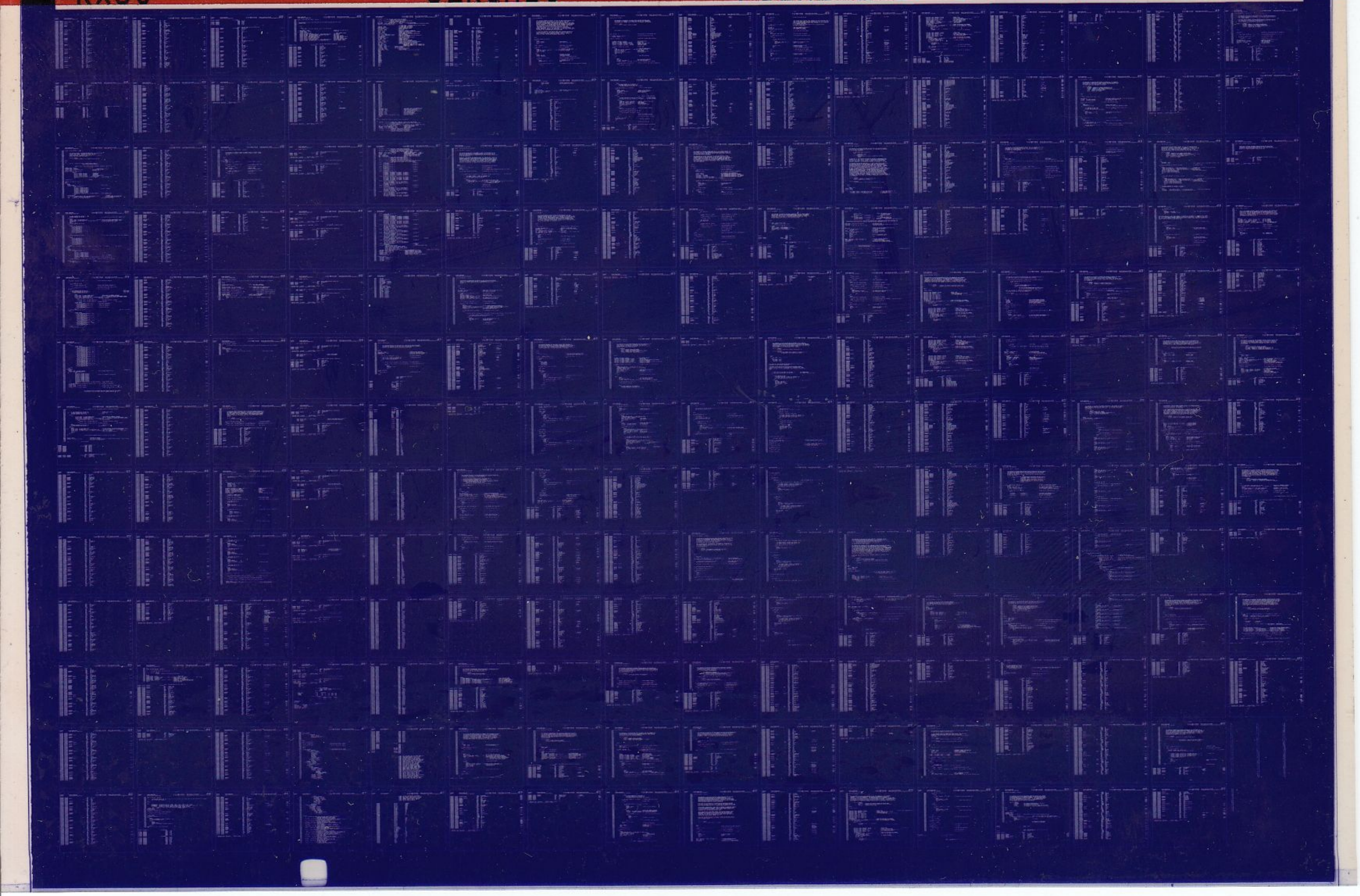
RQDX1.RD51 RQDX1 EXERCISER AH-T399B-MC OCT 1983
RX50 CZRQABO FICHE 1 OF 3 COPYRIGHT© 1983
MADE IN USA

