLEA=	000007	HC.CMD=	000010	INF324	010065	LOADFL	002202 G
ERM704	014157	F$DU =	000016	HC.CPK=	000104	INF401	010114	LOE =	040000 G
ERRBLK	002174 G	F$END =	000041	HC.ESZ=	000004	INF405	010166	LOG =	000001
ERRC	033632	F$HARD=	000004	HC.INT=	000000	INF406	010206	LOGMSG=	023146
ERRCNT	002475 G	F$HW =	000013	HC.ISZ=	000004	INF407	010224	LOGRPL=	023142
ERRD	033644	F$INIT=	000006	HC.MCT=	000006	INF408	010243	LOT =	000010 G
ERRFMT	002366 G	F$JMP =	000050	HC.MEV=	000014	INF601	010264	LPNT	033302

LPNTB 033254	L$UNIT 002012 G	MD.SCL= 002000	OP.ESP= 000002	P.CPSP= 000042
LPNTF 033244	L.BADR= 000030 G	MD.SEC= 000100	OP.FLU= 000023	P.CRF = 000000
LPNTS 033274	L.CRF = 000000 G	MD.SEQ= 000020	OP.GCS= 000002	P.CTMO= 000020
LPNTX 033264	L.DS = 000060 G	MD.SER= 000400	OP.GSS= 000001	P.CYLS= 000050
L$ACP 002110 G	L.EVNT= 000012 G	MD.SPD= 000001	OP.GUS= 000003	P.DEXT= 000014
L$APT 002036 G	L.FLGS= 000011 G	MD.SSH= 000200	OP.MRD= 000030	P.DFLG= 000017
L$AU 040306 G	L.FMT = 000010 G	MD.SWP= 000004	OP.MWR= 000031	P.DMDT= 000024
L$AUT 002070 G	L.HDCD= 000050 G	MD.VOL= 000002	OP.ONL= 000011	P.DPRG= 000020
L$AUTO 040122 G	L.SDI = 000054 G	MD.WBN= 000100	OP.RD = 000041	P.DTMO= 000024
L$CCP 002106 G	L10000 002144	MD.WBV= 000400	OP.RLC= 000103	P.ELGF= 000034
L$CLEA 040124 G	L10001 002166	MINLBN 002600 G	OP.RPL= 000024	P.FBBK= 000034
L$CO 002032 G	L10002 015264	MODEL 003244	OP.RSD= 000005	P.FLGS= 000011
L$DEPO 002011 G	L10003 015334	MSCP = 000000	OP.SCC= 000004	P.GRPS= 000046
L$DESC 004324 G	L10004 015404	MSCPMS 003330 G	OP.SEX= 000007	P.HSTI= 000020
L$DESP 002076 G	L10005 015502	MSCPND 004226	OP.SHC= 000102	P.HTMO= 000020
L$DEVP 002060 G	L10006 015566	MSCPTB 004230	OP.SSD= 000004	P.LBN = 000034
L$DISP 002124 G	L10007 015646	MSCPVA 003350 G	OP.SUC= 000012	P.MEDI= 000034
L$DLY 002116 G	L10010 015716	MSGADR= 023456	OP.WR = 000042	P.MLUN= 000014
L$DTP 002040 G	L10011 015766	MSGBRL 042261	OSTRE 033422	P.MOD = 000012
L$DTYP 002034 G	L10012 016040	MSGBST 042272	OSTRNG 033352	P.MODE= 000032
L$DU 040146 G	L10013 016162	MSGLDR 042314	O$APTS= 000000	P.OPCD= 000010
L$DUT 002072 G	L10014 016172	MSGPCK 002330 G	O$AU = 000000	P.OTRF= 000014
L$DVTY 004300 G	L10015 016214	MSGUBA 042224	O$BGNR= 000001	P.OVRL= 000034
L$EF 0C2052 G	L10016 016222	MSGVEC 042252	O$BGNS= 000000	P.RBN = 000014
L$ENVI 002044 G	L10017 035716	MULT 034542	O$DU = 000001	P.RBNS= 000056
L$ERRT 002166 G	L10021 040120	MULTRD 027224	O$ERRT= 000001	P.RCTC= 000057
L$ETP 002102 G	L10022 040122	MULTWR 027514	O$GNSW= 000001	P.RCTS= 000054
L$EXP1 002046 G	L10023 040144	M.RA60= 000004	O$POIN= 000001	P.RGID= 000034
L$EXP4 002064 G	L10024 040304	M.RA80= 000001	O$SETU= 000001	P.RGOF= 000040
L$EXP5 002066 G	L10025 040312	M.RA81= 000005	PB 033146	P.SHST= 000042
L$HARD 042146 G	L10026 040622	M.RA82= 000013	PF 033122	P.SHUN= 000040
L$HIME 002120 G	L10027 042142	M.UNKN= 000000	PHASE 002560 G	P.STS = 000012
L$HPCP 002016 G	L10030 042224	NCONF 033406	PNT = 001000 G	P.TIME= 000024
L$HPTP 002022 G	L10031 042326	NCONS 033362	PNTNUM 033742	P.TRKS= 000044
L$HW 002132 G	L10032 047502	NDX.AC= 000002	PNTNUS 033750	P.UADR= 000020
L$ICP 002104 G	L10034 047514	NDX.RD= 000000	PRI = 002000 G	P.UNFL= 000016
L$INIT 035726 G	MANDIS 014473	NDX.RP= 000003	PRIMRY 002434 G	P.UNIT= 000004
L$LADP 002026 G	MANRPL 002150 G	NDX.WR= 000001	PRINTC 033066	P.UNSZ= 000044
L$LAST 047476 G	MATBLK 002500 G	NXMAD 002224	PRI00 = 000000 G	P.UNTI= 000024
L$LOAD 002100 G	MATFLG 002514 G	NXMI 016154 G	PRI01 = 000040 G	P.USEF= 000022
L$LUN 002074 G	MATOFF 002504 G	NXTSTE 002546 G	PRI02 = 000100 G	P.VRSN= 000014
L$MREV 002050 G	MATRBN 002462 G	OLIMIT 014367	PRI03 = 000140 G	P.VSER= 000050
L$NAME 002000 G	MAXLBN 002604 G	ONEFIL= 000001	PRI04 = 000200 G	P1 = 100000 G
L$PRIO 002042 G	MDLNDX 003246	ONLINE 002612 G	PRI05 = 000240 G	P2 = 040000 G
L$PROT 035720 G	MDLSIZ= 000012	OPCODE 002344 G	PRI06 = 000300 G	RA60 003311
L$PRT 002112 G	MDLTBL 003250	OPHELP 002146 G	PRI07 = 000340 G	RA80 003304
L$REPP 002062 G	MDLTEX 003262	OPTIN 002714 G	PS 033216	RA81 003316
L$REV 002010 G	MD.CMP= 040000	OP.ABO= 000001	PTABLE 002302 G	RA82 003323
L$RPT 035132 G	MD.CWB= 000010	OP.ACC= 000020	PTYPE 033350	RBN 002452 G
L$SOFT 042326 G	MD.ERR= 010000	OP.AVA= 000100	PX 033172	RBNLO = 000020 G
L$SPC 002056 G	MD.EXP= 100000	OP.AVL= 000010	P.BCNT= 000014	RBNS 002422 G
L$SPCP 002020 G	MD.FEU= 000001	OP.CCD= 000021	P.BUFF= 000020	RCTBLK 002476 G
L$SPTP 002024 G	MD.IMF= 000002	OP.CMP= 000040	P.CLAS= 000033	RCTCOP 002424 G
L$STA 002030 G	MD.NXU= 000001	OP.DUP= 000101	P.CMST= 000020	RCTERR 016370
L$SW 002146 G	MD.PRI= 000001	OP.ELP= 000003	P.CNCL= 000022	RCTLBN 002456 G
L$TEST 002114 G	MD.RIP= 000001	OP.END= 000200	P.CNTF= 000016	RCTMAS 002426 G
L$TIML 002014 G	MD.SCH= 004000	OP.ERS= 000022	P.CNTI= 000024	RCTMSG 003660 G

```
RCTOFF  002502 G    SA.STP= 100000      SVCGBL= 000000      T$FLAG= 000040      UNKNOW  003274
RCTSIZ  002416 G    SA.S1 = 004000      SVCINS= 000000      T$FREE= 047514      UNSIZ   002430 G
RCTTBL  002312 G    SA.S2 = 010000      SVCSUB= 000000      T$GMAN= 000000      UNUSE   002440 G
RCTVAL  003670 G    SA.S3 = 020000      SVCTAG= 000000      T$HILI= 000377      VBUFF1  002473 G
RDCNT   002556 G    SA.S4 = 040000      SVCTST= 000000      T$LAST= 000001      VBUFF2  002474 G
RDSTAT  002524 G    SA.TST= 100000      SWMSG0  022525      T$LOLI= 000000      VIBUFF  002472 G
RDVALD  002470 G    SA.VCE= 000177      SWMSG1  022553      T$LSYM= 010000      VOLSN = 000000 G
READLB  023604      SA.VEC= 000177      SWMSG2  022613      T$LTNO= 000002      VSNBUF  002620 G
RECOVR  002466 G    SA.WRP= 040000      SWMSG3  022644      T$NEST= 177777      VSNPTR  002616 G
REPEAT  002610 G    SB.DST= 000150      SWMSG4  022707      T$NSO = 000000      VWP   = 000002 G
REPLAC  024156      SB.ECC= 000350      SWMSG5  022753      T$NS1 = 000005      W     = 000000
REPLY   002654 G    SB.HCE= 000110      SWMSG6  022777      T$PCNT= 000000      WAIT    031072
RESCAN  002513 G    SCNDRY  002436 G    SWMSG7  023024      T$PTAB= 010033      WBC   = 000001 G
RESET   016224      SEARCH  031400      S$LSYM= 010000      T$PTHV= 000001      WFECNT  002542 G
RFLAGS= 000010 G    SETTO   034334      TBLNDX  002300 G    T$PTNU= 000001      WRSTAT  002526 G
RG.FLG= 040000      SFLAG   002515 G    TBUFF   043472 G    T$SAVL= 177777      XACCES  004257
RG.OWN= 100000      SFPTBL  002146 G    TF.BAD= 100000 G    T$SEGL= 177777      XBLOCK  003700
RNTIM   014233      SNDCMD  030744      TF.BBR= 000200 G    T$SIZE= 000007      XCMDRF  003370
RNTIME  020650      SND.S1  037424      TF.BBU= 000100 G    T$SUBN= 000000      XCODE   003421
RNTIMX  021036      SND.S2  037430      TF.DST= 000004 G    T$TAGL= 177777      XCOPY   003732
RNTIM1  014266      SND.S3  037434      TF.ECC= 000010 G    T$TAGN= 010035      XDAINT  036670
RNTIM2  014274      STATBL  020624      TF.FER= 000001 G    T$TEMP= 000000      XDAIST  037200
RPLSTA  002520 G    STATUS  002442 G    TF.HCE= 000002 G    T$TEST= 000002      XDARSD  040002
RSP.CK  037542      STEP    002544 C    TF.LOG= 000040 G    T$TSTM= 177777      XDARSP  037606
RSP.S1  037440      STEPER  024226      TF.SEX= 000020 G    T$TSTS= 000001      XDASRV  016164 G
RSP.S2  037452      STEPFT  024250      TF.WFE= 000400 G    T$$AU = 010025      XDCODE  004112
RSP.S3  037472      STEPSI= 000044      TIMDAT= 000044 G    T$$AUT= 010022      XDHEX   004217
RSP.S4  037510      STEPTB  024112      TINDEX= 000012      T$$CLE= 010023      XDINFO  004152
RTFLAG  002204 G    STEP10  025460      TMPSTA  002444 G    T$$DAT= 010034      XECODE  003537
SADDR   002314 G    STEP11  025610      TRKSIZ  002420 G    T$$DU = 010024      XENDRF  003506
SAEXP   002320 G    STEP12  026132      TTYIN   002666 G    T$$HAR= 010030      XESTAT  003625
SARMSG  003116 G    STEP13  026530      TTYOUT  033242      T$$HW = 010000      XEVENT  004034
SARVAL  003126 G    STEP15  026640      TXT000  014705      T$$INI= 010021      XEVNT   004035
SAVAL   002316 G    STEP16  026760      TXT001  014762      T$$MSG= 010012      XFLAGS  003572
SA.BST= 000374      STEP17  027132      TXT002  014767      T$$PC = 000001      XHADDR  004063
SA.CMD= 034000      STEP3   024260      TXT005  014774      T$$PRO= 010020      XLBN    014575
SA.CME= 000070      STEP4   024772      TXT006  015031      T$$PTA= 010033      XMOD    003454
SA.CM1= 004000      STEP5   025136      TX.BBR  015150      T$$RPT= 010017      XOFF  = 000023 G
SA.CNT= 000360      STEP6   025204      TX.BBU  015155      T$$SOF= 010031      XON   = 000021 G
SA.CTP= 003400      STEP9   025350      TX.BLK  015174      T$$SRV= 010016      XPHASE  003764
SA.DI = 000400      STROFF  002506 G    TX.DST  015213      T$$SW = 010001      XRBN    014615
SA.ERC= 003777      ST.ABO= 000002      TX.ECH  015143      T$$TES= 010027      XRCT    014537
SA.ERR= 100000      ST.AOL= 000400      TX.FER  015136      T1      040314 G     XREAD   004242
SA.GO = 000001      ST.AVL= 000004      TX.HCE  015206      T2      040624 G     XREPLA  004267
SA.INE= 000200      ST.CMD= 000001      TX.LB1  015060      UADR    002340 G     XSADDR  003165
SA.INT= 000200      ST.CMP= 000007      TX.LB2  015100      UAM   = 000200 G     XSAEXP  003214
SA.LFC= 000002      ST.CNT= 000012      TX.LOG  015167      UF.CMR= 000001      XSAVAL  003136
SA.MCV= 000017      ST.DAT= 000010      TX.PRI  015117      UF.CMW= 000002      XSTAT   003626
SA.MP = 000100      ST.DIA= 000037      TX.SEC  015124      UF.INA= 040000      XSTEP   014635
SA.MSE= 000007      ST.DRV= 000013      TX.SEX  015162      UF.RPL= 100000      XWRITE  004250
SA.MSG= 003400      ST.HST= 000011      TX.UNS  015131      UF.SCH= 004000      X$ALWA= 000000
SA.MS1= 000400      ST.MFE= 000005      TX.WFE  015201      UF.SCL= 002000      X$FALS= 000040
SA.NV = 002000      ST.MSK= 000037      TYPSTA  020002      UF.WBN= 000100      X$OFFS= 000400
SA.NVE= 000400      ST.OFL= 000003      T$ARGC= 000002      UF.WPH= 020000      X$TRUE= 000020
SA.PRG= 000001      ST.SUB= 000040      T$CODE= 004052      UF.WPS= 001000      $MUL  = 000037
SA.QB = 001000      ST.SUC= 000000      T$ERRN= 000457      UF.576= 000004      $PATCH  042326 G
SA.STE= 000200      ST.WPR= 000006      T$EXCP= 000000      UNITIN  021050
```

CZUDLAO BBR Replacement Utility MACRO Y05.03c Friday 19-Apr-85 10:42   Page 146-5
Symbol table

. ABS. 047514   000   (RW,I,GBL,ABS,OVR)
Errors detected: 0

*** Assembler statistics

Work file reads: 986
Work file writes: 860
Size of work file: 38688 Words (152 Pages)
Size of core pool: 14080 Words (55 Pages)
Operating system: RT-11 (Under RTEM-11)

Elapsed time: 00:05:30.00
ZUDLAO.BIN/DS:GBL/EN:ABS:AMA,ZUDLAO/CR=SVC35R/ML,ZUDLAO.MAC

```
$MUL    92-22    92-22#
$PATCH 145-17#
ACCESS  21-89#   65-16*   65-17*   142-32*  142-33
ADR      5-9#
ALOCM   59-34   123-32   125-10   126-8    127-9    131-16   132-17#
ARG.CT  36-12    36-12    36-12    36-12#   36-12#   36-12#   36-13    36-13    36-13    36-13#   36-13#   36-13#   36-27    36-27
        36-27#   36-27#   37-11    37-11    37-11#   37-11#   37-36    37-36    37-36    37-36    37-36    37-36#   37-36#   37-36#
        37-36#   37-36#   37-59    37-59    37-59    37-59    37-59#   37-59#   37-59#   37-59#   37-80    37-80    37-80    37-80
        37-80#   37-80#   37-80#   37-80#   38-11    38-11    38-11#   38-11#   38-32    38-32    38-32#   38-32#   39-11    39-11
        39-11#   39-11#   39-12    39-12    39-12    39-12#   39-12#   39-12#   40-17    40-17    40-17    40-17#   40-17#   40-17#
        40-25    40-25    40-25#   40-25#   46-44    46-44    46-44    46-44    46-44    46-44    46-44#   46-44#   46-44#
        46-44#   46-44#   46-45    46-45    46-45    46-45#   46-45#   46-45#   48-20    48-20    48-20    48-20    48-20#   48-20#
        48-20#   48-20#   49-17    49-17    49-17    49-17    49-17#   49-17#   49-17#   49-17#   49-18    49-18    49-18    49-18#
        49-18#   49-18#   50-21    50-21    50-21    50-21    50-21#   50-21#   50-21#   50-21#   50-32    50-32    50-32    50-32#
        50-32#   50-32#   50-34    50-34    50-34#   50-34#   50-39    50-39    50-39    50-39#   50-39#   50-39#   53-32    53-32
        53-32    53-32    53-32#   53-32#   53-32#   53-32#   53-33    53-33    53-33    53-33#   53-33#   53-33#   53-34    53-34
        53-34    53-34    53-34#   53-34#   53-34#   53-34#   53-35    53-35    53-35    53-35#   53-35#   53-35#   53-45    53-45
        53-45    53-45    53-45#   53-45#   53-45#   53-45#   53-46    53-46    53-46    53-46#   53-46#   53-46#   53-57    53-57
        53-57    53-57    53-57#   53-57#   53-57#   53-57#   53-58    53-58    53-58    53-58#   53-58#   53-58#   54-7     54-7
        54-7     54-7     54-7#    54-7#    54-7#    54-7#    54-8     54-8     54-8     54-8#    54-8#    54-8#    54-17    54-17
        54-17    54-17    54-17#   54-17#   54-17#   54-17#   54-18    54-18    54-18    54-18#   54-18#   54-18#   55-12    55-12
        55-12    55-12    55-12#   55-12#   55-12#   55-12#   55-13    55-13    55-13    55-13#   55-13#   55-13#   55-22    55-22
        55-22    55-22    55-22#   55-22#   55-22#   55-22#   55-23    55-23    55-23    55-23#   55-23#   55-23#   56-25    56-25
        56-25    56-25#   56-25#   56-25#   56-29    56-29    56-29    56-29#   56-29#   56-29#   56-34    56-34    56-34    56-34#
        56-34#   56-34#   59-58    59-58    59-58    59-58    59-58    59-58    59-58#   59-58#   59-58#   59-58#   59-58#   59-58#
        59-62    59-62    59-62    59-62    59-62    59-62    59-62#   59-62#   59-62#   59-62#   59-62#   59-62#   60-8     60-8
        60-8#    60-8#    61-24    61-24    61-24#   61-24#   62-17    62-17    62-17#   62-17#   66-24    66-24    66-24#   66-24#
        66-49    66-49    66-49    66-49    66-49#   66-49#   66-49#   66-49#   98-21    98-21    98-21    98-21#   98-21#   98-21#
        98-56    98-56    98-56#   98-56#   98-57    98-57    98-57    98-57    98-57    98-57    98-57#   98-57#   98-57#
        98-57#   98-57#   98-57#   98-57#   98-67    98-67    98-67    98-67    98-67    98-67    98-67#   98-67#   98-67#
        98-67#   98-67#   98-75    98-75    98-75    98-75    98-75    98-75    98-75#   98-75#   98-75#   98-75#   98-75#   98-75#
        98-82    98-82    98-82    98-82    98-82#   98-82#   98-82#   98-82#   101-44   101-44   101-44#  101-44#  103-26   103-26
        103-26#  103-26#  117-42   117-42   117-42#  117-42#  117-45   117-45   117-45#  117-45#  117-67   117-67   117-67#  117-67#
        117-68   117-68   117-68   117-68   117-68   117-68   117-68   117-68#  117-68#  117-68#  117-68#  117-68#  117-68#
        117-68#  117-68#  117-73   117-73   117-73   117-73   117-73   117-73   117-73#  117-73#  117-73#  117-73#  117-73#  117-73#
        117-78   117-78   117-78#  117-78#  118-7    118-7    118-7    118-7#   118-7#   118-7#   118-15   118-15   118-15   118-15
        118-15   118-15#  118-15#  118-15#  118-15#  118-15#  118-24   118-24   118-24   118-24#  118-24#  118-24#  118-33   118-33
        118-33   118-33   118-33   118-33   118-33#  118-33#  118-33#  118-33#  118-33#  121-12   121-12   121-12#  121-12#
        123-8    123-8    123-8#   123-8#   135-13   135-13   135-13   135-13#  135-13#  135-13#  135-16   135-16   135-16   135-16#
        135-16#  135-16#  137-13   137-13   137-13#  137-13#  137-56   137-56   137-56#  137-56#  138-9    138-9    138-9#   138-9#
        138-10   138-10   138-10#  138-10#  138-11   138-11   138-11#  138-11#  138-32   138-32   138-32#  138-32#  142-36   142-36
        142-36   142-36   142-36#  142-36#  142-36#  142-36#  143-3    143-3    143-3    143-3    143-3#   143-3#   143-3#   143-3#
ASSEMB   1-I02    1-I02
AUTORC   4-14#   61-15    62-38
B.BLOC  26-5#    26-9
B.CMDR  25-5#    88-103
B.CODE  25-6#
B.COPY  26-6#    26-9
B.DCOD  27-7#
B.DERR  27-8#
B.ECOD  25-9#
B.EMRE  25-8#    88-103
B.ESTA  25-11#   25-13    58-12    58-41    59-12
B.EVNT  27-5#    47-44    48-5     48-17    50-17    51-5
B.FLAG  25-10#   25-13
B.HADD  27-6#
```

CZUDLAO BBR Replacement Utility MACRO Y05.03c Friday 19-Apr-85 10:42 Page S-2
Cross reference table (CREF V05.03)

```
B.MOD    25-7#
B.MSCP   25-13#   73-39    79-41    79-75    79-85    82-46    84-40    85-39    85-58    88-79    88-103   88-110   139-39
B.PHAS   26-7#    70-30
B.RCT    26-9#
B.SADD   23-5#    45-33    88-87    88-95    128-69   129-35   129-118  130-43   130-53
B.SAEX   23-7#    129-118  130-43
B.SAVA   23-6#    88-87    88-95    128-69   129-118  130-43   130-53
BACKMS   62-71#   123-9
BACKUP   22-4#    123-10   123-13   127-25
BADRBN   18-7#    71-8     71-9     77-42*   77-43*
BASEMS   21-81#   36-12    36-27    37-11    37-59    37-80    38-11    38-32    39-11    45-32*   47-45*   48-6*    48-18*   50-18*
         51-6*    52-40*   58-11*   58-21*   58-40*   59-11*   70-31*   73-38*   76-36*   79-40*   79-74*   79-84*   82-45*   84-39*
         84-66*   85-38*   85-57*   85-79*   88-78*   88-86*   88-94*   88-102*  88-109*  98-35*   128-68*  128-94*  129-34*  129-117*
         130-42*  130-52*  132-33*  139-38*
BBRCNT   21-74#   54-19*   65-39*   117-68
BCNT     20-9#    86-27    86-28    109-15   110-15
BELL     29-17
BITC     5-9#     15-24    18-15    21-53    23-5     25-5     26-5     27-5
BIT00    5-9      5-9#
BIT01    5-9      5-9#
BIT02    5-9      5-9#
BIT03    5-9      5-9#
BIT04    5-9      5-9#
BIT05    5-9      5-9#
BIT06    5-9      5-9#
BIT07    5-9      5-9#
BIT08    5-9      5-9#
BIT09    5-9      5-9#
BIT1     5-9#     15-23    18-16    21-54    23-6     25-6     26-6     27-6
BIT10    5-9#     15-15
BIT11    5-9#     15-14
BIT12    5-9#     15-13    78-57    98-70
BIT13    5-9#     15-12    18-18    78-54    94-72    98-63    98-70
BIT14    5-9#     15-11    18-19    78-89    98-78
BIT15    5-9#     14-14    15-10    18-20    21-62    107-30
BIT2     5-9#     14-20    15-22    21-55    23-7     25-7     26-7     27-7
BIT3     5-9#     14-18    15-21    21-56    25-8     27-8
BIT4     5-9#     14-16    15-20    21-57    25-9
BIT5     5-9#     14-15    15-19    21-58    25-10
BIT6     5-9#     15-18    21-59    25-11
BIT7     5-9#     18-17    21-60
BIT8     5-9#     15-17    21-61
BIT9     5-9#     15-16
BITTBL   53-54    53-60    55-30#
BLD28    116-11#
BLDCMD   88-28    89-32#
BLKCNT   21-88#   141-6    142-19*  142-30*
BLKSIZ   17-4#    20-9     138-58*  142-21
BOE      5-9#
BRF      18-18#   71-5     77-41
BUFF1    78-50    81-28    145-22#
BUFF2    78-86    81-36    145-23#
BUSFLA   21-94#   120-40*
C$AU     1-I02#   136-15
C$AUTO   1-I02#   133-12
C$BRK    1-I02#   45-20    91-35    129-15   130-33   139-4
```

```
C$BSEG    1-I02#
C$BSUB    1-I02#
C$CLCK    1-I02# 121-7    121-9
C$CLEA    1-I02# 134-17
C$CLOS    1-I02#  97-58
C$CLP1    1-I02#
C$CPBF    1-I02#
C$CPME    1-I02#
C$CVEC    1-I02#  45-37   129-30
C$DCLN    1-I02# 123-16   127-28   132-35
C$DODU    1-I02# 137-71   137-77   143-23
C$DRPT    1-I02# 137-57   143-6
C$DU      1-I02# 135-26
C$EDIT    1-I02#   1-I25
C$ERDF    1-I02#  45-34   47-46    48-7     48-19    50-19    51-7     52-42    58-13    58-23    59-13    84-68   85-81   88-80
          88-88   88-96   88-104   88-111   128-70   128-95   129-36   129-119  130-44   130-54   139-40
C$ERHR    1-I02#  58-42   73-40    76-38    79-42    79-76    79-86    82-47    84-41    85-40    85-59
C$ERRO    1-I02#
C$ERSF    1-I02# 132-3
C$ERSO    1-I02#  70-32   98-36
C$ESCA    1-I02#
C$ESEG    1-I02#
C$ESUB    1-I02#
C$ETST    1-I02# 137-82   143-32
C$EXIT    1-I02# 127-34   134-13   137-7    137-11   137-72   137-78   138-7    143-28
C$FREQ    1-I02#
C$FRME    1-I02#
C$GETB    1-I02#  97-37
C$GETW    1-I02#
C$GMAN    1-I02#  61-20   63-28    64-27    66-33
C$GPHR    1-I02# 124-24
C$GPRI    1-I02#
C$INIT    1-I02# 132-39
C$INLP    1-I02#
C$MANI    1-I02# 124-7    137-9    138-34
C$MAP     1-I02#
C$MEM     1-I02# 120-34   123-18
C$MMU     1-I02#
C$MSG     1-I02#  36-16   36-37    37-21    37-46    37-69    37-90    38-21    38-42    39-18
C$OPNR    1-I02#  97-28
C$OPNW    1-I02#
C$PNTB    1-I02#  97-85   99-20
C$PNTF    1-I02#  99-18
C$PNTS    1-I02#  99-24
C$PNTX    1-I02#  99-22
C$PUTB    1-I02#
C$PUTW    1-I02#
C$QIO     1-I02#
C$RDBU    1-I02# 120-37
C$REFG    1-I02# 122-5    122-10   122-15
C$REL     1-I02#
C$RESE    1-I02#   1-I02# 120-28
C$REVI    1-I02#   1-I25
C$RFLA    1-I02#
C$RPT     1-I02# 118-52
C$SEFG    1-I02#
```

```
C$SPRI    1-I02◊  127-30
C$SVEC    1-I02◊   45-18    57-44    121-19   129-27
C$TOME    1-I02◊
C.BST    14-25◊  128-100
C.DRVN   14-26◊   36-12    59-58    89-51
C.DRVT   14-31◊   96-14   125-13*  126-4    126-9*
C.FLG    14-29◊   88-39*   90-44*   91-28    91-54*  125-31*  129-104* 129-107*
C.HCOM   14-30◊  125-12*
C.JAD    14-28◊
C.JSR    14-27◊   57-41
C.REF    14-34◊  125-34*
C.SIZE   14-36◊  125-9
C.TO     14-32◊   91-23    91-42   125-32*  130-24   130-39
C.TOH    14-33◊   91-39   125-33*  130-36
C.UADR   14-23◊  129-25
C.VEC    14-24◊   57-34
CACHES   17-3◊    52-37   110-37   127-8    127-15
CACHID   18-8◊    80-30    83-28
CACHLS   19-28◊  123-23   127-5    127-10*
CACHTB   19-33◊   52-22   118-24   118-27   127-14*
CACHTS   52-20◊  137-41   140-8    141-24
CALBLK   78-30    78-66    93-23◊   94-40
CALR1    36-31    37-15    37-40    37-63    37-84    38-15    38-36    40-11◊   41-27
CALR2    40-32◊   41-28
CALR3    40-39◊   41-29
CALR4    41-5◊    41-30
CALR5    41-12◊   41-31
CALR6    41-19◊   41-32    41-33    41-34
CALRBN   92-20◊   94-67    94-80    98-46
CF.576   10-42◊
CF.ATN   10-37◊
CF.MSC   10-38◊
CF.OTH   10-39◊
CF.SHD   10-41◊
CF.THS   10-40◊   58-35
CHRMSG   63-21*   63-24◊   63-28
CHRRPL   63-22*   63-25◊   63-28    63-30
CLKFLG   19-11◊
CLOG    112-11◊  128-32   128-40
CLRBUF   73-29    84-31   109-12◊
CMDIN    22-9◊
CMDMOD   20-16◊   25-32    67-36*   67-72*   79-36*   79-46*   79-53*   82-33*   84-30*   84-59*   85-30*   85-49*   85-72*   89-43
        139-14*
CMDPCK   20-5◊   131-24*
CMDREF   20-14◊   25-30    65-44*   88-60    90-24*   90-25
CMPUUT  135-13   135-21◊
CNTTBL   53-59*   55-44◊
CNV28    66-36   117-17◊
CON.A   102-15◊  105-51
CON.A1  102-17◊  102-20
CON.A2  102-18   102-21◊
CON.D   102-25◊  105-51
CON.H   102-31◊  105-51
CON.N   103-10◊  105-51
CON.N1  103-11◊  103-13
CON.O   103-4◊   105-51
```

```
CON.QU  102-5◊    102-9      105-51
CON.QX  102-7     102-10◊
CON.R   103-19◊   105-51
CON.R1  103-21    103-26◊
CON.S   103-32◊   105-51
CON.S1  103-33◊   103-35
COPIES  21-7◊     46-49      58-27*    84-51     85-64
COPY    69-33     75-30      75-46     77-30     78-48     78-84     110-12◊
CR      29-15     97-30      97-51     99-11     103-11    134-10
CRLF    34-11◊    40-25      66-24
CT.AVL  14-14◊
CT.BRL  14-12◊    57-38
CT.CMD  14-20◊    88-39      90-44
CT.MSG  14-18◊    43-21      91-28     91-54
CT.REQ  14-16◊
CT.U50  14-15◊    129-104    129-107
CT.UNT  14-10◊
CT.VEC  14-11◊    57-36
CTLRLS  19-27◊    45-24      123-22    125-5     125-11*   137-16    138-23    143-22*
CTLRTB  19-31◊    36-11      45-27*    57-32     59-55     88-27     88-38     89-34     90-22   91-25   98-18   117-64   125-17*  125-30
        126-3     137-16*    138-23*
D.BB    15-28◊
D.BB01  15-29◊
D.BB02  15-30◊
D.BB03  15-31◊
D.BB04  15-32◊
D.BB05  15-33◊
D.BB06  15-34◊
D.BB07  15-35◊
D.BB08  15-36◊
D.BB09  15-37◊
D.BB10  15-38◊
D.BB11  15-39◊
D.BB12  15-40◊
D.BB13  15-41◊
D.BB14  15-42◊
D.BB15  15-43◊
D.BB16  15-44◊
D.BCYL  16-11◊
D.BE    15-19◊
D.BEC   16-2◊
D.BGN1  16-3◊
D.BGN2  16-5◊
D.BGN3  16-7◊
D.BGN4  16-9◊
D.CYL   15-17◊
D.DC    15-23◊
D.DCA   15-24◊
D.DCY   15-12◊
D.ECC   15-13◊
D.ECYL  16-12◊
D.END1  16-4◊
D.END2  16-6◊
D.END3  16-8◊
D.END4  16-10◊
D.IW    15-11◊
```

```
D.PAT    15-27♦
D.RET    15-16♦
D.RO     15-14♦
D.SEQ    15-18♦
D.SERN   16-13♦
D.SIZE   16-15♦   126-7     126-13
D.TR     15-20♦
D.WC     15-21♦
D.WCA    15-22♦
D.WO     15-15♦
DBMO1    118-7    118-43♦
DBMO2    118-15   118-45♦
DBMO3    118-24   118-47♦
DBMO4    118-33   118-49♦
DBUFF    20-10    59-49     60-6     69-28   69-34   75-25   75-31   75-47   76-27   77-32   78-49   78-85   98-27   145-19♦
DCODE    20-36♦   50-31*    50-32
DEBUG    22-3♦    51-11     117-75   138-37  143-7
DELTA    21-37♦   94-51*    94-55    94-96*  94-98*  94-100
DEVNAM   34-10♦   59-58     120-41*
DFPTBL    3-10♦
DIAG      7-13♦
DIAGMC    1-IO2    1-IO2
DIV10    95-37    114-20♦
DIVIDE   56-20    56-22     56-24    71-16   93-27   94-35   107-40  108-8   113-17♦ 116-16  139-23
DRIVTB   19-32♦
DRPUUT  135-16   135-22♦
DSPDSC   98-17♦   137-28    137-65   143-13
DSPHEL   62-25    97-22♦
DSPRCT    4-16♦   62-48     143-11
DSPRPL    4-15♦   62-43
DSTCNT   21-72♦   55-47     65-37*   117-68
DT.AVL   15-10♦
DT.UNT   15-8♦
DU.DFL   13-23♦
DU.FTL   13-26♦
DU.INF   13-24♦
DU.QUE   13-22♦
DU.SPC   13-27♦
DU.TER   13-25♦
DUP       7-14♦
E$END     1-IO2♦
E$LOAD    1-IO2♦   1-I25
ECCCNT   21-73♦   55-48     65-38*   117-68
ECODE    20-21♦   25-34     88-43*   88-47   88-55   88-65
EF.BBR   10-29♦   79-63     141-4
EF.BBU   10-30♦
EF.CON    5-9♦
EF.LOG   10-31♦
EF.NEW    5-9♦    122-15
EF.PWR    5-9♦
EF.RES    5-9♦    122-10
EF.SEX   10-32♦
EF.STA    5-9♦    122-5
EMPTY    21-38♦   94-29*    94-66    94-71*
EMSTAT   20-22♦   25-36     36-13    46-45   58-6    59-6    67-42   67-45   73-33   79-69   79-81   82-40   84-34   85-33
         85-52    88-41*    88-71    139-31
```

```
ENBERR   4-18&    58-33    62-59
ENBRPL   4-17&    62-53    67-76    67-82    137-46    140-15    141-28
ENBWFE   4-19&    62-64    79-50    82-30
ENDREF   20-20&   25-33    88-40*   88-60
ENTRYS   17-6&    92-22    93-26    94-58    94-100    98-86