RQDX1,RD51 RQDX1 EXERCISER AH-T399B-MC OCT 1983
RX50 CZRQABO FICHE 2 OF 3 COPYRIGHT© 1983
MADE IN USA



RQDX1,RD51 RQDX1 EXERCISER AH-T399B-MC OCT 1983
RX50 CZRQABO FICHE 3 OF 3 COPYRIGHT© 1983
MADE IN USA

```
;         0001  module ZRQAM1 (
;         0002
;         0003  %title 'RD/RX EXERCISER'
;         0004                  ident = 'V01.2',
;         0005                  addressing_mode (absolute),
;         0006                  environment (noeis)
;         0007                  ) =
;         0008
;         0009  begin
;         0010
;         0011
;       C 0012  %(
;       C 0013                                IDENTIFICATION
;       C 0014                                --------------
;       C 0015
;       C 0016          PRODUCT CODE:         AC-T398B-MC
;       C 0017
;       C 0018          PRODUCT NAME:         CZRQAB0 RQDX1 EXERCISER
;       C 0019
;       C 0020          PRODUCT DATE:         22 JUL 1983
;       C 0021
;       C 0022          MAINTAINER:           DIAGNOSTIC ENGINEERING
;       C 0023
;       C 0024          AUTHOR:               RAVINDER K. KARWAN
;       C 0025
;       C 0026
;       C 0027          COPYRIGHT (C) 1983
;       C 0028
;       C 0029          DIGITAL EQUIPMENT CORPOTRATION, MAYNARD, MASSACHUSETTS  01754
;       C 0030
;       C 0031          THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;       C 0032          COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;       C 0033          ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;       C 0034          MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;       C 0035          EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;       C 0036          TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;       C 0037          REMAIN IN DEC.
;       C 0038
;       C 0039          THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE
;       C 0040          AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;       C 0041          CORPORATION.
;       C 0042
;       C 0043          DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;       C 0044          SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;       C 0045
;       C 0046
;       C 0047          THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:
;       C 0048
;       C 0049          DIGITAL         PDP             UNIBUS          MASSBUS
;       C 0050          DEC             DECUS           DECTAPE
```

```
;       C 0051
;       C 0052
;       C 0053
;       C 0054
;       C 0055
;       C 0056
;       C 0057
;       C 0058                  TABLE OF CONTENTS
;       C 0059
;       C 0060
;       C 0061
;       C 0062          1.0     GENERAL INFORMATION
;       C 0063          1.1     PROGRAM ABSTRACT
;       C 0064          1.2     SYSTEM REQUIREMENTS
;       C 0065          1.2.1   HARDWARE REQUIREMENTS
;       C 0066          1.2.2   SOFTWARE REQUIREMENTS
;       C 0067          1.3     RELATED DOCUMENTS AND STANDARDS
;       C 0068          1.4     DIAGNOSTIC HIERARCHY PREREQUISITES
;       C 0069          1.5     ASSUMPTIONS
;       C 0070          1.6     MEMORY MAP
;       C 0071
;       C 0072          2.0     OPERATING INSTRUCTIONS
;       C 0073          2.1     HARDWARE QUESTIONS
;       C 0074          2.2     SOFTWARE QUESTIONS
;       C 0075
;       C 0076          3.0     ERROR TYPES
;       C 0077          3.1     ERROR INFORMATION
;       C 0078          3.2     INITIALIZATION ERRORS
;       C 0079          3.3     EXERCISER ERRORS
;       C 0080          3.4     ERROR LOG MESSAGES
;       C 0081          3.5     MSCP ERRORS
;       C 0082          3.6     SAMPLE MSCP ERROR STATEMENT
;       C 0083          3.7     DUP ERRORS
;       C 0084          3.8     SAMPLE DUP ERROR STATEMENT
;       C 0085          4.0     PERFORMANCE AND PROGRESS REPORTS
;       C 0086
;       C 0087          5.0     TEST SUMMARY
;       C 0088          5.1     INITIALIZATION SUBTEST
;       C 0089          5.2     EXERCISER
;       C 0090          5.3     DROP UNIT SUMMARY
;       C 0091
;       C 0092          6.0   ERROR LIST
;       C 0093
;       C 0094          7.0    DATA PATTERNS
```

```
C 0095
C 0096
C 0097          1.0   GENERAL INFORMATION
C 0098                ---------------------
C 0099
C 0100
C 0101          1.1   PROGRAM ABSTRACT
C 0102                ----------------
C 0103
C 0104                This program  will functionally verify  and exercise RQDX1
C 0105                Controller/Disk Drive subsystems. It is designed to verify
C 0106                that the subsystem  is functioning correctly and operating
C 0107                within design specifications.
C 0108
C 0109
C 0110
C 0111          1.2   SYSTEM REQUIREMENTS
C 0112                ---------------------
C 0113
C 0114          1.2.1 HARDWARE REQUIREMENTS
C 0115                ----------------------
C 0116
C 0117                LSI - 11/23 processor with 28K or more of memory,  console
C 0118                device (eg. VT100) and RQDX1 CONTROLLER board and attached
C 0119                RD-51 WINCHESTER drive(s) and RX-50 FLOPPY drive(s)
C 0120
C 0121          1.2.2 SOFTWARE REQUIREMENTS
C 0122                ----------------------
C 0123
C 0124                This  diagnostic  is  designed  to run with the Diagnostic
C 0125                Supervisor as described in paragraph 2.0.
C 0126
C 0127
C 0128          1.3   RELATED DOCUMENTS AND STANDARDS
C 0129                --------------------------------
C 0130
C 0131                XXDP+ SUPERVISOR/USERS MANUAL   CHQUS
C 0132                UQSSP UNIBUS/Q-BUS STORAGE SYSTEMS PORT
C 0133                MSCP  MASS STORAGE SYSTEM PROTOCOL
C 0134                DUP   DIAGNOSTIC/UTILITIES PROTOCOL
C 0135
C 0136          1.4   DIAGNOSTIC HIERARCY PREREQUISITES
C 0137                ---------------------------------
C 0138
C 0139                NONE
C 0140
C 0141
C 0142          1.5   ASSUMPTIONS
C 0143                -----------
C 0144
C 0145                The hardware,  other than  the  subsystem  being  tested is
C 0146                assumed to work properly.  False errors  may be reported if
C 0147                the processor, memory, etc., do not function properly.
```

```
C 0148
C 0149
C 0150          1.6     MEMORY MAP
C 0151                  ----------
C 0152
C 0153
C 0154          Memory layout on 28k machine - XXDP environment
C 0155          ------------------------------------------------
C 0156
C 0157                                                  ADDRESS
C 0158
C 0159                  +-----------------------+       160000
C 0160                  |                       |
C 0161                  |    XXDP+ MONITOR      |
C 0162                  |                       |
C 0163                  +-----------------------+
C 0164                  |                       |
C 0165                  |  XXDP+   TTY I/O AREA |
C 0166                  |                       |
C 0167                  +-----------------------+
C 0168                  |                       |
C 0169                  |                       |
C 0170                  |         DRS           |
C 0171                  |                       |
C 0172                  |                       |
C 0173                  +-----------------------+       121000
C 0174                  |                       |
C 0175                  |      EXERCISER        |
C 0176                  |                       |
C 0177                  +-----------------------+       2000