ERFLAG   20-29&
ERLAST   47-29&   49-9     49-11    49-14*   49-15*
ERLDS    20-37&   50-7     50-31    50-36
ERLEXI   47-47    48-8     48-21    50-42    51-9&
ERLFMT   20-33&   47-35*   47-37
ERLLBN   20-34&   49-4*    49-5*    49-9     49-11    49-14   49-15   49-17   49-17   50-4*   50-5*   50-21   50-21
ERLMSG   27-12&
ERLPSI   47-27&   47-38
ERLPTB   47-20&   47-27    47-52
ERLVAL   27-19&
ERM001   32-7&    132-33
ERM101   32-13&   129-34
ERM102   32-15&   45-32    128-68   129-117  130-42    130-52
ERM103   32-16&   128-94
ERM104   32-17&
ERM201   32-22&   58-11
ERM202   32-24&   58-40
ERM203   32-28&   59-11
ERM204   32-29&   58-21
ERM300   32-35&   37-36
ERM301   32-36&   73-38
ERM302   32-37&   79-74    82-45
ERM303   32-38&   139-38
ERM304   32-39&   79-40
ERM305   32-40&   79-84
ERM306   32-42&   88-102
ERM307   32-43&   88-78
ERM401   33-5&    88-86
ERM402   33-6&    88-94
ERM403   33-7&    88-109
ERM501   33-12&
ERM601   33-17&   84-39
ERM602   33-18&   85-38    85-57    98-35
ERM603   33-19&   84-66
ERM604   33-20&   85-79
ERM605   33-21&   70-31
ERM606   33-22&   76-36
ERM607   33-24&   52-40
ERM700   33-30&   47-45
ERM701   33-31&   48-6
ERM702   33-32&   48-18
ERM703   33-33&   50-18
ERM704   33-34&   51-6
ERR.SZ   41-36&   103-20
ERR.TB   41-27&   41-36    103-23
ERR001   36-7&    58-13    58-23    58-42    59-13    98-36    132-34
ERR101   36-23&   45-34    128-70   128-95   129-36   129-119  130-44   130-54
ERR201   37-7&
ERR300   37-28&
ERR301   37-53&   73-40    79-42    79-76    79-86    82-47    84-41   85-40   85-59   88-80   88-104
ERR302   37-76&   139-40
```

```
ERR401   39-7@    88-88    88-96    88-111
ERR601   38-28@   52-42    70-32    76-38    84-68    85-81
ERR701   39-7@    47-46    48-7     48-19    50-19    51-7
ERRBLK   19-5@
ERRC     101-39   101-46   105-35@
ERRCNT   21-28@   84-26*   84-46*   85-25*   85-45*
ERRD     101-48   105-51@
ERRFMT   20-30@
ERRL0    47-21    48-3@
ERRL1    47-22    48-11@
ERRL2    47-23    49-3@
ERRL3    47-24    50-3@
ERRL4    47-25    51-3@
ERRLOG   47-31@   88-49
ERRME1   34-18@   101-44   103-26
ERRMSG   19-5@
ERRNBR   19-5@
ERRREF   20-27@
ERRSTE   21-80@
ERRTYP   19-5@
EVENT    20-28@   27-20    39-12    47-36*   49-18
EVL      5-9@
EXIT10   77-49@
EXIT11   78-63    78-93@
EXIT12   79-64    79-82    79-94@
EXIT13   80-39@
EXIT15   81-35    81-41@
EXIT16   82-38    82-43    82-50@
EXIT17   83-36@
EXIT3    70-16    72-16    72-19@
EXIT4    73-30    73-36    73-42@
EXIT5    74-28@
EXIT6    75-49@
EXIT9    76-34    76-41@
EXTMSK   20-31@   40-14    40-19*   45-33*   47-44*   48-5*    48-17*   50-17*   51-5*    52-41*   58-12*   58-22*   58-41*   59-12*
         70-30*   73-39*   76-37*   79-41*   79-75*   79-85*   82-46*   84-40*   84-67*   85-39*   85-58*   85-80*   88-79*   88-87*
         88-95*   88-103*  88-110*  128-69*  129-35*  129-118* 130-43*  130-53*  139-39*
F$AU     1-I02@   136-9    136-15
F$AUTO   1-I02@   133-10   133-12
F$BGN    1-I02@   1-I17    4-22     5-3      36-7     36-23    37-7     37-28    37-53    37-76    38-7     38-28    39-7     42-11
         43-20    44-6     44-11    67-102   117-56   117-62   119-9    120-26   127-34   133-10   134-8    134-13   135-8    136-9
         136-16   136-17   137-3    137-7    137-11   137-72   137-78   137-82   138-3    138-7    143-28   143-32   143-34   144-2
         144-12   145-12   145-26   146-2    146-3    146-3    146-9    146-10
F$CLEA   1-I02@   134-8    134-17
F$DU     1-I02@   135-8    135-26
F$END    1-I02    1-I02    1-I02    1-I02    1-I02    1-I02    1-I02    1-I02    1-I02    1-I02    1-I02    1-I02    1-I02    1-I02
         1-I02    1-I02    1-I02@   1-I17    4-22     5-3      36-16    36-35    36-37    37-19    37-21    37-44    37-46    37-67
         37-69    37-88    37-90    38-19    38-21    38-40    38-42    39-16    39-18    42-13    43-23    44-10    44-13    67-102
         117-56   117-79   118-40   118-52   127-34   132-39   133-12   134-13   134-17   135-18   135-26   136-11   136-15   136-16
         136-17   137-3    137-3    137-3    137-7    137-11   137-72   137-78   137-82   137-82   138-3    138-3    138-3    138-7
         143-28   143-32   143-32   143-34   144-2    144-30   145-14   145-26   146-2    146-3    146-9    146-10
F$HARD   1-I02@   144-12   144-30
F$HW     1-I02@   3-10     3-17
F$INIT   1-I02@   120-26   132-39
F$JMP    1-I02@   36-35    36-35    37-19    37-19    37-44    37-44    37-67    37-67    37-88    37-88    38-19    38-19    38-40
         38-40    39-16    39-16    117-79   117-79   118-40   118-40   127-34   134-13   135-18   135-18   136-11   136-11   137-7
```

CZUDLAO BBR Replacement Utility MACRO Y05.03c Friday 19-Apr-85 10:42 Page S-9
Cross reference table (CREF V05.03)

```
            137-11    137-72    137-78    138-7     143-28
F$MOD       1-I02#    1-I17     4-22      5-3       67-102    117-56    136-16    136-17    143-34    144-2     145-26
F$MSG       1-I02#    36-7      36-16     36-23     36-37     37-7      37-21     37-28     37-46     37-53     37-69     37-76     37-90     38-7
            38-21     38-28     38-42     39-7      39-18
F$PROT      1-I02#    119-9     119-15
F$PWR       1-I02#
F$RPT       1-I02#    117-62    118-52
F$SEG       1-I02#
F$SOFT      1-I02#    145-12    145-14
F$SRV       1-I02#    42-11     42-13     43-20     43-23     44-6      44-10     44-11     44-13
F$SUB       1-I02#
F$SW        1-I02#    4-10      4-21
F$TEST      1-I02#    137-3     137-82    138-3     143-32
F.CRLF      34-5#     99-13
F.TEXT      34-6#     97-85     99-10
FEF         18-17#    70-22     75-40
FERCNT      21-70#    55-45     65-35*    117-68
FFREE       19-7#     59-32     59-36*    120-34*   120-35    123-18*   123-19    132-18    132-22*
FLAGS       20-23#    25-35     67-41     79-63     88-42*    141-4
FMERR       128-46    132-20    132-32#
FMT.AS      34-8#
FSIZE       19-8#     59-32     59-36*    120-35*   123-19*   128-44    132-19*
G$CNTO      1-I02#
G$DELM      1-I02#
G$DISP      1-I02#
G$EXCP      1-I02#
G$HILI      1-I02#
G$LOLI      1-I02#
G$NO        1-I02#
G$OFFS      1-I02#    61-20     63-28     64-27     66-33     144-21    144-23    144-25    144-27    144-29
G$OFSI      1-I02#    61-20     63-28     64-27     66-33     144-21    144-23    144-25    144-27    144-29
G$PRMA      1-I02#    144-21    144-23
G$PRMD      1-I02#    63-28     66-33     144-25    144-27    144-29
G$PRML      1-I02#    61-20     64-27
G$RADA      1-I02#    63-28     66-33
G$RADB      1-I02#
G$RADD      1-I02#    144-25    144-27    144-29
G$RADL      1-I02#    61-20     64-27
G$RADO      1-I02#    144-21    144-23
G$XFER      1-I02#
G$YES       1-I02#    61-20     63-28     64-27     66-33     144-21    144-23    144-25    144-27    144-29
GBCNT       21-87#    65-25*    65-26*    87-26     87-27     142-17*   142-18*   142-21*   142-22*   142-28*   142-29*
GETCDN      106-17    106-19    106-29#
GETCHR      62-31     63-20#
GETCNT      102-15    103-10    103-19    103-32    106-14#   107-16
GETCNX      106-16#   106-28
GETCXX      106-30    106-32#
GETLBN      66-20#    137-36    138-46    138-52
GETLOG      62-21     62-40     62-45     62-50     62-55     62-61     62-66     64-19#    123-12    137-24    137-62
GETVSN      59-56     95-21#
GLBN        21-86#    65-18*    65-19*    87-28     87-29     138-56*   138-57*   139-24*   139-25*   139-26    139-27    140-19*   140-20*   140-30*
            140-31*   141-6*    141-7*    141-15*   141-16*   141-32*   141-33*   141-43*   141-44*   142-7     142-11    142-36    142-36    143-3
            143-3
GRPSIZ      17-5#     21-88     138-58    142-15    142-30
GTDRVT      96-13#
H.BRL       125-20    144-16#   144-25    144-25    144-25
```

```
H.BST    125-25   144-17#  144-27   144-27   144-27
H.DRV    125-26   144-18#  144-29   144-29   144-29
H.UBA    144-14#  144-21   144-21   144-21
H.VEC    125-23   144-15#  144-23   144-23   144-23
HADDR     20-35#   27-21    48-12*   48-13*   48-20    48-20
HC.CCT     7-31#   90-41*
HC.CEV     7-36#    7-37    89-37
HC.CMD     7-30#    7-31     7-33    90-40*  131-23*
HC.CPK     7-37#    7-39    37-57    37-58    46-44    46-44    58-31*   58-37*   60-11*   60-12*   72-6*    72-7*    73-27*   73-28*
          74-25*   74-26*   78-43*   78-44*   78-79*   78-80*   79-30*   79-31*   79-32*   79-33*   79-57*   79-58*   81-30*   81-31*
          81-38*   81-39*   82-36*   82-37*   84-53*   84-54*   85-66*   85-67*   86-26*   86-27*   86-28*   86-29*   86-30*   87-25*
          87-26*   87-27*   87-28*   87-29*   89-42*   89-43*   89-51*   90-25*   94-47*   94-48*   98-40*   98-41*  109-14   131-22
HC.ESZ     7-22#    7-34     7-37    90-29    90-30    90-39
HC.INT     7-25#    7-27
HC.ISZ     7-20#    7-27   128-43
HC.MCT     7-28#   47-43*   48-4*    48-15*   49-6*    50-15*   51-4*    90-37*
HC.MEV     7-33#    7-34    90-26
HC.MPK     7-34#    7-36    47-35    47-36    48-12    48-13    49-4     49-5     50-4     50-5     50-6     58-17    58-18    58-26
          58-27    59-17    59-18    59-19    88-40    88-41    88-42    88-43   131-19   131-22   139-20   139-21   141-13   141-14
HC.MSG     7-27#    7-28     7-30    90-36*  129-20   131-20*
HC.PSZ     7-23#    7-36     7-39    89-38    89-41    90-30    90-39
HC.RSZ     7-21#    7-30     7-33
HC.SIZ     7-39#  131-15
HCECNT    21-71#   55-46    65-36*  117-68
HCOM      20-4#    46-43    47-34    50-14    51-10    58-16    59-16    59-54    60-4     67-33    69-26    73-23    74-22    75-24
          76-25    77-34    78-26    79-28    80-23    81-24    82-24    83-21    84-28    85-28    86-25    87-24    88-26    88-37
          89-35    90-23    98-19   109-13   117-65   125-12   128-50   128-85   129-19   131-18*  139-19   141-12
HCOMM    123-30   131-15#
HDR001    32-6#   132-34
HDR101    32-12#   45-34   128-70   128-95   129-36   129-119  130-44   130-54
HDR201    32-21#   58-13    58-23    58-42    59-13
HDR301    32-34#   73-40    79-42    79-76    79-86    82-47    84-41    85-40    85-59    88-80    88-104   139-40
HDR401    33-4#    88-88    88-96    88-111
HDR501    33-11#
HDR601    33-16#   52-42    70-32    76-38    84-68    85-81    98-36
HDR701    33-29#   47-46    48-7     48-19    50-19    51-7
HELP      1-H88#   1-H97    1-I10
HLPFIL    34-28#   97-28
HOE        5-9#
I$AU       1-I02# 136-9#   136-15#
I$AUTO     1-I02# 133-10#  133-12#
I$CLN      1-I02# 134-8#   134-13   134-17#
I$DU       1-I02# 135-8#   135-26#
I$HRD    144-12#  144-30#
I$INIT     1-I02# 120-26#  127-34   132-39#
I$MOD      1-I02#   1-I17   1-I17#   4-22     4-22#    5-3      5-3#    67-102   67-102# 117-56   117-56#  136-16   136-16#  136-17
         136-17#  143-34   143-34#  144-2    144-2#   145-26   145-26#
I$MSG      1-I02#  36-7#    36-16#   36-23#   36-37#   37-7#    37-21#   37-28#   37-46#   37-53#   37-69#   37-76#   37-90#   38-7#
          38-21#   38-28#   38-42#   39-7#    39-18#
I$PROT     1-I02# 119-9#
I$PTAB     1-I02# 146-3    146-3#   146-9    146-9#
I$PWR      1-I02#
I$RPT      1-I02# 117-62#  118-52#
I$SECT   120-24#  127-32#
I$SEG      1-I02# 137-3    138-3
I$SETU     1-I02# 146-2    146-2#   146-3    146-10   146-10#
```

```
I$SFT   145-12#  145-14#
I$SRV     1-I02#   42-11#   42-13#   43-20#   43-23#   44-6#    44-10#   44-11#   44-13#
I$SUB     1-I02#  137-3    138-3
I$TST     1-I02#  137-3    137-3#   137-7    137-11   137-72   137-78   137-82   137-82#  137-82#  138-3    138-3#   138-7    143-28
        143-32   143-32#  143-32#
IBE       5-9#
IBUFF    72-3     73-25    79-45    82-26   139-11   145-21#
IDU       5-9#
IER       5-9#
INCARN   18-9#
INCRLB   21-82#
INF100   30-5#   123-8
INF101   30-9#   121-12
INF102   30-11#   62-17
INF103   30-14#  117-78
INF110   30-18#   59-58
INF111   30-23#   59-62
INF200   30-29#  117-67
INF201   30-30#  117-68
INF202   30-36#  117-73
INF210   30-43#   98-21
INF211   30-44#   98-67
INF212   3C-45#   98-75
INF213   30-46#   98-82
INF220   30-48#   98-56
INF221   30-49#   98-57
INF301   31-4#    60-8
INF302   31-5#   138-32
INF310   31-6#   137-13
INF311   31-7#   137-56
INF320   31-8#   138-9
INF321   31-9#   138-10
INF322   31-16#  138-11
INF323   31-20#
INF324   31-21#
INF401   31-25#   46-44
INF405   31-27#   49-17    53-32    53-45    53-57    54-7     54-17   142-36   143-3
INF406   31-28#   53-33    53-46    53-58    54-8     54-18
INF407   31-29#   53-34    55-12    55-22
INF408   31-30#   53-35    55-13    55-23
INF601   31-34#   61-24
INF602   31-35#
INF603   31-36#
INIPAR   61-28    65-15#
INISWP   62-16#  124-9
INITAC   87-23#  139-12
INITBL  129-43   129-64#
INITRW   59-51    60-10    69-30    72-5     73-26    74-24    75-27    75-43    77-47    78-42    78-78    79-56    80-36    81-29
         81-37    82-35    83-34    86-24#   94-46    98-39
INTCPK   58-3     59-48    60-5     69-27    76-26    79-29    80-24    81-25    82-25    83-22    89-30#   98-23
INTRCV   20-7#    44-12*
INTSRV   44-11#
INVCHR   34-16#  117-45
IPVAL    19-42#
ISR       5-9#
ISSUE    58-4     58-38    59-4     73-30    79-38    79-59    82-38    84-32    85-31    85-50    88-24#  139-15
```

```
IXE       5-9#
J$JMP     1-I02#   36-35    37-19    37-44    37-67    37-88    38-19    38-40    39-16    117-79   118-40   135-18   136-11
KTMEM     19-22#
KW.BRL    19-16#   121-16*
KW.CSR    19-15#   44-9*    56-14    91-37    121-11*  121-15*  121-20*  130-34
KW.EL     19-19#   44-7*    44-8*    56-17    56-18    65-28*   65-29*   91-39    91-42    111-26   111-27   111-28   121-5*   121-6*
          130-36   130-39
KW.HZ     19-18#   56-19    111-16   121-18*
KW.OUT    44-9     121-4    121-20
KW.VEC    19-17#   121-17*  121-19
KW11I     44-6#    121-19
L$ACP     1-I25#
L$APT     1-I25#
L$AU     136-9#
L$AUT     1-I25#
L$AUTO    1-I25    133-10#
L$CCP     1-I25#
L$CLEA    1-I25    134-8#
L$CO      1-I25#
L$DEPO    1-I25#
L$DESC    1-I25    29-25#
L$DESP    1-I25#
L$DEVP    1-I25#
L$DISP    1-I25    2-8#
L$DLY     1-I25#
L$DTP     1-I25#
L$DTYP    1-I25#
L$DU      1-I25    135-8#
L$DUT     1-I25#
L$DVTY    1-I25    29-20#
L$EF      1-I25#
L$ENVI    1-I25#
L$ERRT    1-I25    19-5#
L$ETP     1-I25#
L$EXP1    1-I25#
L$EXP4    1-I25#
L$EXP5    1-I25#
L$HARD    1-I25    144-12   144-12#
L$HIME    1-I25#
L$HPCP    1-I25#
L$HPTP    1-I25#
L$HW      1-I25    3-10     3-10#
L$ICP     1-I25#
L$INIT    1-I25    120-26#
L$LADP    1-I25#
L$LAST    1-I25    145-25#  146-10
L$LOAD    1-I25#
L$LUN     1-I25#   59-58    98-21    124-15*  124-18*  124-21   124-24   135-13   135-16   137-15*  137-71   137-77   138-13*  138-16*
          138-17   143-23
L$MREV    1-I25#
L$NAME    1-I25#
L$PRIO    1-I25#
L$PROT    1-I25    119-9#
L$PRT     1-I25#
L$REPP    1-I25#
L$REV     1-I25#
```

```
L$RPT     1-I25   117-62#
L$SOFT  145-12    145-12#
L$SPC     1-I25#
L$SPCP    1-I25#
L$SPTP    1-I25#
L$STA     1-I25#
L$SW      1-I25     4-10      4-10#
L$TEST    1-I25#
L$TIML    1-I25#
L$UNIT    1-I25#  124-13*   124-21    138-17
L.BADR   12-20#    48-12     48-13
L.CRF    12-16#
L.DS     12-23#    50-6
L.EVNT   12-19#    47-36
L.FLGS   12-18#
L.FMT    12-17#    47-35
L.HDCD   12-21#    49-4      49-5      50-4      50-5
L.SDI    12-22#    12-23
L10000    3-10     3-17#
L10001    4-10     4-21#
L10002   36-16#
L10003   36-35    36-37#
L10004   37-19    37-21#
L10005   37-44    37-46#
L10006   37-67    37-69#
L10007   37-88    37-90#
L10010   38-19    38-21#
L10011   38-40    38-42#
L10012   39-16    39-18#
L10013   42-13#
L10014   43-23#
L10015   44-10#
L10016   44-13#
L10017  117-79   118-40    118-52#
L10021  127-34   132-39#
L10022  133-12#
L10023  134-13   134-17#
L10024  135-18   135-26#
L10025  136-11   136-15#
L10026  137-7    137-11    137-72    137-78    137-82#
L10027  138-7    143-28    143-32#
L10030  144-12   144-30#
L10031  145-12   145-14#
L10032  146-3#
L10034  146-3    146-9#
LBN      21-18#   37-36     37-36     37-80     37-80     52-28     52-32     52-56     52-57     53-32     53-32     53-45     53-45     53-57
         53-57    54-7      54-7      54-17     54-17     70-20*    70-21*    71-13     71-14     73-27     7?-28     75-35     75-36     78-52
         78-53    79-32     79-33     79-57     79-58     82-36     82-37     94-32     94-33     94-75     94-77    137-35    139-5*    139-6*
        139-26*  139-27*   141-13*   141-14*   141-15    141-16
LBNDX    21-66#
LBNLO    18-5#    70-20     70-21     75-35*    75-36*    80-30     83-28
LBNSTA   21-64#   53-39     53-60*    54-4      54-14     65-42*    67-37*    67-41*    67-46*    67-50*    67-55*    67-60*    67-65*    67-70
         67-87    67-94     70-18*    70-25*    72-11*    75-38     79-48     79-54*    79-79     82-28    139-7*    140-12    140-28    141-41
LF       29-16    97-31     97-52
LINBUF   22-12#   97-29     97-84     97-86
LMT28    34-15#  117-42
```

```
LOADFL  19-9#    120-30   120-32*
LOE     5-9#
LOG     7-12#
LOGMSG  64-20*   64-23#   64-27
LOGRPL  64-21*   64-24#   64-27    64-29
LOT     5-9#
LPNT    56-25    56-29    56-34    100-7    100-10   100-13   100-17#
LPNTB   36-12    36-13    36-27    37-11    37-36    37-59    37-80    38-11    38-32    39-11    39-12    40-25    46-44    46-45
        53-32    53-33    53-34    53-35    53-45    53-46    53-57    53-58    54-7     54-8     54-17    54-18    55-12    55-13
        55-22    55-23    59-58    59-62    60-8     61-24    62-17    66-24    98-21    98-56    98-57    98-67    98-75    98-82
        100-9#   117-67   117-68   117-73   117-78   118-7    118-15   118-24   118-33   138-10   138-11   138-32   142-36   143-3
LPNTF   66-49    100-6#   101-44   103-26   117-42   117-45   121-12   123-8    135-13   135-16   137-13   137-56   138-9
LPNTS   100-15#
LPNTX   40-17    48-20    49-17    49-18    50-21    50-32    50-34    50-39    100-12#
M.RA60  24-5#    24-15    50-27
M.RA80  24-4#    24-14
M.RA81  24-6#    24-16
M.RA82  24-7#    24-17
M.UNKN  24-3#    24-13
MANDIS  34-19#
MANRPL  4-13#    61-12    62-27*   62-34*   124-11   137-5    138-5
MATBLK  21-32#   78-72*   78-79    81-38
MATFLG  21-40#   53-26    53-36*   71-4*    71-7*    77-39    78-61    94-28*   94-79*
MATOFF  21-34#   78-69*
MATRBN  21-21#   53-34    53-34    71-8*    71-9*    77-42    77-43    78-65    78-65    94-81*   94-82*
MAXLBN  21-91#   65-22*   65-23*   138-48*  138-49*  138-51   142-4    142-5
MD.CMP  10-5#    79-46    84-30    85-30
MD.CWB  10-23#
MD.ERR  10-7#    79-53    82-33    85-49
MD.EXP  10-6#
MD.FEU  10-17#
MD.IMF  10-21#
MD.NXU  10-19#
MD.PRI  10-24#   79-36
MD.RIP  10-20#
MD.SCH  10-8#
MD.SCL  10-9#
MD.SEC  10-10#   67-72
MD.SEQ  10-15#
MD.SER  10-11#
MD.SPD  10-16#
MD.SSH  10-12#
MD.SWP  10-22#
MD.VOL  10-18#
MD.WBN  10-13#
MD.WBV  10-14#
MDLNDX  24-10#   50-23    59-28*   59-61
MDLSIZ  24-19#   59-21
MDLTBL  24-12#   24-19    59-23
MDLTEX  24-21#   59-62
MINLBN  21-90#   65-20*   65-21*   138-42*  138-43*  138-45   138-56   138-57
MODEL   24-9#    50-27    59-19*   59-23
MSCP    7-11#    89-39
MSCPMS  25-17#   37-13    37-38    37-61    37-82
MSCPND  28-7#    37-32
MSCPTB  28-10#   37-36
```

```
MSCPVA  25-29#   37-14    37-39    37-62    37-83
MSGADR  66-22*   66-28#   66-33
MSGBRL  144-25   144-37#
MSGBST  144-27   144-38#
MSGLDR  144-29   144-39#
MSGPCK  20-6#    131-21*
MSGUBA  144-21   144-35#
MSGVEC  144-23   144-36#
MULT    71-22    92-22    94-39    115-15#
MULTRD  59-52    60-13    69-31    72-8     75-28    78-45    78-81    84-23#   94-49    98-42
MULTWR  74-27    75-44    77-48    78-59    78-91    80-37    81-32    81-40    83-35    85-23#
NCONF   101-41   101-46#
NCONS   101-40#  101-43
NDX.AC  28-4#
NDX.RD  28-2#
NDX.RP  28-5#
NDX.WR  28-3#
NODOCU  1-21
NXMAD   19-21#   42-12*   45-15*   45-29    129-26*  129-31
NXMI    42-11#   45-18    129-27
NXTSTE  21-79#   61-9     68-63    68-71    68-78    68-83*   68-85    68-90*   70-15*   72-15*   72-18*   73-43*   74-29*   75-50*
        76-30*   76-39*   76-42*   77-50*   78-94*   79-89*   79-92*   79-95*   80-40*   81-42*   82-51*   83-37*   137-51*  140-25*
        141-38*
O$APTS  1-I02#   1-I25
O$AU    1-I02#   1-I25
O$BGNR  1-I02#   1-I23#   1-I25
O$BGNS  1-I02#   1-I25
O$DU    1-I02#   1-I23#   1-I25
O$ERRT  1-I02#   1-I23#   1-I25
O$GNSW  1-I02#   1-I23#   1-I25
O$POIN  1-I02#   1-I23    1-I23#   1-I23#   1-I23#   1-I23#   1-I23#   1-I25
O$SETU  1-I02#   1-I23#   1-I25    145-25
OLIMIT  34-17#   66-49
ONEFIL  1-H92#
ONLINE  21-93#
OP.ABO  9-3#
OP.ACC  9-4#     139-15
OP.AVA  9-22#
OP.AVL  9-5#
OP.CCD  9-6#
OP.CMP  9-7#
OP.DUP  9-23#
OP.ELP  9-29#
OP.END  9-20#    88-47    88-55    88-64
OP.ERS  9-8#
OP.ESP  9-28#
OP.FLU  9-9#
OP.GCS  9-10#
OP.GSS  9-27#
OP.GUS  9-11#    58-4
OP.MRD  9-18#
OP.MWR  9-19#
OP.ONL  9-12#    59-4
OP.RD   9-13#    73-30    84-32
OP.RLC  9-25#
OP.RPL  9-14#    79-38
```

```
OP.RSD    9-31◊
OP.SCC    9-15◊    58-38
OP.SEX    9-21◊    88-55
OP.SHC    9-24◊
OP.SSD    9-30◊
OP.SUC    9-16◊
OP.WR     9-17◊    79-59    82-38    85-31    85-50
OPCODE   20-15◊    25-31    88-25*   88-63    89-42
OPHELP    4-12◊    62-19    62-22
OPTIN    22-10◊
OSTRE   101-38    101-45   101-50◊
OSTRNG  100-23    101-37◊  101-49
P.BCNT   11-9◊     12-10◊   86-27*   86-28*   87-26*   87-27*   139-20   139-21
P.BUFF   11-10◊
P.CLAS   12-56◊
P.CMST   12-28◊
P.CNCL   12-69◊
P.CNTF   11-37◊    12-67◊   58-37*
P.CNTI   12-70◊
P.CPSP   11-26◊
P.CRF    11-5◊     12-5◊    88-40    90-25*
P.CTMO   12-68◊
P.CYLS   12-42◊
P.DEXT   12-75◊
P.DFLG   12-76◊
P.DMDT   11-51◊
P.DPRG   12-77◊
P.DTMO   12-78◊
P.ELGF   11-24◊
P.FBBK   12-11◊    141-13   141-14
P.FLGS   12-8◊     88-42
P.GRPS   12-41◊
P.HSTI   11-23◊    12-35◊   12-53◊
P.HTMO   11-38◊    58-31*
P.LBN    11-12◊    37-57    37-58    46-44    46-44    60-11*   60-12*   72-6*    72-7*    73-27*   73-28*   74-25*   74-26*   78-43*
         78-44*    78-79*   78-80*   79-32*   79-33*   79-57*   79-58*   81-30*   81-31*   81-38*   81-39*   82-36*   82-37*   84-53*
         84-54*    85-66*   85-67*   86-29*   86-30*   87-28*   87-29*   94-47*   94-48*   98-40*   98-41*
P.MEDI   12-37◊    12-57◊
P.MLUN   12-33◊    12-51◊
P.MOD    11-8◊     89-43*
P.MODE   12-55◊    59-19
P.OPCD   11-7◊     12-7◊    88-43    89-42*
P.OTRF   11-17◊    12-27◊
P.OVRL   11-52◊
P.RBN    11-31◊    79-30*   79-31*
P.RBNS   12-44◊    58-26
P.RCTC   12-45◊    58-27
P.RCTS   12-43◊    58-18
P.RGID   11-45◊
P.RGOF   11-46◊
P.SHST   12-39◊    12-59◊
P.SHUN   11-25◊    12-38◊   12-58◊
P.STS    12-9◊     88-41
P.TIME   11-40◊
P.TRKS   12-40◊    58-17
P.UADR   11-11◊    86-26*   87-25*   109-14
```

```
P.UNFL  11-22#    12-34#    12-52#
P.UNIT  11-6#     12-6#     89-51*
P.UNSZ  12-60#    59-17     59-18
P.UNTI  12-36#    12-54#    12-55     12-56
P.USEF  11-39#
P.VRSN  11-36#    12-66#
P.VSER  12-61#
P1      18-20#    70-5      72-13     75-37     77-45
P2      18-19#    70-10     77-38
PB      99-20#    100-9
PF      99-18#    100-6     100-28
PHASE   21-84#
PNT      5-9#
PNTNUM 102-26    102-32    103-5     107-15#
PNTNUS 107-17#
PRI      5-9#
PRI00    5-9#    127-30
PRI01    5-9#
PRI02    5-9#
PRI03    5-9#
PRI04    5-9#
PRI05    5-9#
PRI06    5-9#
PRI07    1-I25     5-9#     45-18    121-19    129-27
PRIMRY  21-12#    55-11*    65-31*    98-24*    98-53     98-57     98-65*  117-70   117-73
PRINTC  56-28     56-33     56-36     99-8#    102-8    102-19    103-11   103-33   108-16   108-28   108-38   134-10
PS      99-24#   100-15
PTABLE  19-30#   124-24*   125-18
PTYPE   99-15    100-6*    100-9*    100-12*   100-15*  100-28#
PX      99-22#   100-12
RA60    24-24     24-31#
RA80    24-23     24-30#
RA81    24-25     24-32#
RA82    24-26     24-33#
RBN     21-19#    55-12     55-12     55-22     55-22     70-27*   70-28*   71-23    71-28    77-36    77-37    78-29    78-29    79-30
        79-31     92-20*    92-21*    92-22     92-22     92-22*   92-22*   92-23*   92-24*   94-81    94-82    98-67    98-67    98-75
        98-75     98-82     98-82
RBNLO   18-6#     70-27     70-28     77-36*    77-37*
RBNS    21-6#     58-26*    71-19     94-36     98-57
RCTBLK  21-31#    26-22     46-37     46-44     59-50*   60-7*    60-11    60-15*   60-16    69-29*   72-4*    72-6     74-23*   74-25
        75-26*    75-42*    77-46*    78-36*    78-43    80-35*   81-30    83-33*   92-20    94-42*   94-44*   94-47    94-93*   94-103*
        98-28*    98-31*    98-32     98-40
RCTCOP  21-8#     26-23     46-28     46-44     84-25*   84-50*   84-51    85-27*   85-63*   85-64
RCTERR  46-24#    84-47     85-46
RCTLBN  21-20#
RCTMAS  21-9#     46-29*    46-33*    46-40     46-47
RCTMSG  26-13#    38-34
RCTOFF  21-33#    78-33*    92-23     94-56*    98-30*   98-85*   98-86
RCTSIZ  21-4#     58-18*    59-33     59-40     60-16    84-53    85-66    98-32   118-19
RCTTBL  19-34#    46-36     59-35*   118-7    118-10
RCTVAL  26-21#    38-35
RDCNT   21-83#    67-34*    67-67*    67-90*
RDSTAT  21-67#
RDVALD  21-24#
READLB  67-32#   137-39    140-4    141-20
RECOVR  21-22#    26-24     65-30*    70-4*    70-7*    70-12*
```

```
REPEAT   21-92@
REPLAC   61-25     68-60@   137-52   140-26   141-39
REPLY    22-6@     61-19*   61-20    61-21    62-29    62-32    137-22   137-25   137-60
RESCAN   21-39@    94-30*   94-86    94-91*
RESET    45-13@    134-11
RFLAGS   18-4@     70-5     70-10    70-22    71-5     72-13    75-37*   75-40*   77-38*   77-41*   77-45*   80-27    83-25
RG.FLG   7-6@      47-43    48-4     48-15    49-6     50-15    51-4     90-37
RG.OWN   7-5@      47-43    48-4     48-15    49-6     50-15    51-4     90-37    90-41          .