C 0178                  |                       |
C 0179                  |       STACKS          |
C 0180                  |                       |
C 0181                  +-----------------------+       400
C 0182                  |                       |
C 0183                  |     VECTOR AREA       |
C 0184                  |                       |
C 0185                  +-----------------------+       0
C 0186
C 0187
C 0188          In a machine  with more memory,  free space will ocuur
C 0189          between the exerciser and the DRS.
```

```
C 0190
C 0191
C 0192          2.0  OPERATING INSTRUCTIONS
C 0193               ----------------------
C 0194
C 0195
C 0196          This is a Rev C  Supervisor  Diagnostic:  for operating
C 0197          instructions,  please see chapter 5 of XXDP+ operator's
C 0198          manual.  They are no longer  included in the diagnostic
C 0199          because  it is desired  that a change in those instruc-
C 0200          tions not require a re-assembly of all Supervisor Diag-
C 0201          nostics.
C 0202
C 0203
C 0204          2.1  HARDWARE QUESTIONS
C 0205               ------------------
C 0206
C 0207          The  following series  of questions  comprise the para-
C 0208          meters necessary to identify each disk subsystem.
C 0209
C 0210
C 0211          Hardware Configuration Questions
C 0212          --------------------------------
C 0213
C 0214          The  program  will  ask  the  following  questions  in
C 0215          response to a START command (non-script).
C 0216
C 0217          1.  CHANGE HW (L) Y ?
C 0218
C 0219          Answer NO to use the pre-built answers for all  hardware
C 0220          questions.   This  program will be released pre-built to
C 0221          test three unit with default answers  shown below.   The
C 0222          pre-built  answers  may  be changed at any time with the
C 0223          setup utility.  Answer YES if you want all the  hardware
C 0224          questions to be asked.
C 0225
C 0226          2.  NUMBER OF UNITS (D) ?
C 0227
C 0228          No default.  Answer with the number of  disk drive units
C 0229          to be exercised or tested.   This  answer will determine
C 0230          how  many  times  the  following questions are asked.  A
C 0231          range  of 1 to 4 units  may be specified.  A unit number
C 0232          will be assigned  sequentially from 0  by the Diagnostic
C 0233          supervisor for each unit.
C 0234
C 0235          3.  IP ADDRESS (O) 172150 ?
C 0236
C 0237          Enter the address of the IP register of one R0DX1 as ad-
C 0238          dressed  by  the processor with memory management turned
C 0239          off.  The program expects an even 16-bit address in  the
C 0240          range of 160000 to 177774.  172150 is the default.
```

```
C 0241
C 0242
C 0243          4.  VECTOR ADDRESS (O) 154 ?
C 0244
C 0245              Answer  with the  interrupt vector of same  RQDX1 in  the
C 0246          above  question.   A vector  address in the range of 4 to
C 0247          774 may be specified.  154 is the default.
C 0248
C 0249          5.  BR LEVEL (D) 4 ?
C 0250
C 0251              Answer with the bus request interrupt level used by  the
C 0252          above RQDX1. Levels 4 through 7 are acceptable. 4 is the
C 0253          default.
C 0254
C 0255          6.  RQDX1 DRIVE NUMBER (D) 0 ?
C 0256
C 0257              Enter the  logical unit  number for one drive associated
C 0258          with the  IP address  above.  Drive  numbers are  in the
C 0259          range  of 0  through 3.  The  number  entered  here must
C 0260          match the unit plug on the front panel of the device.  0
C 0261          is the default answer.
C 0262
C 0263          7.  ENTER UNIT TYPE:  RD51 - YES  OR  RX50 - NO ?
C 0264
C 0265              Question  will  be  asked to  determine the type of disk
C 0266          that this particular unit is; either RD51, or RX50.  The
C 0267          software will  not check for the correctness of the con-
C 0268          figuration  as indicated by the answers to the UNIT TYPE
C 0269          question.
C 0270
C 0271          8.  ALSO EXERCISE DIAGNOSTIC AREA (-NON-CUSTOMER AREA) ?
C 0272
C 0273              A "Yes" answer to this question will turn on the DUP
C 0274          (Diagnostic/Utilities Protocol) Exerciser.  The DUP Ex-
C 0275          erciser will read the DBNs (Diagnostic Blocks) in a
C 0276          sequential order from 0 to octal (217). DBNs can only
C 0277          be accessed throw DUP protocol making the blocks unreach-
C 0278          able throw normal customer use.  The DUP Exerciser is
C 0279          contained in the middle of the MSCP (Mass Storage Control
C 0280          Protocol) Exerciser which reads LBNs (Logical blocks).
C 0281          So physically a large amount of LBNs will be READ and/or
C 0282          WRITTEN then the DUP Exerciser will turn on and READ
C 0283          and/or WRITE a few DBNs. The DUP Exerciser will then
C 0284          reinitialize the controller so the MSCP Exerciser can
C 0285          continue where it left off. The processes of reinitial-
C 0286          izing takes a few seconds.
C 0287
C 0288          9.  WRITE ON DIAGNOSTIC AREA?
C 0289
C 0290              A "yes" answer to this statement will allow the DUP Ex-
C 0291          erciser to WRITE/READ and compare the DBNs (Diagnostic
C 0292          Blocks).  Where a "NO" answer will allow the program to
C 0293          READ only the DBNs.
```

```
;       C 0294
;       C 0295                  NOTE  **  IF yes ANSWER, DATA ON DBNs WILL BE DESTROYED.
```

```
C 0296
C 0297
C 0298          10.  STARTING LBN (D) 0 ?
C 0299
C 0300              Enter the starting logical block number of the  customer
C 0301              data area that you are going to test. LBNs  range from 0
C 0302              to  21599  (RD51),  or  0 to 799  (RX50),  with 0 as the
C 0303              default.
C 0304
C 0305          11.  ENDING LBN (MAX - RX50: 799, RD51: 21599) (D) 21599 ?
C 0306
C 0307              Answer this  question for the last customer LBN you wish
C 0308              to test.  LBNs range from 0 to 21599 (RD51), or 0 to 799
C 0309              (RX50), with 21599 (or 799) as the default.
C 0310
C 0311              NOTE: The two previous  questions are generally Software
C 0312                    Parameter questions,  but since two different disk
C 0313                    devices exist on the RQDX1 subsystem, this becomes
C 0314                    a unit by unit question. It is possible to specify
C 0315                    an  LBN which  is too large  since we  are dealing
C 0316                    with different devices. The program will check for
C 0317                    block number bounds and if they are exceeded, will
C 0318                    assign the maximum bound for that device.
C 0319
C 0320
C 0321          12.  EXERCISE ON CUSTOMER DATA AREA ON THIS DISK UNIT (L) ?
C 0322
C 0323              Answering YES will destroy any customer data that is  on
C 0324              the  disk; therefore, the following warning message will
C 0325              appear, followed by a confirmation prompt:
C 0326
C 0327              ** WARNING - CUSTOMER DATA AREA WILL BE OVERWRITTEN! ...
C 0328               CONFIRM (L) ?
C 0329
C 0330              This question will default to NO if the operator has de-
C 0331              cided  to  bypass  the  hardware  questions.  Otherwise,
C 0332              there is no default.
```