RNTIM    34-12@    56-25
RNTIM1   34-13@    56-29
RNTIM2   34-14@    56-34
RNTIME   56-14@    142-37   143-4
RNTIMX   56-15     56-36@
RPLSTA   21-65@    55-3     55-25*   65-43*   71-3*    80-41*   139-8*
RSP.CK   129-78    129-87   129-95   129-111@
RSP.S1   129-64    129-74@
RSP.S2   129-66    129-82@
RSP.S3   129-68    129-91@
RSP.S4   129-70    129-99@
RTFLAG   19-10@
S$LSYM   1-I02@    3-17@    4-21@    36-16@   36-37@   37-21@   37-46@   37-69@   37-90@   38-21@   38-42@   39-18@   42-13@   43-23@
         44-10@    44-13@   61-20    61-20    61-20    61-20@   63-28    63-28    63-28    63-28@   64-27    64-27    64-27    64-27@
         66-33     66-33    66-33    66-33@   118-52@  132-39@  133-12@  134-17@  135-26@  136-15@  137-82@  143-32@  144-30@  145-14@
SA.BST   6-73@
SA.CM1   6-28@
SA.CMD   6-23@
SA.CME   6-49@
SA.CNT   6-79@     129-101  129-102
SA.CTP   6-52@
SA.DI    6-35@     129-75
SA.ERC   6-15@
SA.ERR   6-6@      130-21
SA.GO    6-71@     128-103
SA.INE   6-64@
SA.INT   6-21@
SA.LFC   6-72@
SA.MCV   6-78@     129-101
SA.MP    6-36@     129-76
SA.MS1   6-27@
SA.MSE   6-48@
SA.MSG   6-22@
SA.NV    6-33@
SA.NVE   6-65@
SA.PRG   6-42@
SA.QB    6-34@     129-76
SA.S1    6-10@     129-42   129-75
SA.S2    6-9@      129-86
SA.S3    6-8@      129-94
SA.S4    6-7@      129-53   129-101
SA.STE   6-51@
SA.STP   6-25@     129-17
SA.TST   6-58@     129-21
SA.VCE   6-63@
SA.VEC   6-20@
SA.WRP   6-24@
SADDR    19-38@    23-20    57-33*
```

```
SAEXP    19-40#   23-22    129-115* 130-20*
SARMSG   23-11#   36-29    38-13
SARVAL   23-19#   36-30    38-14
SAVAL    19-39#   23-21    91-30*   128-67*  129-116* 130-29*
SB.DST   13-32#   67-58
SB.ECC   13-33#   67-63
SB.HCE   13-31#   67-53
SCNDRY   21-13#   55-21*   65-32*   98-25*   98-54    98-57    98-72*   117-71   117-73
SEARCH   76-28    94-27#
SETTO    91-24    111-14#  130-25
SFLAG    21-44#   55-6     55-16    71-11*   71-31*   76-33    78-55    79-34    80-44*   94-66*   94-88*   139-9*
SFPTBL    4-10#
SND.S1  129-18*  129-65#  129-83   129-92
SND.S2  129-19*  129-20*  129-67#
SND.S3  129-21*  129-69#
SNDCMD   88-29    90-20#
ST.ABO   13-7#
ST.AOL   13-18#   59-8
ST.AVL   13-9#    58-8
ST.CMD   13-6#
ST.CMP   13-12#
ST.CNT   13-15#
ST.DAT   13-13#   67-48    73-35    79-71    79-81    82-42    84-36    85-35    85-54    139-33
ST.DIA   13-17#
ST.DRV   13-16#
ST.HST   13-14#
ST.MFE   13-10#
ST.MSK   13-3#    58-7     59-7     73-34    79-70    82-41    84-35    85-34    85-53    139-32
ST.OFL   13-8#
ST.SUB   13-4#
ST.SUC   13-5#
ST.WPR   13-11#
STATBL   53-58    53-64    55-37#
STATUS   21-16#   49-22*   51-16*   51-21*   52-45*   52-53*   52-60*   61-33*   61-39*   66-56*   67-94*   67-95*   67-99*   68-85*
         84-42*   84-60*   84-63    84-69*   84-75    85-41*   85-60*   85-73*   85-76    85-82*   85-88    88-71*   88-72    88-77*
         88-85*   88-93*   88-101*  88-108*  88-116   135-10   137-70*  137-76*  143-20*
STEP     21-78#   68-64*
STEP10   68-49    77-29#
STEP11   68-50    78-25#
STEP12   68-51    79-27#
STEP13   68-52    80-22#
STEP15   68-54    81-23#
STEP16   68-55    82-23#
STEP17   68-56    83-20#
STEP3    61-6     68-42    69-25#
STEP4    67-39    67-74    68-43    73-22#
STEP5    67-78    67-84    68-44    74-21#
STEP6    68-45    75-23#
STEP9    68-48    76-24#
STEPER   68-66    68-68    68-73    68-82#
STEPFT   68-39    68-40    68-41    68-46    68-47    68-53    68-89#   69-31    72-8     73-30    73-41    74-27    75-28    75-44
         77-48    78-45    78-59    78-81    78-91    79-38    79-43    79-59    79-77    80-37    81-32    81-40    82-38    82-48
         83-35
STEPSI   68-58#   68-67
STEPTB   68-38#   68-58    68-70
STROFF   21-36#   94-42*   94-54    94-92*   94-102*
```

```
SVCGBL   1-I02#   1-I07#   1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    2-8      2-8      2-8      3-10     3-10     3-10     3-10     3-10     3-10     4-10
         4-10     4-10     4-10     4-10     4-10     19-5     19-5     19-5     29-20    29-20    29-20    29-25    29-25    29-25
         36-7     36-7     36-7     36-23    36-23    36-23    37-7     37-7     37-7     37-28    37-28    37-28    37-53    37-53
         37-53    37-76    37-76    37-76    38-7     38-7     38-7     38-28    38-28    38-28    39-7     39-7     39-7     42-11
         42-11    42-11    43-20    43-20    43-20    44-6     44-6     44-6     44-11    44-11    44-11    117-62   117-62   117-62
         119-9    119-9    119-9    120-26   120-26   120-26   133-10   133-10   133-10   134-8    134-8    134-8    135-8    135-8
         135-8    136-9    136-9    136-9    144-12   144-12   144-12   145-12   145-12   145-12   145-25   145-25   145-25   145-25#
SVCINS   1-I02#   1-I04#   1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    2-8      2-8
         2-8      2-8      2-8      2-8      3-10     3-10     4-10     4-10     29-20    29-20    29-20    29-20    29-25    29-25
         29-25    29-25    36-16    36-16    36-35    36-35    36-35    36-35    36-37    36-37    37-19    37-19    37-19    37-19
         37-21    37-21    37-44    37-44    37-44    37-44    37-46    37-46    37-67    37-67    37-67    37-67    37-69    37-69
         37-88    37-88    37-88    37-88    37-90    37-90    38-19    38-19    38-19    38-19    38-21    38-21    38-40    38-40
         38-40    38-42    38-42    38-42    39-16    39-16    39-16    39-16    39-18    39-18    42-13    42-13    43-23    43-23
         44-10    44-10    44-13    44-13    45-18    45-18    45-18    45-18    45-18    45-18    45-18    45-18    45-18    45-18
         45-18    45-18    45-20    45-20    45-34    45-34    45-34    45-34    45-34    45-34    45-34    45-34    45-37    45-37
         45-37    45-37    47-46    47-46    47-46    47-46    47-46    47-46    47-46    47-46    48-7     48-7     48-7     48-7
         48-7     48-7     48-7     48-7     48-19    48-19    48-19    48-19    48-19    48-19    50-19    50-19    50-19    50-19
         50-19    50-19    50-19    50-19    50-19    50-19    51-7     51-7     51-7     51-7     51-7     51-7     51-7     51-7
         52-42    52-42    52-42    52-42    52-42    52-42    52-42    52-42    57-44    57-44    57-44    57-44    57-44    57-44
         57-44    57-44    57-44    57-44    57-44    57-44    58-13    58-13    58-13    58-13    58-13    58-13    58-13    58-13
         58-23    58-23    58-23    58-23    58-23    58-23    58-23    58-23    58-42    58-42    58-42    58-42    58-42    58-42
         58-42    58-42    59-13    59-13    59-13    59-13    59-13    59-13    59-13    59-13    61-20    61-20    61-20    61-20
         61-20    61-20    61-20    61-20    61-20    61-20    61-20    61-20    63-28    63-28    63-28    63-28    63-28    63-28
         63-28    63-28    63-28    63-28    63-28    63-28    63-28    63-28    64-27    64-27    64-27    64-27    64-27    64-27
         64-27    64-27    64-27    64-27    64-27    64-27    64-27    64-27    66-33    66-33    66-33    66-33    66-33    66-33
         66-33    66-33    66-33    66-33    66-33    66-33    66-33    66-33    70-32    70-32    70-32    70-32    70-32    70-32
         70-32    70-32    70-32    70-32    73-40    73-40    73-40    73-40    73-40    73-40    73-40    73-40    76-38    76-38
         76-38    76-38    76-38    76-38    76-38    76-38    79-42    79-42    79-42    79-42    79-42    79-42    79-42    79-42
         79-76    79-76    79-76    79-76    79-76    79-76    79-76    79-76    79-86    79-86    79-86    79-86    79-86    79-86
         79-86    79-86    82-47    82-47    82-47    82-47    82-47    82-47    82-47    82-47    84-41    84-41    84-41    84-41
         84-41    84-41    84-41    84-41    84-68    84-68    84-68    84-68    84-68    84-68    84-68    84-68    85-40    85-40
         85-40    85-40    85-40    85-40    85-40    85-40    85-59    85-59    85-59    85-59    85-59    85-59    85-59    85-59
         85-81    85-81    85-81    85-81    85-81    85-81    85-81    85-81    88-80    88-80    88-80    88-80    88-80    88-80
         88-80    88-80    88-88    88-88    88-88    88-88    88-88    88-88    88-88    88-88    88-96    88-96    88-96    88-96
         88-96    88-96    88-96    88-96    88-104   88-104   88-104   88-104   88-104   88-104   88-104   88-104   88-111   88-111
         88-111   88-111   88-111   88-111   88-111   88-111   91-35    91-35    97-28    97-28    97-28    97-28    97-37    97-37
         97-37    97-37    97-38    97-38    97-58    97-58    97-85    97-85    97-85    97-85    97-85    97-85    97-85    97-85
         97-85    97-85    97-85    97-85    98-36    98-36    98-36    98-36    98-36    98-36    98-36    98-36    99-18    99-18
         99-18    99-18    99-18    99-18    99-18    99-18    99-18    99-18    99-18    99-18    99-20    99-20    99-20    99-20
         99-20    99-20    99-20    99-20    99-20    99-20    99-20    99-20    99-22    99-22    99-22    99-22    99-22    99-22
         99-22    99-22    99-22    99-22    99-22    99-22    99-24    99-24    99-24    99-24    99-24    99-24    99-24    99-24
         99-24    99-24    99-24    99-24    117-79   117-79   117-79   117-79   118-40   118-40   118-40   118-40   118-52   118-52
```

```
        120-28  120-28  120-34  120-34  120-34  120-34  120-37  120-37  120-38  120-38  121-7   121-7   121-7   121-7
        121-8   121-8   121-9   121-9   121-9   121-9   121-10  121-10  121-19  121-19  121-19  121-19  121-19  121-19
        121-19  121-19  121-19  121-19  121-19  121-19  122-5   122-5   122-5   122-5   122-6   122-6   122-10  122-10
        122-10  122-10  122-11  122-11  122-15  122-15  122-15  122-15  122-16  122-16  123-16  123-16  123-18  123-18
        123-18  123-18  124-7   124-7   124-8   124-8   124-24  124-24  124-24  124-24  124-24  124-24  124-25  124-25
        127-28  127-28  127-30  127-30  127-30  127-30  127-34  127-34  127-34  127-34  128-70  128-70  128-70  128-70
        128-70  128-70  128-70  128-70  128-95  128-95  128-95  128-95  128-95  128-95  128-95  128-95  129-15  129-15
        129-27  129-27  129-27  129-27  129-27  129-27  129-27  129-27  129-27  129-27  129-27  129-27  129-30  129-30
        129-30  129-30  129-36  129-36  129-36  129-36  129-36  129-36  129-36  129-36  129-119 129-119 129-119 129-119
        129-119 129-119 129-119 129-119 130-33  130-33  130-44  130-44  130-44  130-44  130-44  130-44  130-44  130-44
        130-54  130-54  130-54  130-54  130-54  130-54  130-54  130-54  132-34  132-34  132-34  132-34  132-34  132-34
        132-34  132-34  132-35  132-35  132-39  132-39  133-12  133-12  134-13  134-13  134-13  134-13  134-17  134-17
        135-18  135-18  135-18  135-18  135-26  135-26  136-11  136-11  136-11  136-11  136-15  136-15  137-7   137-7
        137-7   137-7   137-9   137-9   137-10  137-10  137-11  137-11  137-11  137-11  137-57  137-57  137-71  137-71
        137-71  137-71  137-72  137-72  137-72  137-72  137-77  137-77  137-77  137-77  137-78  137-78  137-78  137-78
        137-82  137-82  138-7   138-7   138-7   138-7   138-34  138-34  138-35  138-35  139-4   139-4   139-40  139-40
        139-40  139-40  139-40  139-40  139-40  139-40  143-6   143-6   143-23  143-23  143-23  143-23  143-28  143-28
        143-28  143-28  143-32  143-32  144-12  144-12  144-21  144-21  144-21  144-21  144-21  144-21  144-21  144-21
        144-23  144-23  144-23  144-23  144-23  144-23  144-23  144-23  144-25  144-25  144-25  144-25  144-25  144-25
        144-25  144-25  144-25  144-25  144-27  144-27  144-27  144-27  144-27  144-27  144-27  144-27  144-27  144-27
        144-29  144-29  144-29  144-29  144-29  144-29  144-29  144-29  144-29  144-29  144-30  144-30  145-12  145-12
        145-14  145-14  145-25  145-25  145-25  145-25  145-25  145-25  146-3   146-3   146-3   146-3

SVCSUB  1-I02#  1-I06#
SVCTAG  1-I02#  1-I08#  3-17    3-17    3-17    4-21    4-21    4-21    36-16   36-16   36-16   36-37   36-37   36-37
        37-21   37-21   37-21   37-46   37-46   37-46   37-69   37-69   37-69   37-90   37-90   37-90   38-21   38-21
        38-21   38-42   38-42   38-42   39-18   39-18   39-18   42-13   42-13   42-13   43-23   43-23   44-10   44-10
        44-10   44-13   44-13   44-13   61-20   61-20   61-20   63-28   63-28   63-28   64-27   64-27   64-27
        66-33   66-33   66-33   118-52  118-52  118-52  132-39  132-39  132-39  133-12  133-12  133-12  134-17  134-17
        134-17  135-26  135-26  135-26  136-15  136-15  136-15  137-82  137-82  137-82  143-32  143-32  143-32  144-30
        144-30  144-30  145-14  145-14  145-14  146-3   146-3   146-3   146-9   146-9   146-9
SVCTST  1-I02#  1-I05#  137-3   137-3   137-3   138-3   138-3   138-3
SWMSG0  62-18   62-72#
SWMSG1  62-28   62-73#
SWMSG2  62-37   62-74#
SWMSG3  62-42   62-75#
SWMSG4  62-47   62-76#  137-21  137-59
SWMSG5  62-52   62-77#
SWMSG6  62-58   62-78#
SWMSG7  62-63   62-79#
T$$AU   136-9#  136-11  136-15
T$$AUT  133-10# 133-12
T$$CLE  134-8#  134-13  134-17
T$$DAT  146-3   146-3#  146-9
T$$DU   135-8#  135-18  135-26
T$$HAR  144-12  144-12# 144-30
T$$HW   3-10    3-10#   3-17
T$$INI  120-26# 127-34  132-39
T$$MSG  36-7#   36-16   36-23#  36-35   36-37   37-7#   37-19   37-21   37-28#  37-44   37-46   37-53#  37-67   37-69
        37-76#  37-88   37-90   38-7#   38-19   38-21   38-28#  38-40   38-42   39-7#   39-16   39-18
T$$PC   146-2#  146-10
T$$PRO  119-9#
T$$PTA  146-2#  146-3   146-3#
T$$RPT  117-62# 117-79  118-40  118-52
T$$SOF  145-12  145-12# 145-14
T$$SRV  42-11#  42-13   43-20#  43-23   44-6#   44-10   44-11#  44-13
T$$SW   4-10    4-10#   4-21
```

CZUDLAO BBR Replacement Utility MACRO Y05.03c Friday 19-Apr-85 10:42 Page S-22
Cross reference table (CREF V05.03)

```
T$$TES 137-3#    137-7     137-11    137-72    137-78    137-82    138-3#    138-7     143-28    143-32
T$ARGC   1-I25     1-I25     1-I25     1-I25     1-I25     1-I25     1-I25     1-I25     1-I25     1-I25#    1-I25#    1-I25#    1-I25#    1-I25#
        97-85     97-85     97-85     97-85#    97-85#    99-18     99-18     99-18     99-18#    99-18#    99-20     99-20     99-20     99-20#
        99-20#    99-22     99-22     99-22     99-22#    99-22#    99-24     99-24     99-24     99-24#    99-24#
T$CODE  61-20     61-20     61-20     61-20#    61-20#    61-20#    63-28     63-28     63-28     63-28#    63-28#    63-28#    64-27     64-27
        64-27     64-27#    64-27#    64-27#    66-33     66-33     66-33     66-33#    66-33#    66-33#    144-21    144-21    144-21    144-21#
        144-21#   144-21#   144-23    144-23    144-23    144-23#   144-23#   144-23#   144-25    144-25    144-25    144-25#   144-25#   144-25#
        144-27    144-27    144-27    144-27#   144-27#   144-27#   144-29    144-29    144-29    144-29#   144-29#   144-29#
T$ERRN   1-I02#   45-34     45-34#    47-46     47-46#    48-7      48-7#     48-19     48-19#    50-19     50-19#    51-7      51-7#     52-42
        52-42#    58-13     58-13#    58-23     58-23#    58-42     58-42#    59-13     59-13#    70-32     70-32#    73-40     73-40#    76-38
        76-38#    79-42     79-42#    79-76     79-86     79-86#    82-47     82-47#    84-41     84-41#    84-68     84-68#    85-40
        85-40#    85-59     85-59#    85-81     85-81#    88-80     88-80#    88-88     88-88#    88-96     88-96#    88-104    88-104#   88-111
        88-111#   98-36     98-36#    128-70    128-70#   128-95    128-95#   129-36    129-36#   129-119   129-119#  130-44    130-44#   130-54
        130-54#   132-34    132-34#   139-40    139-40#
T$EXCP  63-28     63-28#    66-33     66-33#    144-21    144-21#   144-23    144-23#   144-25    144-25#   144-27    144-27#   144-29    144-29#
T$FLAG  36-35     36-35#    36-35#    37-19     37-19#    37-19#    37-44     37-44#    37-44#    37-67     37-67#    37-67#    37-88     37-88#
        37-88#    38-19     38-19#    38-19#    38-40     38-40#    38-40#    39-16     39-16#    39-16#    117-79    117-79#   117-79#   118-40
        118-40#   118-40#   127-34    127-34    127-34#   127-34#   134-13    134-13    134-13    134-13#   134-13#   135-18    135-18#   136-11
        136-11#   136-11#   137-7     137-7     137-7#    137-7#    137-11    137-11    137-11#   137-11#   137-72    137-72    137-72#   137-72#
        137-78    137-78    137-78#   137-78#   138-7     138-7     138-7#    138-7#    143-28    143-28    143-28#   143-28#
T$FREE  145-25    146-10#
T$GMAN   1-I02#   63-28     63-28#    63-28#    66-33     66-33#    66-33#
T$HILI  63-28     63-28#    66-33     66-33#    144-21    144-21#   144-23    144-23#   144-25    144-25#   144-27    144-27#   144-29    144-29#
T$LAST   1-I02#  145-25#   146-2
T$LOLI  63-28     63-28#    66-33     66-33#    144-21    144-21#   144-23    144-23#   144-25    144-25#   144-27    144-27#   144-29    144-29#
T$LSYM   1-I02     1-I02#    3-17      4-21      36-16     36-37     37-21     37-46     37-69     37-90     38-21     38-42     39-18     42-13
        43-23     44-10     44-13     118-52    132-39    133-12    134-17    135-26    136-15    137-82    143-32    144-30    145-14
T$LTNO  145-25#
T$NEST   1-I02#    1-I17     1-I17     1-I17#    3-10      3-10      3-10#     3-17      3-17      3-17      3-17#     4-10      4-10      4-10#
         4-21      4-21      4-21      4-21#     4-22      4-22      4-22      4-22#     5-3       5-3       5-3#      36-7      36-7      36-7#
        36-16     36-16     36-16     36-16#    36-23     36-23     36-23#    36-37     36-37     36-37     36-37#    37-7      37-7      37-7#
        37-21     37-21     37-21     37-21#    37-28     37-28     37-28#    37-46     37-46     37-46     37-46#    37-53     37-53     37-53#
        37-69     37-69     37-69     37-69#    37-76     37-76     37-76#    37-90     37-90     37-90     37-90#    38-7      38-7      38-7#
        38-21     38-21     38-21     38-21#    38-28     38-28     38-28#    38-42     38-42     38-42     38-42#    39-7      39-7      39-7#
        39-18     39-18     39-18     39-18#    42-11     42-11     42-11#    42-13     42-13     42-13     42-13#    43-20     43-20     43-20#
        43-23     43-23     43-23     43-23#    44-6      44-6      44-6#     44-10     44-10     44-10     44-10#    44-11     44-11     44-11#
        44-13     44-13     44-13     44-13#    67-102    67-102    67-102    67-102#   117-56    117-56    117-56#   117-62    117-62    117-62#
        118-52    118-52    118-52    118-52#   119-9     119-9     119-9#    119-15    119-15    119-15    119-15#   120-26    120-26    120-26#
        132-39    132-39    132-39    132-39#   133-10    133-10    133-10#   133-12    133-12    133-12    133-12#   134-8     134-8     134-8#
        134-17    134-17    134-17    134-17#   135-8     135-8     135-8#    135-26    135-26    135-26    135-26#   136-9     136-9     136-9#
        136-15    136-15    136-15    136-15#   136-16    136-16    136-16    136-16#   136-17    136-17    136-17#   137-3     137-3     137-3#
        137-82    137-82    137-82    137-82#   138-3     138-3     138-3#    143-32    143-32    143-32    143-32#   143-34    143-34    143-34
        143-34#   144-2     144-2     144-2#    144-12    144-12    144-12#   144-30    144-30    144-30    144-30#   145-12    145-12    145-12#
        145-14    145-14    145-14    145-14#   145-26    145-26    145-26    145-26#
T$NSO    1-I17#    4-22      5-3#      67-102    117-56#   136-16    136-17#   143-34    144-2#    145-26
T$NS1    3-10#     3-17      4-10#     4-21      36-7#     36-16     36-23#    36-37     37-7#     37-21     37-28#    37-46     37-53#    37-69
        37-76#    37-90     38-7#     38-21     38-28#    38-42     39-7#     39-18     42-11#    42-13     43-20#    43-23     44-6#     44-10
        44-11#    44-13     117-62#   118-52    119-9#    119-15    120-26#   132-39    133-10#   133-12    134-8#    134-17    135-8#    135-26
        136-9#    136-15    137-3#    137-82    138-3#    143-32    144-12#   144-30    145-12#   145-14
T$PCNT  146-2#    146-3     146-3     146-3#
T$PTAB  146-3     146-3#
T$PTHV   1-I25    146-10#
T$PTNU   1-I02#   146-3     146-3#    146-10    146-10
T$SAVL   1-I02#
T$SEGL   1-I02#
T$SIZE  145-25    146-10#
```

```
T$SUBN    1-IO2#  137-3#   138-3#
T$TAGL    1-IO2#
T$TAGN    1-IO2#   3-10     3-10      3-10#    4-10     4-10     4-10#    36-7     36-7     36-7#    36-23    36-23    36-23#   37-7
         37-7     37-7#   37-28    37-28    37-28#   37-53    37-53    37-53#   37-76    37-76    37-76#   38-7     38-7     38-7#
         38-28    38-28   38-28#   39-7     39-7     39-7#    42-11    42-11    42-11#   43-20    43-20    43-20#   44-6     44-6
         44-6#    44-11   44-11    44-11#   117-62   117-62   117-62#  119-9    119-9    119-9#   120-26   120-26   120-26#  133-10
         133-10   133-10# 134-8    134-8    134-8#   135-8    135-8    135-8#   136-9    136-9    136-9#   137-3    137-3    137-3#
         138-3    138-3   138-3#   144-12   144-12   144-12#  145-12   145-12   145-12#  146-2    146-2    146-2#   146-3    146-3
         146-3    146-3   146-3#   146-3#
T$TEMP    2-8      2-8     2-8      2-8      2-8#     2-8#     2-8#     3-17     3-17#    4-21     4-21#    4-22     4-22#    36-16
         36-16#   36-35   36-35#   36-37    36-37#   37-19    37-19#   37-21    37-21#   37-44    37-44#   37-46    37-46#   37-67
         37-67#   37-69   37-69#   37-88    37-88#   37-90    37-90#   38-19    38-19#   38-21    38-21#   38-40    38-40#   38-42
         38-42#   39-16   39-16#   39-18    39-18#   42-13    42-13#   43-23    43-23#   44-10    44-10#   44-13    44-13#   61-20
         61-20    61-20   61-20#   61-20#   61-20#   63-28    63-28    63-28    63-28#   63-28#   63-28#   64-27    64-27    64-27
         64-27#   64-27#  64-27#   66-33    66-33    66-33    66-33#   66-33#   66-33#   67-102   67-102#  117-79   117-79#  118-40
         118-40#  118-52  118-52#  119-15   119-15#  127-34   127-34#  132-39   132-39#  133-12   133-12#  134-13   134-13#  134-17
         134-17#  135-18  135-18#  135-26   135-26#  136-11   136-11#  136-15   136-15#  136-16   136-16#  137-7    137-7#   137-11
         137-11#  137-72  137-72#  137-78   137-78#  137-82   137-82#  138-7    138-7#   143-28   143-28#  143-32   143-32#  143-34
         143-34#  144-21  144-21   144-21   144-21#  144-21#  144-21#  144-23   144-23   144-23   144-23#  144-23#  144-23#  144-25
         144-25   144-25  144-25#  144-25#  144-25#  144-27   144-27   144-27   144-27#  144-27#  144-27#  144-29   144-29   144-29
         144-29#  144-29# 144-29#  144-30   144-30#  145-14   145-14#  145-26   145-26
T$TEST    1-IO2#  137-3    137-3    137-3#   138-3    138-3    138-3#   145-25
T$TSTM    1-IO2#  36-16   36-37    37-21    37-46    37-69    37-90    38-21    38-42    39-18    45-18    45-20    45-34    45-37
         47-46    48-7    48-19    50-19    51-7     52-42    57-44    58-13    58-23    58-42    59-13    61-20    63-28    64-27
         66-33    70-32   73-40    76-38    79-42    79-76    79-86    82-47    84-41    84-68    85-40    85-59    85-81    88-80
         88-88    88-96   88-104   88-111   91-35    97-28    97-37    97-58    97-85    98-36    99-18    99-20    99-22    99-24
         118-52   120-28  120-34   120-37   121-7    121-9    121-19   122-5    122-10   122-15   123-16   123-18   124-7    124-24
         127-28   127-30  127-34   128-70   128-95   129-15   129-27   129-30   129-36   129-119  130-33   130-44   130-54   132-34
         132-35   132-39  133-12   134-13   134-17   135-26   136-15   137-7    137-9    137-11   137-57   137-71   137-72   137-77
         137-78   137-82  138-7    138-34   139-4    139-40   143-6    143-23   143-28   143-32
T$TSTS    1-IO2#  137-3#   138-3#
T1        2-8     137-3#
T2        2-8     138-3#
TBLNDX    19-29#  45-16*   45-22*   45-23    124-16*  124-19*  125-4    127-4    138-14*  138-21*  138-22   143-21
TBUFF     69-35    75-32   75-48    77-31    80-25    83-23    145-20#
TF.BAD    21-62#  67-46   67-95    70-25    72-11
TF.BBR    21-60#  54-14   140-12
TF.BBU    21-59#
TF.DST    21-55#  55-33   67-60
TF.ECC    21-56#  55-34   67-65    67-70
TF.FER    21-53#  55-31   67-50    70-25
TF.HCE    21-54#  55-32   67-55
TF.LOG    21-58#
TF.SEX    21-57#
TF.WFE    21-61#  54-4    79-54    140-28   141-41
TIMDAT    18-10#
TINDEX    14-22   14-23   14-23    14-23#   14-24    14-24    14-24#   14-25    14-25    14-25#   14-26    14-26    14-26#   14-27
         14-27    14-27#  14-28    14-28    14-28#   14-29    14-29    14-29#   14-30    14-30    14-30#   14-31    14-31    14-31#
         14-32    14-32   14-32#   14-33    14-33    14-33#   14-34    14-34    14-34#   14-36    15-26#   15-27    15-27    15-27#
         15-28    15-28   15-28#   15-29    15-29    15-29#   15-30    15-30    15-30#   15-31    15-31    15-31#   15-32    15-32
         15-32#   15-33   15-33    15-33#   15-34    15-34    15-34#   15-35    15-35    15-35#   15-36    15-36    15-36#   15-37
         15-37    15-37#  15-38    15-38    15-38#   15-39    15-39    15-39#   15-40    15-40    15-40#   15-41    15-41    15-41#
         15-42    15-42   15-42#   15-43    15-43    15-43#   15-44    15-44    15-44#   16-2     16-2     16-2#    16-3     16-3
         16-3#    16-4    16-4     16-4#    16-5     16-5     16-5#    16-6     16-6     16-6#    16-7     16-7     16-7#    16-8
         16-8     16-8#   16-9     16-9     16-9#    16-10    16-10    16-10#   16-11    16-11    16-11#   16-12    16-12    16-12#
         16-13    16-13   16-13#   16-15    144-13#  144-14   144-14   144-14#  144-15   144-15   144-15#  144-16   144-16   144-16#
```

	144-17	144-17	144-17#	144-18	144-18	144-18#							
TMPSTA	21-17#	84-27*	84-45*	84-60	85-26*	85-44*	85-73						
TRKSIZ	21-5#	58-17*	71-15	94-34									
TTYIN	22-8#	66-25*	66-33	66-34	66-52*	66-56	116-23	117-20					
TTYOUT	99-8*	99-18	99-20	99-22	99-24	99-27#							
TX.BBR	34-44#	54-18											
TX.BBU	34-45#												
TX.BLK	34-48#	53-33	53-46										
TX.DST	34-51#	55-40											
TX.ECH	34-43#	55-41											
TX.FER	34-42#	55-38											
TX.HCE	34-50#	55-39											
TX.LB1	34-36#	138-44											
TX.LB2	34-37#	138-50											
TX.LOG	34-47#												
TX.PRI	34-39#	55-13											
TX.SEC	34-40#	55-23											
TX.SEX	34-46#												
TX.UNS	34-41#	53-35											
TX.WFE	34-49#	54-8											
TXT000	34-30#	137-34											
TXT001	34-31#												
TXT002	34-32#												
TXT005	34-33#												
TXT006	34-34#	61-20											
TYPSTA	53-22#	80-43	137-48	140-18	141-31								
UADR	20-10#	59-49*	60-6*	69-28*	69-36	72-3*	73-25*	75-25*	75-34	76-27*	77-35	78-37	78-73	79-45*
	80-26*	81-28*	81-36*	82-26*	83-24*	86-26	94-53	95-23	98-27*	98-44	139-11*		
UAM	5-9#												
UF.576	10-56#												
UF.CMR	10-47#												
UF.CMW	10-48#												
UF.INA	10-50#												
UF.RPL	10-49#												
UF.SCH	10-51#												
UF.SCL	10-52#												
UF.WBN	10-53#												
UF.WPH	10-54#												
UF.WPS	10-55#												
UNITIN	57-31#	137-18	138-26										
UNKNOW	24-22	24-29#											
UNSIZ	21-10#	59-17*	59-18*	59-62	59-62	65-22	65-23	66-38	66-39	86-29	86-30	138-48	138-49
UNUSE	21-14#	53-31*	65-33*	98-26*	98-55	98-57	98-80*	117-73					
VBUFF1	21-26#	78-27*	78-47*	81-26									
VBUFF2	21-27#	78-28*	78-83*	81-34									
VIBUFF	21-25#	70-19*	70-24*	72-10*	73-24*	73-32*							
VOLSN	18-3#												
VSNBUF	21-97#	95-28											
VSNPTR	21-96#	59-62	95-46*										
VWP	18-16#												
W	97-28#												
WAIT	88-31	91-19#											
WBC	18-15#												
WFECNT	21-75#	54-9*	65-40*	117-68									
WRSTAT	21-68#												
X$ALWA	1-I02#												

```
X$FALS    1-I02#
X$OFFS    1-I02#
X$TRUE    1-I02#
XACCES   28-13     28-20#
XBLOCK   26-14     26-28#
XCMDRF   25-18     25-42#
XCODE    25-19     25-43#
XCOPY    26-15     26-29#
XDAINT   57-47    128-27#
XDAIST  128-58    129-14#
XDARSD  129-42*   129-52*   129-53   130-20   130-21*   130-30   130-66#
XDARSP  128-77    129-48    130-19#
XDASRV   43-20#   125-28
XDCODE   27-30#    50-32
XDHEX    27-32#    50-39
XDINFO   27-31#    50-34
XECODE   25-22     25-46#
XENDRF   25-21     25-45#
XESTAT   25-24     25-48#    36-13
XEVENT   27-13     27-27#    39-12
XEVNT    27-28#    46-45     49-18
XFLAGS   25-23     25-47#
XHADDR   27-14     27-29#    48-20
XLBN     34-21#    50-21
XMOD     25-20     25-44#
XOFF     29-14
XON      29-13
XPHASE   26-16     26-30#
XRBN     34-22#
XRCT     34-20#
XREAD    28-11     28-18#
XREPLA   28-14     28-21#
XSADDR   23-12     23-29#
XSAEXP   23-14     23-30#
XSAVAL   23-13     23-28#
XSTAT    25-49#
XSTEP    34-23#
XWRITE   28-12     28-19#
```

```
AND      1-D68@
ASSUME   1-D96@
BCOMPL 121-8     121-10    122-6     122-11    122-16    137-10
BGNAU  136-9
BGNAUT 133-10
BGNCLN 134-8
BGNDU  135-8
BGNHRD 144-12
BGNHW    3-10
BGNINI 120-26
BGNMOD   1-I17    5-3      117-56    136-17    144-2
BGNMSG  36-7     36-23     37-7      37-28     37-53     37-76     38-7      38-28     39-7
BGNPRO 119-9
BGNPTA 146-3
BGNRPT 117-62
BGNSET 146-2
BGNSFT 145-12
BGNSRV  42-11    43-20     44-6      44-11
BGNSW    4-10
BGNTST 137-3     138-3
BNCOMP  97-38   120-38    124-8     124-25    138-35
BREAK   45-20    91-35    129-15    130-33    139-4
BRESET 12C-28
BRX      1-G64@
BUILD  105-6@   105-35    105-51
CHECKT   1-H37@
CLOCK  121-7    121-9
CLOSE   97-58
CLRVEC  45-37   129-30
DESCRI  29-25
DEVTYP  29-20
DISPAT   2-8
DOCLN  123-16   127-28    132-35
DODU   137-71   137-77    143-23
DORPT  137-57   143-6
END      1-E54@  14-36     16-15     144-19
ENDAU  136-15
ENDAUT 133-12
ENDCLN 134-17
ENDDU  135-26
ENDHRD 144-30
ENDHW    3-17
ENDINI 132-39
ENDMOD   4-22    67-102   136-16    143-34    145-26
ENDMSG  36-16    36-37     37-21     37-46     37-69     37-90     38-21     38-42     39-18
ENDPRO 119-15
ENDPTA 146-9
ENDRPT 118-52
ENDSET 146-10
ENDSFT 145-14
ENDSRV  42-13    43-23     44-10     44-13
ENDSW    4-21
ENDTST 137-82   143-32
ENTRY  105-25@  105-35    105-35    105-35    105-35    105-35    105-35    105-35    105-35    105-41@  105-51    105-51    105-51    105-51
       105-51   105-51    105-51    105-51
EQUALS   5-9
```

CZUDLAO BBR Replacement Utility MACRO Y05.03c Friday 19-Apr-85 10:42 Page M-2
Cross reference table (CREF V05.03)

```
ERRDF    45-34    47-46    48-7     48-19    50-19    51-7     52-42    58-13    58-23    59-13    84-68    85-81    88-80    88-88