2.2 SOFTWARE QUESTIONS

Software Parameter Questions

The program will ask the following questions in response to the START, RESTART, and CONTINUE commands.

1. CHANGE SW (L) Y ?

   Answer NO to bypass the following questions in this section. This question should normally be answered NO when the Exerciser is first run. A NO answer will cause the Exerciser to select the default parameters shown with each question below. Then, depending on the errors detected, it may be desirable to change this answer to YES to alter the test parameters and further isolate the problem.

2. HARD ERROR LIMIT (D) 32 ?

   Enter the number of hard errors allowed before a unit is dropped from testing. A number in the range of 1 to 65535 will be accepted.

3. TRANSFER LIMIT IN MEGABYTES (0 FOR QUICK PASS) (D) 0 ?

   When the specified number of bytes have been transferred to/from a unit, the unit will be dropped from testing. When all units are dropped, an end-of-pass will be indicated. This is the method used to determine how long the Exerciser is to run.

   The only other way the Exerciser will declare end-of-pass is if all units are dropped because the error limit on each is exceeded. However, the operator can always abort the program at any time by typing CONTROL-C.

4. PERCENTAGE OF RD51 OPERATIONS OUT OF TOTAL OPERATONS (D) 99 ?

   In order to maintain typical usage for the devices of this exercise, a certain percentage of operations must be directed to the RD51s (the rest go to the RX50s). It turns out that this percentage is very high (as indicated by the 99% figure given as the default). It may be desirable in some cases to direct more activity to the RX50s. This is easily done by directing a smaller percentage of the operations to the RD51s. The numbers associated with usage are adjusted internally by the program according to device type and percentage.

```
C 0386
C 0387
C 0388          5.  NUMBER OF DBNs READ AT ONE TIME (18) ?
C 0389
C 0390              This variable adjusts the amount of DBNs read in one pass
C 0391              of the DUP Exerciser.  The DUP Exerciser has to reinit
C 0392              the controller every time it goes back into the MSCP Exer-
C 0393              ciser.  This Hard reinitialization takes a few seconds.
C 0394              The ratio of LBN tranfers to DBN tranfers DOESN'T change
C 0395              only the amount of DBNs read on one pass.  So the higher
C 0396              this number the less reinitializing the program does and
C 0397              the less time to run. The lower this number the more
C 0398              reinitializations the longer amount of time to tranfer the
C 0399              same amount LBNs and DBNs.
C 0400
C 0401          6.  CLEAR STATISTICAL TABLES AFTER PRINTING (L) N ?
C 0402
C 0403              Answering YES causes the statistical fields to be clear-
C 0404              ed to zero after the report is printed (either at end of
C 0405              pass,  or at  operator  request).  Otherwise, cumulative
C 0406              totals are maintained.
C 0407
C 0408          7.  RANDOM SEEK MODE (L) Y ?
C 0409
C 0410              Answer YES to cause block numbers to be chosen randomly.
C 0411              Answer  NO to cause block numbers to be selected sequen-
C 0412              tially.
C 0413
C 0414          8.  UNITS TO BE SELCTED AT RANDOM (NO, IMPLIES SEQUENTIAL) (L) N ?
C 0415
C 0416              This  question is  optionally asked if the answer to the
C 0417              previous  question is N[o].  The  selection of units for
C 0418              sequential  operations is affected by the answer to this
C 0419              question.  If the default  answer is chosen (N[o]), then
C 0420              units  shall be  selected in a  predetermined manner  in
C 0421              accordance  with the typical  seek time margins for each
C 0422              device. If the alternate answer is chosen (Y[es]),  then
C 0423              the  units will be chosen  at random  in accordance with
C 0424              the percentages specified in Software question 4.
C 0425
```

```
C 0426
C 0427
C 0428          9.  READ-COMPARES PERFORMED AT THE CONTROLLER (L) N ?
C 0429
C 0430              Answering YES causes all read commands to include the
C 0431          "compare" modifier.  This essentially forces the con-
C 0432          troller to perform two read operations on the same disk
C 0433          address, and to compare the results.
C 0434
C 0435              The following message will appear after the operator has
C 0436          answered this question:
C 0437
C 0438              THE REMAINING QUESTIONS ONLY APPLY TO UNPROTECTED DISK UNITS.
C 0439
C 0440          10.  WRITE-COMPARES PERFORMED AT THE CONTROLLER (L) N ?
C 0441
C 0442              Answering YES causes all write I/O requests to be
C 0443          changed to write-compare.  After each write, the con-
C 0444          troller will read the data and compare it to data
C 0445          re-obtained from the host.
C 0446
C 0447          11.  CHECK ALL WRITES AT HOST BY READING (L) N ?
C 0448
C 0449              This question will only be asked if the previous ques-
C 0450          tion was answered NO.  Answering YES causes all writes
C 0451          to be checked by the host by reading the data immediate-
C 0452          ly after the write operation.  This option consumes