         88-96    88-104   88-111   128-70   128-95   129-36   129-119  130-44   130-54   139-40
ERRHRD   58-42    73-40    76-38    79-42    79-76    79-86    82-47    84-41    85-40    85-59
ERRSF    132-34
ERRSOF   70-32    98-36
ERRTBL   19-5
EXIT     36-35    37-19    37-44    37-67    37-88    38-19    38-40    39-16    117-79   118-40   127-34   134-13   135-18   136-11
         137-7    137-11   137-72   137-78   138-7    143-28
EXIT$L   1-H12#
GETBYT   97-37
GMANID   63-28    66-33
GMANIL   61-20    64-27
GPHARD   124-24
GPRMA    144-21   144-23
GPRMD    63-28    63-28#   66-33    66-33#   144-25   144-27   144-29
GPRML    61-20    61-20#   64-27    64-27#
HEADER   1-I25
ITEM     1-E35#   14-23    14-24    14-25    14-26    14-27    14-28    14-29    14-30    14-31    14-32    14-33    14-34    15-27
         15-28    15-29    15-30    15-31    15-32    15-33    15-34    15-35    15-36    15-37    15-38    15-39    15-40    15-41
         15-42    15-43    15-44    16-2     16-3     16-4     16-5     16-6     16-7     16-8     16-9     16-10    16-11    16-12
         16-13    144-14   144-15   144-16   144-17   144-18
JMPX     1-G88#
LASTAD   145-25
M$BYTE   1-I25    1-I25    1-I25    1-I25    1-I25#
M$CHEC   36-35    36-35#   37-19    37-19#   37-44    37-44#   37-67    37-67#   37-88    37-88#   38-19    38-19#   38-40    38-40#
         39-16    39-16#   117-79   117-79#  118-40   118-40#  127-34   127-34#  134-13   134-13#  135-18   135-18#  136-11   136-11#
         137-7    137-7#   137-11   137-11#  137-72   137-72#  137-78   137-78#  138-7    138-7#   143-28   143-28#
M$CNTO   61-20    61-20#   63-28    63-28#   64-27    64-27#   66-33    66-33#   144-21   144-21#  144-23   144-23#  144-25   144-25#
         144-27   144-27#  144-29   144-29#
M$COUN   97-85    97-85#   99-18    99-18#   99-20    99-20#   99-22    99-22#   99-24    99-24#
M$DATA   1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25
         1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25    1-I25#   1-I25#   29-20    29-20#   29-25
         29-25#
M$DECR   3-17     3-17#    4-21     4-21#    4-22     4-22#    36-16    36-16#   36-37    36-37#   37-21    37-21#   37-46    37-46#
         37-69    37-69#   37-90    37-90#   38-21    38-21#   38-42    38-42#   39-18    39-18#   42-13    42-13#   43-23    43-23#
         44-10    44-10#   44-13    44-13#   67-102   67-102#  118-52   118-52#  119-15   119-15#  132-39   132-39#  133-12   133-12#
         134-17   134-17#  135-26   135-26#  136-15   136-15#  136-16   136-16#  137-82   137-82#  143-32   143-32#  143-34   143-34#
         144-30   144-30#  145-14   145-14#  145-26   145-26#  146-3    146-3#
M$DEFA   61-20    61-20#   63-28    63-28#   64-27    64-27#   66-33    66-33#   144-21   144-21#  144-23   144-23#  144-25   144-25#
         144-27   144-27#  144-29   144-29#
M$ENDE   3-17#    4-21#    4-22#    36-16#   36-37#   37-21#   37-46#   37-69#   37-90#   38-21#   38-42#   39-18#   42-13#   43-23#
         44-10#   44-13#   67-102#  118-52#  132-39#  133-12#  134-17#  135-26#  136-15#  136-16#  137-82#  143-32#  143-34#  144-30#
         145-14#  145-26#
M$ERRI   45-34    45-34#   47-46    47-46#   48-7     48-7#    48-19    48-19#   50-19    50-19#   51-7     51-7#    52-42    52-42#
         58-13    58-13#   58-23    58-23#   58-42    58-42#   59-13    59-13#   70-32    70-32#   73-40    73-40#   76-38    76-38#
         79-42    79-42#   79-76    79-76#   79-86    79-86#   82-47    82-47#   84-41    84-41#   84-68    85-40    85-40#
         85-59    85-59#   85-81    85-81#   88-80    88-80#   88-88    88-88#   88-96    88-96#   88-104   88-104#  88-111   88-111#
         98-36    98-36#   128-70   128-70#  128-95   128-95#  129-36   129-36#  129-119  129-119# 130-44   130-44#  130-54   130-54#
         132-34   132-34#  139-40   139-40#
M$EXCP   63-28    63-28    63-28#   66-33    66-33    66-33#   144-21   144-21   144-21#  144-23   144-23   144-23#  144-25   144-25
         144-25#  144-27   144-27   144-27#  144-29   144-29   144-29#
M$EXIT   36-35#   37-19#   37-44#   37-67#   37-88#   38-19#   38-40#   39-16#   117-79#  118-40#  127-34   127-34#  134-13   134-13#
         135-18#  136-11#  137-7    137-7#   137-11   137-11#  137-72   137-72#  137-78   137-78#  138-7    138-7#   143-28   143-28#
M$EXSE   36-35#   37-19#   37-44#   37-67#   37-88#   38-19#   38-40#   39-16#   117-79#  118-40#  127-34#  134-13#  135-18#  136-11#
         137-7#   137-11#  137-72#  137-78#  138-7#   143-28#
```

M$EXTJ 36-35	36-35#	37-19	37-19#	37-44	37-44#	37-67	37-67#	37-88	37-88#	38-19	38-19#	38-40	38-40#
39-16	39-16#	117-79	117-79#	118-40	118-40#	127-34#	134-13#	135-18	135-18#	136-11	136-11#	137-7#	137-11#
137-72#	137-78#	138-7#	143-28#										
M$GEN 1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25
1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25
1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25#	1-I25#	1-I25#	1-I25#
1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#
1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#
1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	2-8	2-8#	3-10	3-10	3-10#	3-10#	3-17
3-17# 4-10	4-10	4-10#	4-10#	4-21	4-21#	19-5	19-5#	29-20	29-20#	29-25	29-25#	36-7	36-7#
36-16	36-16#	36-23	36-23#	36-37	36-37#	37-7	37-7#	37-21	37-21#	37-28	37-28#	37-46	37-46#
37-53	37-53#	37-69	37-69#	37-76	37-76#	37-90	37-90#	38-7	38-7#	38-21	38-21#	38-28	38-28#
38-42	38-42#	39-7	39-7#	39-18	39-18#	42-11	42-11#	42-13	42-13#	43-20	43-20#	43-23	43-23#
44-6	44-6#	44-10	44-10#	44-11	44-11#	44-13	44-13#	61-20	61-20#	63-28	63-28#	64-27	64-27#
66-33	66-33#	117-62	117-62#	118-52	118-52#	119-9	119-9#	120-26	120-26#	132-39	132-39#	133-10	133-10#
133-12	133-12#	134-8	134-8#	134-17	134-17#	135-8	135-8#	135-26	135-26#	136-9	136-9#	136-15	136-15#
137-3	137-3#	137-82	137-82#	138-3	138-3#	143-32	143-32#	144-12	144-12#	144-30	144-30#	145-12	145-12#
145-14	145-14#	145-25	145-25#	146-3	146-3#	146-9	146-9#						
M$GENB 61-20	61-20#	63-28	63-28#	64-27	64-27#	66-33	66-33#						
M$GETS 3-17	3-17#	4-21	4-21#	4-22	4-22#	36-16	36-16#	36-37	36-37#	37-21	37-21#	37-46	37-46#
37-69	37-69#	37-90	37-90#	38-21	38-21#	38-42	38-42#	39-18	39-18#	42-13	42-13#	43-23	43-23#
44-10	44-10#	44-13	44-13#	67-102	67-102#	118-52	118-52#	119-15	119-15#	132-39	132-39#	133-12	133-12#
134-17	134-17#	135-26	135-26#	136-15	136-15#	136-16	136-16#	137-82	137-82#	143-32	143-32#	143-34	143-34#
144-30	144-30#	145-14	145-14#	145-26	145-26#								
M$GETT 36-35#	37-19#	37-44#	37-67#	37-88#	38-19#	38-40#	39-16#	117-79#	118-40#	127-34#	134-13#	135-18#	136-11#
137-7#	137-11#	137-72#	137-78#	138-7#	143-28#								
M$GNGB 1-I17#	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25
1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25
1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25#	1-I25#	1-I25#
1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#
1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	2-8	2-8#	3-10	3-10	3-10#	4-10	4-10
4-10#	5-3#	19-5	19-5#	29-20	29-20#	29-25	29-25#	36-7	36-7#	36-23	36-23#	37-7	37-7#
37-28	37-28#	37-53	37-53#	37-76	37-76#	38-7	38-7#	38-28	38-28#	39-7	39-7#	42-11	42-11#
43-20	43-20#	44-6	44-6#	44-11	44-11#	117-56#	117-62	117-62#	119-9	119-9#	120-26	120-26#	133-10
133-10#	134-8	134-8#	135-8	135-8#	136-9	136-9#	136-17#	144-2#	144-12	144-12#	145-12	145-12#	145-25
145-25#													
M$GNIN 1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25
1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25
1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25	1-I25
1-I25	1-I25	1-I25	1-I25	1-I25	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#
1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#
1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	1-I25#	2-8	2-8	2-8	2-8#	2-8#	2-8#	3-10
3-10#	4-10	4-10#	29-20	29-20	29-20#	29-20#	29-25	29-25	29-25#	29-25#	36-16	36-16#	36-35
36-35	36-35#	36-35#	36-37	36-37#	37-19	37-19	37-19#	37-19#	37-21	37-21#	37-44	37-44	37-44#
37-44#	37-46	37-46#	37-67	37-67	37-67#	37-69	37-69#	37-88	37-88	37-88#	37-88#	37-90	
37-90#	38-19	38-19	38-19#	38-19#	38-21	38-21#	38-40	38-40	38-40#	38-40#	38-42	38-42#	39-16
39-16	39-16#	39-16#	39-18	39-18#	42-13	42-13#	43-23	43-23#	44-10	44-10#	44-13	44-13#	45-18
45-18	45-18	45-18	45-18	45-18	45-18#	45-18#	45-18#	45-18#	45-18#	45-18#	45-20	45-20#	45-34
45-34	45-34	45-34	45-34#	45-34#	45-34#	45-34#	45-34#	45-37	45-37	45-37#	45-37#	47-46	47-46
47-46	47-46#	47-46#	47-46#	47-46#	47-46#	48-7	48-7	48-7	48-7#	48-7#	48-7#	48-7#	
48-7#	48-19	48-19	48-19	48-19	48-19	48-19#	48-19#	48-19#	48-19#	48-19#	50-19	50-19	50-19
50-19	50-19#	50-19#	50-19#	50-19#	50-19#	51-7	51-7	51-7	51-7	51-7#	51-7#	51-7#	
51-7#	52-42	52-42	52-42	52-42	52-42#	52-42#	52-42#	52-42#	52-42#	57-44	57-44	57-44	57-44
57-44	57-44	57-44#	57-44#	57-44#	57-44#	57-44#	57-44#	58-13	58-13	58-13	58-13	58-13#	58-13#

```
          58-13#   58-13#   58-13#   58-23    58-23    58-23    58-23    58-23#   58-23#   58-23#   58-23#   58-23#   58-42    58-42
          58-42    58-42    58-42#   58-42#   58-42#   58-42#   58-42#   59-13    59-13    59-13    59-13    59-13#   59-13#   59-13#
          59-13#   59-13#   61-20    61-20    61-20    61-20    61-20    61-20    61-20#   61-20#   61-20#   61-20#   63-28    63-28
          63-28    63-28    63-28    63-28    63-28    63-28    63-28#   63-28#   63-28#   63-28#   64-27    64-27    64-27    64-27
          64-27    64-27    64-27#   64-27#   64-27#   64-27#   66-33    66-33    66-33    66-33    66-33    66-33    66-33    66-33
          66-33#   66-33#   66-33#   66-33#   70-32    70-32    70-32    70-32    70-32#   70-32#   70-32#   70-32#   73-40    73-40
          73-40    73-40    73-40    73-40#   73-40#   73-40#   73-40#   76-38    76-38    76-38    76-38    76-38    76-38#   76-38#
          76-38#   76-38#   76-38#   79-42    79-42    79-42    79-42    79-42#   79-42#   79-42#   79-42#   79-42#   79-76    79-76
          79-76    79-76    79-76#   79-76#   79-76#   79-76#   79-76#   79-86    79-86    79-86    79-86    79-86#   79-86#   79-86#
          79-86#   79-86#   82-47    82-47    82-47    82-47    82-47#   82-47#   82-47#   82-47#   82-47#   84-41    84-41    84-41
          84-41    84-41#   84-41#   84-41#   84-41#   84-41#   84-68    84-68    84-68    84-68    84-68#   84-68#   84-68#   84-68#
          84-68#   85-40    85-40    85-40    85-40    85-40#   85-40#   85-40#   85-40#   85-59    85-59    85-59    85-59    85-59
          85-59#   85-59#   85-59#   85-59#   85-81    85-81    85-81    85-81    85-81#   85-81#   85-81#   85-81#   85-81#
          88-80    88-80    88-80    88-80    88-80#   88-80#   88-80#   88-80#   88-80#   88-88    88-88    88-88    88-88    88-88#
          88-88#   88-88#   88-88#   88-88#   88-96    88-96    88-96    88-96    88-96#   88-96#   88-96#   88-96#   88-104
          88-104   88-104   88-104   88-104#  88-104#  88-104#  88-104#  88-104#  88-111   88-111   88-111   88-111   88-111#  88-111#
          88-111#  88-111#  88-111#  91-35    91-35#   97-28    97-28    97-28    97-28#   97-28#   97-37    97-37    97-37#   97-38
          97-38#   97-58    97-58#   97-85    97-85    97-85    97-85    97-85#   97-85#   97-85#   97-85#   97-85#   97-85#   97-85#
          98-36    98-36    98-36    98-36    98-36#   98-36#   98-36#   98-36#   98-36#   99-18    99-18    99-18    99-18    99-18
          99-18    99-18#   99-18#   99-18#   99-18#   99-18#   99-18#   99-20    99-20    99-20    99-20    99-20#   99-20#   99-20#
          99-20#   99-20#   99-20#   99-22    99-22    99-22    99-22    99-22    99-22#   99-22#   99-22#   99-22#   99-22#   99-22#
          99-24    99-24    99-24    99-24    99-24    99-24    99-24#   99-24#   99-24#   99-24#   99-24#   99-24#   117-79   117-79#
          117-79#  118-40   118-40   118-40#  118-40#  118-52   118-52#  120-28   120-28#  120-34   120-34   120-34#  120-34#  120-37
          120-37#  120-38   120-38#  121-7    121-7    121-7#   121-7#   121-7#   121-8    121-8#   121-9    121-9    121-9#   121-9#
          121-9#   121-10   121-10#  121-19   121-19   121-19   121-19   121-19   121-19   121-19#  121-19#  121-19#  121-19#  121-19#
          121-19#  122-5    122-5    122-5#   122-5#   122-6    122-6#   122-10   122-10   122-10#  122-10#  122-11   122-11#  122-15
          122-15   122-15#  122-15#  122-16   122-16#  123-16   123-16#  123-18   123-18   123-18#  123-18#  124-7    124-7#   124-8
          124-8#   124-24   124-24   124-24   124-24#  124-24#  124-24#  124-25   124-25#  127-28   127-28#  127-30   127-30   127-30#
          127-30#  127-34   127-34   127-34#  127-34#  128-70   128-70   128-70   128-70#  128-70#  128-70#  128-70#  128-70#  128-70#
          128-95   128-95   128-95   128-95   128-95#  128-95#  128-95#  128-95#  128-95#  129-15   129-15#  129-27   129-27   129-27
          129-27   129-27   129-27   129-27#  129-27#  129-27#  129-27#  129-27#  129-27#  129-30   129-30   129-30#  129-30#  129-36
          129-36   129-36   129-36   129-36#  129-36#  129-36#  129-36#  129-36#  129-119  129-119  129-119  129-119  129-119# 129-119#
          129-119# 129-119# 129-119# 130-33   130-33#  130-44   130-44   130-44   130-44   130-44#  130-44#  130-44#  130-44#  130-44#
          130-54   130-54   130-54   130-54   130-54#  130-54#  130-54#  130-54#  132-34   132-34   132-34   132-34   132-34   132-34#
          132-34#  132-34#  132-34#  132-34#  132-35   132-35#  132-39   132-39#  133-12   133-12#  134-13   134-13   134-13#  134-13#
          134-17   134-17#  135-18   135-18   135-18#  135-18#  135-26   135-26#  136-11   136-11   136-11#  136-11#  136-15   136-15#
          137-7    137-7    137-7#   137-7#   137-9    137-9#   137-10   137-10#  137-11   137-11   137-11#  137-11#  137-57   137-57#
          137-71   137-71   137-71#  137-71#  137-72   137-72   137-72#  137-72#  137-77   137-77   137-77#  137-77#  137-78   137-78
          137-78#  137-78#  137-82   137-82#  138-7    138-7    138-7#   138-7#   138-34   138-34#  138-35   138-35#  139-4    139-4#
          139-40   139-40   139-40   139-40   139-40#  139-40#  139-40#  139-40#  139-40#  143-6    143-6#   143-23   143-23   143-23#
          143-23#  143-28   143-28   143-28#  143-28#  143-32   143-32#  144-12   144-12#  144-21   144-21   144-21   144-21   144-21#
          144-23   144-23   144-23   144-23   144-23#  144-25   144-25   144-25   144-25   144-25   144-25#  144-27   144-27   144-27
          144-27   144-27   144-27#  144-29   144-29   144-29   144-29   144-29#  144-30   144-30#  145-12   145-12#  145-14   145-14
          145-14#  145-25   145-25   145-25   145-25#  146-3    146-3    146-3#   146-3#

M$GNLS    61-20    61-20#   63-28    63-28#   64-27    64-27#   66-33    66-33#
M$GNTA    3-17     3-17#    4-21     4-21#    36-16    36-16#   36-37    36-37#   37-21    37-21#   37-46    37-46#   37-69    37-69#
          37-90    37-90#   38-21    38-21#   38-42    38-42#   39-18    39-18#   42-13    42-13#   43-23    43-23#   44-10    44-10#
          44-13    44-13#   118-52   118-52#  132-39   132-39#  133-12   133-12#  134-17   134-17#  135-26   135-26#  136-15   136-15#
          137-82   137-82#  143-32   143-32#  144-30   144-30#  145-14   145-14#  146-3    146-3#   146-9    146-9#
M$GNTE    137-3    137-3#   138-3    138-3#