C 0453          extra CPU time, and doubles the amount of storage re-
C 0454          quired for writes.  Therefore, it is only recommended
C 0455          when device write-compare operations are suspect.
C 0456
C 0457
C 0458          12. USER-DEFINED DATA PATTERN (L) N ?
C 0459
C 0460              An answer of YES allows the operator to define his/her
C 0461          own data pattern to be used in all write operations.  A
C 0462          NO answer will allow the operator to select a
C 0463          pre-defined data pattern in the next question.
C 0464
C 0465          13. SELECT PRE-DEFINED DATA PATTERN (0 FOR SEQUENTIAL SELEC-
C 0466          TION) (D) 0 ?
C 0467
C 0468              There are 21 predefined data patterns available, select-
C 0469          ed as 1 to 21 (see section 4.9).  A zero answer will
C 0470          cause patterns 1 to 21 to be sequentially selected for
C 0471          each write.  (Note that pattern 1 consists entirely of
C 0472          random numbers).
C 0473
```

```
;       C 0474
;       C 0475
;       C 0476          14. NUMBER OF WORDS IN DATA PATTERN (16 MAXIMUM)  (D)  16  ?
;       C 0477              PATTERN VALUES (O) ?
;       C 0478
;       C 0479              These questions will only be asked if the  operator  has
;       C 0480              decided  to define his/her own data pattern.  The actual
;       C 0481              bit patterns will be entered as octal (PDP-11).
```

```
C 0482
C 0483
C 0484          3.0  ERROR TYPES
C 0485               ----------
C 0486
C 0487          This program has four types of error  classifications;
C 0488          system fatal, device fatal, hard and soft.
C 0489
C 0490          SYSTEM FATAL ERRORS
C 0491          -------------------
C 0492
C 0493          System fatal errors are used to indicate that an error
C 0494          was detected by the  Diagnostic Supervisor in relation
C 0495          to loading/controlling the diagnostic process.
C 0496
C 0497          The content  of each  error is  such that it should be
C 0498          self explanatory.  However,  the messages utilize some
C 0499          terms that are specific to the disk subsystem, and may
C 0500          require some getting use to.
C 0501
C 0502
C 0503          DEVICE FATAL ERRORS
C 0504          -------------------
C 0505
C 0506          Device fatal errors are a result of:
C 0507
C 0508          an error that is  considered fatal to the device,  but
C 0509          testing will continue.
C 0510
C 0511
C 0512          HARD ERRORS
C 0513          -----------
C 0514
C 0515          Hard errors are a result of:
C 0516
C 0517               1.  retries of a soft error or *
C 0518               2.  a non-recoverable error
C 0519               3.  a soft error if retrys are not set.
C 0520
C 0521               *  Note: Retries are executed in the controller
C 0522
C 0523          SOFT ERRORS
C 0524          -----------
C 0525
C 0526          Soft errors  are media related errors.  All soft errors
C 0527          will be retried by the controller.
C 0528
C 0529          Note: Soft errors are retrieved from the controller via
C 0530          the error log  capabilities of  MSCP.
```

```
C 0531
C 0532
C 0533          3.1     ERROR INFORMATION
C 0534                  ----------------
C 0535
C 0536          All general error messages will include the type of error
C 0537          (system-fatal, device-fatal, hard, soft) and a unit number.
C 0538          If the error applies to a controller, then only the first
C 0539          unit number of the controller will be given. (The user will
C 0540          know the other unit numbers when subsequent "drop unit"
C 0541          messages are printed).
C 0542
C 0543              Basic error messages provide more details about the
C 0544          error. The Exerciser will print all basic error messages,
C 0545          along with the disk address, if applicable. In some
C 0546          cases where a device-fatal error applies to a controller,
C 0547          the controller's IP address will be printed.
C 0548
C 0549              Extended error messages will be used to print the
C 0550          relevant fields of command and end message packets, status
C 0551          codes, SA register contents, and error log messages. All
C 0552          values will be in octal (PDP-11).
C 0553
C 0554              The error messages in this section do not include
C 0555          errors detected and printed by the Diagnostic Supervisor.
C 0556
C 0557
C 0558
C 0559          3.2 INITIALIZATION ERRORS
C 0560              ---------------------
C 0561
C 0562              Two kinds of errors will be reported to the operator
C 0563          during the Initialization Test. The System-fatal error is -
C 0564          too many units specified. A system-fatal error will cause
C 0565          the Exerciser to abort.
C 0566
C 0567              Device-fatal errors only affect the unit(s) involved.
C 0568          Testing will continue on all other units. This class of
C 0569          errors includes, but is not limited to, the following:
C 0570
C 0571              1. Register Existence Test failure (no device present)
C 0572
C 0573              2. Vector Test failure
C 0574
C 0575              3. BR Level Test failure
C 0576
C 0577              4. Initialization sequence failure
C 0578
C 0579              5. Online failed
C 0580
C 0581              6. Access failed
```

C 0582
C 0583
C 0584 ## 3.3 EXERCISER ERRORS
C 0585
C 0586
C 0587 Most errors reported during this test will originate
C 0588 from MSCP end message packets. The status code field will
C 0589 be converted to text and printed as part of a basic error
C 0590 message. Any subcode value will follow if extended error
C 0591 messages are enabled.
C 0592
C 0593 The following list represents some of the error conditions
C 0594 reported via MSCP:
C 0595
C 0596
C 0597 1. Disk unit went offline (a sub-code may follow
C 0598 detailing the reason)
C 0599
C 0600 2. Compare error
C 0601
C 0602 3. Data error (a sub-code may follow)
C 0603
C 0604 4. Drive error (a sub-code may follow)
C 0605
C 0606 5. Host buffer access error
C 0607
C 0608 6. Media format error (a sub-code may follow)
C 0609
C 0610
C 0611
C 0612
C 0613
C 0614 ## 3.4 ERROR LOG MESSAGES
C 0615
C 0616
C 0617 The contents of the error-log messages received from the
C 0618 controller are printed as received, and should be deciphered
C 0619 using the MSCP specs.
C 0620
C 0621
C 0622 ## 3.5 MSCP ERRORS
C 0623
C 0624
C 0625 An MSCP error occurs when the host receives an Invalid
C 0626 Command End Message from the RQDX1. In such cases, the host
C 0627 will print out the erroneous command followed by the reason
C 0628 for the error. If extended printouts are enabled, then the
C 0629 entire contents of the end message will be displayed in octal
C 0630 without interpretation of the data.

```
C 0631
C 0632
C 0633          3.6   SAMPLE MSCP ERROR STATEMENT
C 0634                ---------------------------
C 0635
C 0636
C 0637                The errors listed by the exerciser are ussually very des-
C 0638          criptive and  are self explanatory.  The  following is an example
C 0639          error  statement.  This  error  statement  is  the extented error
C 0640          message.
C 0641
C 0642
C 0643          (example)                              (comments)
C 0644
C 0645          * DISK: XXX                            !DISK UNIT NUMBER
C 0646          CRN: XXXXX                             !CONTROLLER PACKET NUMBER
C 0647          MESSAGE TYPE: - SEQUENTIAL             !THIS IS THE PORT MESSAGE TYPE
C 0648          COMMAND:-MSCP-READ-COMPARE             !CONNECTION ID AND COMMAND AND MODIFIER GIVEN TO DRIVE
C 0649          STATUS CODE:  UNIT OFFLINE             ! STATUS CODE OF COMMAND
C 0650          STATUS SUB-CODE:  NO VOLUME MOUNTED OR DRIVE DISABLED BY SWITCH ! SUB CODE
C 0651          BYTE COUNT IN COMMAND XXXXX            !NUMBER OF BYTES WANTED TO READ
C 0652          ACTUAL # OF BYTES TRANSFERED XXXXX !NUMBER OF BYTES ACTUALLY READ
C 0653          I/O BUFFER ADDRESS (32 BITS)        XXXXXX XXXXXX
C 0654          LBN: XXX                               ! LBN WANTED TO READ
C 0655          END MESSAGE FLAGS:                     ! THERE ARE NO END FLAGS FOR THIS ERROR
C 0656
C 0657
C 0658
C 0659                The  status  code in  an end  messages is  broken into two
C 0660          pieces. The first 5 bits represent the major status which is given
C 0661          by the "invalid command" message.  The 11 remaining bits represent
C 0662          the sub-code,  which tells in greater detail the error in the con-
C 0663          troller.  The LBN is the logical  block on the disk the controller
C 0664          was trying to read.  The byte count refers  to the number of bytes
C 0665          the controller was going to read off the LBN. The actual number of
C 0666          bytes  transfered refers  to the  number of bytes  read before the
C 0667          error.  The end message flags give any flags that might have been
C 0668          set by the controller. It is pretty apparent that this error was
C 0669          caused by something physically switching the disk offline.
C 0670
C 0671
C 0672          3.7   DUP ERRORS
C 0673                ----------
C 0674
C 0675                A DUP error occurs when the host receives an  Invalid
C 0676          Command End Message from the RQDX1.  In such cases, the host
C 0677          will print out the erroneous command followed by the  reason
C 0678          for the error.
C 0679                There are two major places where errors in  DUP will
C 0680          occur. The first being in the status code much the same as MSCP,
C 0681          and the second in the DUP I/O Buffer. Using the DUP sub-protocol
C 0682          (using Receive/Send commands to communicate with controller local
C 0683          programs (p. 25 of DUP.V05)) the controller may send an error
```

```
;       C 0684          message to the host by writting in the DUP I/O Buffer.
```

```
C 0685
C 0686
C 0687          3.8   SAMPLE DUP ERROR STATEMENT
C 0688                ----------------------------
C 0689
C 0690
C 0691                The errors listed by the exerciser are ussually very des-
C 0692          criptive and  are self explanatory.  The  following is an example
C 0693          error  statement.  This  error  statement  is  the extented error
C 0694          message.
C 0695
C 0696
C 0697          (example)                              (comments)
C 0698
C 0699          * DISK: XXX                            !DISK UNIT NUMBER
C 0700          CRN: XXXXX                             !CONTROLLER PACKET NUMBER
C 0701          MESSAGE TYPE: - SEQUENTIAL             !THIS IS THE PORT MESSAGE TYPE
C 0702          COMMAND:-DUP-RECEIVE DATA              !CONNECTION ID AND COMMAND GIVEN TO DRIVE
C 0703          STATUS CODE: -SUCCESS                  ! STATUS CODE OF COMMAND
C 0704          ACTUAL # OF BYTES TRANSFERED XXXXX !NUMBER OF BYTES ACTUALLY READ
C 0705          I/O BUFFER ADDRESS (32 BITS)    XXXXXX XXXXXX
C 0706          MESSAGE TYPE:                          ! TYPE OF COMMUNICATION
C 0707                ** FATAL ERROR
C 0708          MESSAGE NUMBER:                        ! MESSAGE TOBE GIVEN OR RESULT EXPECTED
C 0709           -- SUCCESS/FAILURE CODE
C 0710          MESSAGE ERROR CODE:                    ! ERROR LISTING
C 0711           - ILLEGAL UNIT NUMBER
C 0712          DBN: XXX
C 0713
C 0714                The DUP Error messages are almost like the MSCP messages
C 0715          but they contain an extra three classifications.  These are the
C 0716          MESSAGE TYPE, MESSAGE NUMBER, MESSAGE ERROR CODE.  These are part
C 0717          of a DUP sub-protocal for communicating with a controller local
C 0718          program.  The I/O buffer address contains the address of the DUP
C 0719          I/O Buffer which contains the MESSAGE information.
C 0720                The  status  code in  this end  message is succesful which
C 0721          means the controller was OK but it did not understand the host
C 0722          message or something on the host side caused the controller to error
C 0723          while running the controller local program.  This error was produced
C 0724          by an ILLEGAL UNIT NUMBER.  The disk went off line.
C 0725
```

```
C 0726
C 0727
C 0728          4.0   PERFORMANCE AND PROGRESS REPORTS
C 0729                ----------------------------------
C 0730
C 0731
C 0732              A summary report is printed at the end of  each  pass of
C 0733          the  Exerciser or upon demand by the operator. The fields may
C 0734          be cleared to zero after the  report is printed  depending on
C 0735          the operator's  response to this option in the software ques-
C 0736          tions. Any units added to the test cycle will also begin with