M$HAPT    1-I25    1-I25#
M$HNAP    1-I25    1-I25#
M$INCR    1-I17    1-I17#   3-10     3-10     3-10#    3-10#    4-10     4-10     4-10#    4-10#    5-3      5-3#     36-7     36-7
          36-7#    36-7#    36-16#   36-23    36-23    36-23#   36-23#   36-37#   37-7     37-7     37-7#    37-7#    37-21#   37-28
          37-28    37-28#   37-28#   37-46#   37-53    37-53    37-53#   37-53#   37-69#   37-76    37-76    37-76#   37-76#   37-90#
          38-7     38-7     38-7#    38-7#    38-21#   38-28    38-28    38-28#   38-28#   38-42#   39-7     39-7     39-7#    39-7#
```

```
        39-18♦  42-11   42-11   42-11♦  42-11♦  43-20   43-20   43-20♦  43-20♦  44-6    44-6    44-6♦   44-6♦   44-11
        44-11   44-11♦  44-11♦  45-18♦  45-20♦  45-34♦  45-37♦  47-46♦  48-7♦   48-19♦  50-19♦  51-7♦   52-42♦  57-44♦
        58-13♦  58-23♦  58-42♦  59-13♦  61-20   61-20♦  61-20♦  63-28   63-28♦  63-28♦  64-27   64-27♦  64-27♦  66-33
        66-33♦  66-33♦  70-32♦  73-40♦  76-38♦  79-42♦  79-76♦  79-86♦  82-47♦  84-41♦  84-68♦  85-40♦  85-59♦  85-81♦
        88-80♦  88-88♦  88-96♦  88-104♦ 88-111♦ 91-35♦  97-28♦  97-37♦  97-58♦  97-85♦  98-36♦  99-18♦  99-20♦  99-22♦
        99-24♦  117-56  117-56♦ 117-62  117-62  117-62♦ 117-62♦ 118-52♦ 119-9   119-9   119-9♦  119-9♦  120-26  120-26
        120-26♦ 120-26♦ 120-28♦ 120-34♦ 120-37♦ 121-7♦  121-9♦  121-19♦ 122-5♦  122-10♦ 122-15♦ 123-16♦ 123-18♦ 124-7♦
        124-24♦ 127-28♦ 127-30♦ 127-34♦ 128-70♦ 128-95♦ 129-15♦ 129-27♦ 129-30♦ 129-36♦ 129-119♦ 130-33♦ 130-44♦ 130-54♦
        132-34♦ 132-35♦ 132-39♦ 133-10  133-10♦ 133-10♦ 133-10♦ 133-12♦ 134-8   134-8   134-8♦  134-8♦  134-13♦ 134-17♦
        135-8   135-8   135-8♦  135-8♦  135-26♦ 136-9   136-9   136-9♦  136-9♦  136-15♦ 136-17  136-17♦ 137-3   137-3
        137-3   137-3♦  137-3♦  137-3♦  137-7♦  137-9♦  137-11♦ 137-57♦ 137-71♦ 137-72♦ 137-77♦ 137-78♦ 137-82♦ 138-3
        138-3   138-3   138-3♦  138-3♦  138-3♦  138-7♦  138-34♦ 139-4♦  139-40♦ 143-6♦  143-23♦ 143-28♦ 143-32♦ 144-2
        144-2♦  144-12  144-12  144-12♦ 144-12♦ 145-12  145-12  145-12♦ 145-12♦ 146-2   146-2♦  146-3   146-3   146-3
        146-3♦

M$LDRO  45-37   45-37♦  97-28   97-28♦  121-7   121-7♦  121-9   121-9♦  122-5   122-5♦  122-10  122-10♦ 122-15  122-15♦
        124-24  124-24♦ 127-30  127-30♦ 129-30  129-30♦ 137-71  137-71♦ 137-77  137-77♦ 143-23  143-23♦

M$MCHI  1-I02   1-I02♦
M$MCLO  1-I02   1-I02♦
M$POP   3-17    3-17♦   4-21    4-21♦   4-22    4-22♦   36-16   36-16♦  36-37   36-37♦  37-21   37-21♦  37-46   37-46♦
        37-69   37-69♦  37-90   37-90♦  38-21   38-21♦  38-42   38-42♦  39-18   39-18♦  42-13   42-13♦  43-23   43-23♦
        44-10   44-10♦  44-13   44-13♦  67-102  67-102♦ 118-52  118-52♦ 119-15  119-15♦ 132-39  132-39♦ 133-12  133-12♦
        134-17  134-17♦ 135-26  135-26♦ 136-15  136-15♦ 136-16  136-16♦ 137-82  137-82♦ 143-32  143-32♦ 143-34  143-34♦
        144-30  144-30♦ 145-14  145-14♦ 145-26  145-26♦

M$PRIN  97-85   97-85♦  99-18   99-18♦  99-20   99-20♦  99-22   99-22♦  99-24   99-24♦
M$PUSH  1-I17   1-I17♦  3-10    3-10♦   4-10    4-10♦   5-3     5-3♦    36-7    36-7♦   36-23   36-23♦  37-7    37-7♦
        37-28   37-28♦  37-53   37-53♦  37-76   37-76♦  38-7    38-7♦   38-28   38-28♦  39-7    39-7♦   42-11   42-11♦
        43-20   43-20♦  44-6    44-6♦   44-11   44-11♦  117-56  117-56♦ 117-62  117-62♦ 119-9   119-9♦  120-26  120-26♦
        133-10  133-10♦ 134-8   134-8♦  135-8   135-8♦  136-9   136-9♦  136-17  136-17♦ 137-3   137-3♦  138-3   138-3♦
        144-2   144-2♦  144-12  144-12♦ 145-12  145-12♦

M$PUT   45-18   45-18   45-18   45-18   45-18♦  57-44   57-44   57-44   57-44♦  97-85   97-85   97-85   97-85♦
        99-18   99-18   99-18   99-18♦  99-20   99-20   99-20   99-20♦  99-22   99-22   99-22   99-22♦  99-24   99-24
        99-24   99-24♦  121-19  121-19  121-19  121-19  121-19♦ 129-27  129-27  129-27  129-27  129-27♦

M$PUT1  45-18   45-18   45-18   45-18   45-18♦  45-18   45-18   45-18♦  57-44   57-44   57-44   57-44   57-44♦  57-44♦
        57-44♦  57-44♦  97-85   97-85   97-85   97-85♦  97-85♦  97-85♦  99-18   99-18   99-18   99-18♦  99-18♦  99-18♦
        99-20   99-20   99-20   99-20♦  99-20♦  99-20♦  99-22   99-22   99-22   99-22♦  99-22♦  99-22♦  99-24   99-24
        99-24   99-24♦  99-24♦  99-24♦  121-19  121-19  121-19  121-19  121-19♦ 121-19♦ 121-19♦ 121-19♦ 129-27  129-27
        129-27

M$RADI  61-20   61-20♦  63-28   63-28♦  64-27   64-27♦  66-33   66-33♦  144-21  144-21♦ 144-23  144-23♦ 144-25  144-25♦
        144-27  144-27♦ 144-29  144-29♦

M$RBRO  97-37♦
M$RNRO  97-37   120-34  120-34♦ 121-7   121-7♦  121-9   121-9♦  123-18  123-18♦ 124-24  124-24♦
M$SETS  1-I17   1-I17♦  3-10    3-10♦   4-10    4-10♦   5-3     5-3♦    36-7    36-7♦   36-23   36-23♦  37-7    37-7♦
        37-28   37-28♦  37-53   37-53♦  37-76   37-76♦  38-7    38-7♦   38-28   38-28♦  39-7    39-7♦   42-11   42-11♦
        43-20   43-20♦  44-6    44-6♦   44-11   44-11♦  117-56  117-56♦ 117-62  117-62♦ 119-9   119-9♦  120-26  120-26♦
        133-10  133-10♦ 134-8   134-8♦  135-8   135-8♦  136-9   136-9♦  136-17  136-17♦ 137-3   137-3♦  138-3   138-3♦
        144-2   144-2♦  144-12  144-12♦ 145-12  145-12♦

M$SVC   36-16   36-16♦  36-35♦  36-37   36-37♦  37-19♦  37-21   37-21♦  37-44♦  37-46   37-46♦  37-67♦  37-69   37-69♦
        37-88♦  37-90   37-90♦  38-19♦  38-21   38-21♦  38-40♦  38-42   38-42♦  39-16♦  39-18   39-18♦  45-18   45-18♦
        45-20   45-20♦  45-34   45-37   45-37♦  47-46   48-7    48-19   50-19   51-7    52-42   57-44   57-44♦  58-13
        58-23   58-42   59-13   61-20   61-20♦  63-28   63-28♦  64-27   64-27♦  66-33   66-33♦  70-32   73-40   76-38
        79-42   79-76   79-86   82-47   84-41   84-68   85-40   85-59   85-81   88-80   88-88   88-96   88-104  88-111
        91-35   91-35♦  97-28   97-28♦  97-37   97-37♦  97-58   97-58♦  97-85   97-85♦  98-36   99-18   99-18♦  99-20
        99-20♦  99-22   99-22♦  99-24   99-24♦  117-79♦ 118-40♦ 118-52  118-52♦ 122-5   122-5♦  122-10  122-10♦ 122-15  122-15♦ 123-16
        120-37♦ 121-7   121-7♦  121-9   121-9♦  121-19  121-19♦ 122-5   122-5♦  122-10  122-10♦ 122-15  122-15♦ 123-16
        123-16♦ 123-18  123-18♦ 124-7   124-7♦  124-24  124-24♦ 127-28  127-28♦ 127-30  127-30♦ 127-34  127-34♦ 128-70
        128-95  129-15  129-15♦ 129-27  129-27♦ 129-30  129-30♦ 129-36  129-119 130-33  130-33♦ 130-44  130-54  132-34
```

```
         132-35   132-35◊  132-39   132-39◊  133-12   133-12◊  134-13   134-13◊  134-17   134-17◊  135-18◊  135-26   135-26◊  136-11◊
         136-15   136-15◊  137-7    137-7◊   137-9    137-9◊   137-11   137-11◊  137-57   137-57◊  137-71   137-71◊  137-72   137-72◊
         137-77   137-77◊  137-78   137-78◊  137-82   137-82◊  138-7    138-7◊   138-34   138-34◊  139-4    139-4◊   139-40   143-6
         143-6◊   143-23   143-23◊  143-28   143-28◊  143-32   143-32◊

MSTLAB   36-16◊   36-37◊   37-21◊   37-46◊   37-69◊   37-90◊   38-21◊   38-42◊   39-18◊   45-18◊   45-20◊   45-34◊   45-37◊   47-46◊
         48-7◊    48-19◊   50-19◊   51-7◊    52-42◊   57-44◊   58-13◊   58-23◊   58-42◊   59-13◊   61-20◊   63-28◊   64-27◊   66-33◊
         70-32◊   73-40◊   76-38◊   79-42◊   79-76◊   79-86◊   82-47◊   84-41◊   84-68◊   85-40◊   85-59◊   85-81◊   88-80◊   88-88◊
         88-96◊   88-104◊  88-111◊  91-35◊   97-28◊   97-37◊   97-58◊   97-85◊   98-36◊   99-18◊   99-20◊   99-22◊   99-24◊   118-52◊
         120-28◊  120-34◊  120-37◊  121-7◊   121-9◊   121-19◊  122-5◊   122-10◊  122-15◊  123-16◊  123-18◊  124-7◊   124-24◊  127-28◊
         127-30◊  127-34◊  128-70◊  128-95◊  129-15◊  129-27◊  129-30◊  129-36◊  129-119◊ 130-33◊  130-44◊  130-54◊  132-34◊  132-35◊
         132-39◊  133-12◊  134-13◊  134-17◊  135-26◊  136-15◊  137-7◊   137-9◊   137-11◊  137-57◊  137-71◊  137-72◊  137-77◊  137-78◊
         137-82◊  138-7◊   138-34◊  139-4◊   139-40◊  143-6◊   143-23◊  143-28◊  143-32◊

MSTSTL   36-16    36-16◊   36-37    36-37◊   37-21    37-21◊   37-46    37-46◊   37-69    37-69◊   37-90    37-90◊   38-21    38-21◊
         38-42    38-42◊   39-18    39-18◊   45-18    45-18◊   45-20    45-20◊   45-34    45-34◊   45-34◊   45-37    45-37◊   47-46
         47-46◊   47-46◊   48-7     48-7◊    48-7◊    48-19    48-19◊   48-19◊   50-19    50-19◊   50-19◊   51-7     51-7◊    51-7◊
         52-42    52-42◊   52-42◊   57-44    57-44◊   58-13    58-13◊   58-13◊   58-23    58-23◊   58-42    58-42◊   58-42◊
         59-13    59-13◊   59-13◊   61-20    61-20◊   63-28    63-28◊   64-27    64-27◊   66-33    66-33◊   70-32    70-32◊   70-32◊
         73-40    73-40◊   73-40◊   76-38    76-38◊   76-38◊   79-42    79-42◊   79-42◊   79-76    79-76◊   79-86    79-86◊
         79-86◊   82-47    82-47◊   82-47◊   84-41    84-41◊   84-41◊   84-68    84-68◊   84-68◊   85-40    85-40◊   85-40◊   85-59
         85-59◊   85-59◊   85-81    85-81◊   85-81◊   88-80    88-80◊   88-80◊   88-88    88-88◊   88-88◊   88-96    88-96◊   88-96◊
         88-104   88-104◊  88-104◊  88-111   88-111◊  88-111◊  91-35    91-35◊   97-28    97-28◊   97-37    97-37◊   97-58    97-58◊
         97-85    97-85◊   98-36    98-36◊   99-18    99-18◊   99-20    99-20◊   99-22    99-22◊   99-24    99-24◊   118-52
         118-52◊  120-28   120-28◊  120-34   120-34◊  120-37   120-37◊  121-7    121-7◊   121-9    121-9◊   121-19   121-19◊  122-5
         122-5◊   122-10   122-10◊  122-15   122-15◊  123-16   123-16◊  123-18   123-18◊  124-7    124-7◊   124-24   124-24◊  127-28
         127-28◊  127-30   127-30◊  127-34   127-34◊  128-70   128-70◊  128-70◊  128-95   128-95◊  128-95◊  129-15   129-15◊  129-27
         129-27◊  129-30   129-30◊  129-36   129-36◊  129-36◊  129-119  129-119◊ 129-119◊ 130-33   130-33◊  130-44   130-44◊  130-44◊
         130-54   130-54◊  130-54◊  132-34   132-34◊  132-34◊  132-35◊  132-39   132-39◊  133-12   133-12◊  134-13   134-13◊
         134-17   134-17◊  135-26   135-26◊  136-15   136-15◊  137-7    137-7◊   137-9    137-9◊   137-11   137-11◊  137-57   137-57◊
         137-71   137-71◊  137-72   137-72◊  137-77   137-77◊  137-78   137-78◊  137-82   137-82◊  138-7    138-7◊   138-34   138-34◊
         139-4    139-4◊   139-40   139-40◊  139-40◊  143-6    143-6◊   143-23   143-23◊  143-28   143-28◊  143-32   143-32◊

MSWORD   1-I25    1-I25◊   2-8      2-8      2-8      2-8◊     36-35    36-35◊   37-19    37-19◊   37-44    37-44◊   37-67    37-67◊
         37-88    37-88◊   38-19    38-19◊   38-40    38-40◊   39-16    39-16◊   45-34    45-34    45-34    45-34◊   47-46    47-46
         47-46    47-46◊   48-7     48-7     48-7     48-7◊    48-19    48-19    48-19    48-19◊   50-19    50-19    50-19    50-19◊
         51-7     51-7     51-7     51-7◊    52-42    52-42    52-42    52-42◊   58-13    58-13    58-13    58-13◊   58-23    58-23
         58-23    58-23◊   58-42    58-42    58-42    58-42◊   59-13    59-13    59-13    59-13◊   61-20    61-20    61-20◊   61-20◊
         63-28    63-28    63-28◊   63-28◊   64-27    64-27    64-27◊   64-27◊   66-33    66-33    66-33◊   66-33◊   70-32    70-32
         70-32    70-32◊   73-40    73-40    73-40    73-40◊   76-38    76-38    76-38    76-38◊   79-42    79-42    79-42    79-42◊
         79-76    79-76    79-76    79-76◊   79-86    79-86    79-86    79-86◊   82-47    82-47    82-47    82-47◊   84-41    84-41
         84-41    84-41◊   84-68    84-68    84-68    84-68◊   85-40    85-40    85-40    85-40◊   85-59    85-59    85-59    85-59◊
         85-81    85-81    85-81    85-81◊   88-80    88-80    88-80    88-80◊   88-88    88-88    88-88    88-88◊   88-96    88-96
         88-96    88-96◊   88-104   88-104   88-104   88-104◊  88-111   88-111   88-111   88-111◊  98-36    98-36    98-36    98-36◊
         117-79   117-79◊  118-40   118-40◊  127-34◊  128-70   128-70   128-70   128-70◊  128-95   128-95   128-95   128-95◊  129-36
         129-36   129-36   129-36◊  129-119  129-119  129-119  129-119◊ 130-44   130-44   130-44   130-44◊  130-54   130-54   130-54
         130-54◊  132-34   132-34   132-34   132-34◊  134-13◊  135-18   135-18◊  136-11   136-11◊  137-7◊   137-11◊  137-72◊  137-78◊
         138-7◊   139-40   139-40   139-40   139-40◊  143-28◊  144-21   144-21◊  144-23   144-23◊  144-25   144-25◊  144-27   144-27◊
         144-29   144-29◊  146-3    146-3◊

MANUAL   124-7    137-9    138-34
MEMORY   120-34   123-18
MULT     1-H59◊   92-22
MULTCM   1-G34◊   59-52    60-13    69-31    72-8     74-27    75-28    75-44    77-48    78-45    78-59    78-81    78-91    80-37
         81-32    81-40    83-35    94-49    98-42
OPEN     97-28
OR       1-D82◊
PNT      1-F65◊   56-25    56-29    56-34
PNT...   1-E94◊   36-12    36-13    36-27    37-11    37-36    37-59    37-80    38-11    38-32    39-11    39-12    40-17    40-25
         46-44    46-45    48-20    49-17    49-18    50-21    50-32    50-34    50-39    53-32    53-33    53-34    53-35    53-45
```

	53-46	53-57	53-58	54-7	54-8	54-17	54-18	55-12	55-13	55-22	55-23	56-25	56-29	56-34
	59-58	59-62	60-8	61-24	62-17	66-24	66-49	98-21	98-56	98-57	98-67	98-75	98-82	101-44
	103-26	117-42	117-45	117-67	117-68	117-73	117-78	118-7	118-15	118-24	118-33	121-12	123-8	135-13
	135-16	137-13	137-56	138-9	138-10	138-11	138-32	142-36	143-3					
PNTB	1-F38◊	36-12	36-13	36-27	37-11	37-36	37-59	37-80	38-11	38-32	39-11	39-12	40-25	46-44
	46-45	53-32	53-33	53-34	53-35	53-45	53-46	53-57	53-58	54-7	54-8	54-17	54-18	55-12
	55-13	55-22	55-23	59-58	59-62	60-8	61-24	62-17	66-24	98-21	98-56	98-57	98-67	98-75
	98-82	117-67	117-68	117-73	117-78	118-7	118-15	118-24	118-33	138-10	138-11	138-32	142-36	143-3
PNTF	1-F29◊	66-49	101-44	103-26	117-42	117-45	121-12	123-8	135-13	135-16	137-13	137-56	138-9	
PNTS	1-F56◊													
PNTX	1-F47◊	40-17	48-20	49-17	49-18	50-21	50-32	50-34	50-39					
POINTE	1-I23													
POP	1-D52◊	36-15	36-33	37-17	37-42	37-65	37-86	38-17	38-38	39-14	43-22	45-38	46-60	46-67
	46-74	49-20	51-14	51-19	52-43	52-51	52-58	55-26	56-30	56-35	57-37	59-36	66-55	71-18
	78-33	78-36	78-69	78-72	84-61	84-73	85-74	85-86	89-53	90-35	90-45	91-47	91-55	91-62
	93-30	94-42	94-43	95-47	97-59	99-16	100-24	103-27	106-32	108-32	108-41	109-20	110-20	111-40
	112-17	113-29	114-34	115-41	116-24	116-29	117-52	128-81	128-105	128-111	129-59	130-26	132-23	
PRINT	1-E77◊	56-28	56-33	56-36	102-8	102-19	103-11	103-33	108-16	108-38	134-10			
PRINTB	97-85	99-20												
PRINTF	99-18													
PRINTS	99-24													
PRINTX	99-22													
PUSH	1-D41◊	36-9	36-25	37-9	37-30	37-55	37-78	38-9	38-30	39-9	45-14	46-26	47-32	52-21
	53-23	56-16	56-23	57-34	59-32	66-21	71-12	78-29	78-65	84-24	85-24	89-33	90-21	90-28
	91-20	93-23	94-31	95-22	97-24	99-9	100-17	100-22	106-14	107-17	107-23	107-41	109-12	110-12
	111-14	112-11	113-17	114-20	115-15	116-11	116-18	117-17	117-29	128-28	128-79	129-16	130-19	132-18
READBU	120-37													
READEF	122-5	122-10	122-15											
SETCMD	1-F91◊	58-4	58-38	59-4	73-30	79-38	79-59	82-38	84-32	85-31	85-50	139-15		
SETPRI	127-30													
SETVEC	45-18	57-44	121-19	129-27										
STARS	1-C77◊	120-8	120-21											
SVC	1-I01◊	1-I02												
TABLE	1-E30◊	14-22	15-26	144-13										
XFER	36-35◊	37-19◊	37-44◊	37-67◊	37-88◊	38-19◊	38-40◊	39-16◊	117-79◊	118-40◊	127-34◊	134-13◊	135-18◊	136-11◊
	137-7◊	137-11◊	137-72◊	137-78◊	138-7◊	143-28◊								