C 0737          cleared statistics.
C 0738              There are two basic listings one for LBNs and one for
C 0739          DBNs.  The DBN listing will only contain RD51s for the simple
C 0740          reason that they contain DBNs and RX50s do not.  The DBNs
C 0741          are READ and WRITTEN by blocks instead of by bytes. All units
C 0742          contain LBNs. The errors for each unit will be listed next to
C 0743          the LBNs.
C 0744              Errors are grouped into two  basic catagories:  hard and
C 0745          soft.  Each is sub divided into four more catagories, depend-
C 0746          ing on the most probable classification for that error.
C 0747
C 0748               The sub catagories are:
C 0749
C 0750          1. disk related errors
C 0751
C 0752          2. seek (or format) related errors
C 0753
C 0754          3. controller or drive related errors
C 0755
C 0756          4. host (the CPU) related errors.
C 0757
C 0758               All numeric values are in decimal radix.
C 0759
C 0760          UNT        # OF BYTS  # OF    BYTES    -- HRD ERS --   -- SFT ERS --
C 0761           #  TYPE  READS READ WRITES WRITTEN DAT SEK DRV HST DAT SEK DRV HST
C 0762          --- ----- ----- ---- ------ ------- ---------------- ----------------
C 0763
C 0764           X  RD51  XXXX XXXX  XXXXX  XXXXXX  X   X   X   X   X   X   X   X
C 0765           .  RX50   ..   ..    ...    ....  .   .   .   .   .   .   .   .
C 0766           .  RX50   ..   ..    ...    ....  .   .   .   .   .   .   .   .
C 0767
C 0768          UNIT   DISK          # OF    # BLKS       # OF    # BLKS
C 0769           #      #    TYPE   READS    READ       WRITES  WRITTEN
C 0770          ----  ----- ------- -----  -------     -------  -------
C 0771            X     X   DBNRD51 XXXXX   XXXXX       XXXXX    XXXXX
C 0772
```

```
C 0773
C 0774
C 0775          5.0  TEST SUMMARY
C 0776               ------------
C 0777
C 0778          The  RQDX1  functional  tester and exerciser consists of
C 0779     two  parts,  the  initialization subtest  and the performance
C 0780     exerciser.  The operator is not able to select which of these
C 0781     two parts he/she wishes to run; they both must be executed.
C 0782
C 0783          5.1  INITIALIZATION SUBTEST
C 0784               ----------------------
C 0785
C 0786          The  purpose of  this subtest  is to verify the hardware
C 0787     configuration as specified by the operator, and to bring each
C 0788     unit online.  The Initialization  Subtest will always precede
C 0789     the execution of any other test.
C 0790
C 0791          First,  the presence of each device register will be ve-
C 0792     rified,  along with a check on  the BR level specified by the
C 0793     operator.  Then,  an  initialization  will be issued to  each
C 0794     controller configured for testing.  When  the initialization
C 0795     sequence has been completed, an attempt will be made to bring
C 0796     each unit online.  If this succeeds,  one  or two MSCP reads
C 0797     will be issued to the inner-most LBN of each selected disk to
C 0798     to ensure that each disk drive can seek and be read.
C 0799
C 0800          Any device-fatal or hard errors encountered  during this
C 0801     test  will  cause the  appropriate unit(s) to be dropped.  If
C 0802     basic error messages are enabled, then the program will print
C 0803     out the specific reason for dropping the unit(s). Henceforth,
C 0804     the  failed  unit(s) will  not be  tested unless the operator
C 0805     intervenes (adds unit(s) or restarts Exerciser).
C 0806
C 0807          Upon succesful completion of the Initialization Subtest,
C 0808     the program  will  begin executing the Exerciser.
C 0809
C 0810          5.2  EXERCISER
C 0811               ---------
C 0812
C 0813          The purpose of this subtest is to exercise the  disk  drives
C 0814     in a manner similar to the typical usage under standard oper-
C 0815     ating systems.  Execution of this test should give an indica-
C 0816     tion of the operating performance of the disk drive subunits.
C 0817     This test will utilitize random disk addresses,  random  word
C 0818     counts,  and data  patterns,  all subject  to the  limits and
C 0819     specifications made by the operator. All protected disks will
C 0820     be subject to read-only  operations,  while unprotected disks
C 0821     may be read or written, depending on the answers given to the
C 0822     software parameter questions.  End-of-pass  will be declared
C 0823     when the specified number  of bytes have been transferred for
C 0824     all the disks taken as a whole.
```

```
C 0825
C 0826
C 0827          If a read/write error occurs during this test, then the
C 0828      RQDX1 CONTROLLER will initiate an appropriate number of re-
C 0829      tries. If all retries fail, then a hard error will be repor-
C 0830      ted to the host, an error message will be diplayed on the
C 0831      console terminal and the error will be tallied for the sum-
C 0832      mary report. The unit will be dropped if the hard error count
C 0833      has exceeded the specified limit.
C 0834
C 0835          The Exerciser is actually two exercisers combined together.
C 0836      The main MSCP exerciser writes and reads LBNs while the less
C 0837      used DUP exerciser writes and reads DBNs. The DUP exerciser is
C 0838      used once per 25 LBN transfers or alot less than the MSCP
C 0839      exerciser. The two Exercisers use a some what different
C 0840      protocol to tranfer I/O. It is possible to go from MSCP
C 0841      protocol to DUP protocol without reinitializing the controller
C 0842      but impossible to go from DUP to MSCP protocol. Therfore after
C 0843      the DUP Exercise pass the controller is reinitialize and control
C 0844      given back to the MSCP Exerciser. The reinitialization process
C 0845      takes a few seconds. For this reason a variable is placed in
C 0846      the DUP Exerciser to allow it to transfer more than 1 DBN per pass.
C 0847      The variable,"X", is multiplied by 25 to give the amount of MSCP
C 0848      transfers before the DUP Exerciser can tranfer "X" amount of DBNs.
C 0849      The higher the variable the less reinits the controller must do.
C 0850
C 0851
C 0852  5.3 DROP UNIT SUMMARY
C 0853      ----------------
C 0854
C 0855          During the Initialization Subtest, individual units will
C 0856      be dropped from the test sequence if they are unable to be
C 0857      brought online or the operator specified device does not match
C 0858      the hardware.
C 0859
C 0860          During the Exercise, the program will drop a unit for one
C 0861      of three reasons. The normal path is for each unit to complete
C 0862      the transfer of N megabytes, where N is specified by the oper-
C 0863      ator during SW questioning and be soft-dropped. Otherwise, a
C 0864      unit will be hard-dropped if the number of hard errors encoun-
C 0865      tered exceeds the operator-specified limit, or if a fatal error
C 0866      is detected. Units hard-dropped may later be added to the test
C 0867      cycle. However, statistics for the hard-added unit will be
C 0868      cleared to zero; if a transfer limit was specified, in which
C 0869      case the unit was soft-dropped, the statistics may or may not
C 0870      be cleared depending on the operators answer to Software
C 0871      question 12.
```

6.0 ERROR CODES GENERATED BY ZRQA EXERCISER

SYSTEM FATAL ERRORS

1 More than 4 units specified

DEVICE FATAL ERRORS

10	Controller couldn't be addressed at the address given.	Wrong IP address selected
11	Controller didn't interrupt at the interrupt vector given.	Wrong vector address selected.
12	Controller didn't interrupt at the BR lelvel given.	Wrong BR level selected.
13	Init sequence failed.	Either one of the four initialization steps did not receive the correct response from the Controller, or one of the steps timed-out.
14	Fatal Controller error.	The error bit (bit 15) in the SA register was set.
15	Failed to bring unit on-line.	On-line response had an error code. (see also #s 22 and 23.)
16	Write protect conflict.	The unit was hardware write protected and write operations were requested on the unit.
17	Access to inner track failed.	Innermost track's header may be corrupted.
18	Unit went off-line.	---
20	Failed to send "Set Controller Characteristics" command.	Either the unit is off-line or the Diagnostic is corrupted because of any problems with its RAM.

```
C 0923
C 0924
C 0925          21  Controller  returned wrong  'end     Problem with the  Control-
C 0926              code'  for  the  'Set Controller      ler microcode or the port/
C 0927              Characteristics' command.             DMA intterface.
C 0928
C 0929          22  Failed to send 'On-line' command      Either  the  unit  is off-
C 0930                                                    line or  the diagnostic is
C 0931                                                    corrupted because  of  any
C 0932                                                    problems with its RAM.
C 0933
C 0934          23  Controller  returned  wrong 'end     Problem with the  Control-
C 0935              code' for the 'On-line' command.      ler's  microcode  or  the
C 0936                                                    port/DMA interface.
C 0937
C 0938          24 Device went to available state        Fault switch or door mechanism
C 0939
C 0940
C 0941          HARD ERRORS
C 0942          ~~~~~~~~~~~
C 0943
C 0944  MSCP ERRORS
C 0945
C 0946          31  Controller received  an  invalid     The  diagnostic is corrup-
C 0947              command.                              ted  because  of any prob-
C 0948                                                    lems  with  its  RAM,  or
C 0949                                                    there  is  a  problem with
C 0950                                                    the  Controller  microde
C 0951                                                    (RAM or ROM)  or  there is
C 0952                                                    problem with  the port/DMA
C 0953                                                    interface.
C 0954
C 0955          32  Command aborted by the  Control-     Command  timed-out  in  the
C 0956              ler.                                  Controller.
C 0957
C 0958          35  Media format error.                  ---
C 0959
C 0960          36  Device write protected.              ---
C 0961
C 0962          37  Controller  read  or  write com-     ---
C 0963              pare error.
C 0964
C 0965          38  Data error.                          CRC  error  in  the  data
C 0966                                                    field of a disk block.
C 0967
C 0968          39  Host buffer access error            ---
C 0969
C 0970          40  Controller error.                    Difficult  to  catagorize
C 0971                                                    without  looking  at  the
C 0972                                                    error  sub-code  or  any
C 0973                                                    associated  error-log mes-
C 0974                                                    sage.
```

```
C 0975
C 0976
C 0977              41  Drive error.                          See #40.
C 0978
C 0979              42  Host write compare error.             Error detected  when  Host
C 0980                                                        CPU  compared   the   data
C 0981                                                        written and read back. May
C 0982                                                        be a problem with the Host
C 0983                                                        or Controller RAM.
C 0984
C 0985              43 Message from internal diagnostics      See #40.
C 0986
C 0987              44  Duplicate  unit number  detected      ---
C 0988                  by the Controller.
C 0989
C 0990              45  Unknown end code received.            Problem with the  Control-
C 0991                                                        ler microcode or the port/
C 0992                                                        DMA interface.
C 0993
C 0994  DUP ERRORS
C 0995  Host found error
C 0996              46 DBN compare error.                     see # 42
C 0997
C 0998  Message errors
C 0999              47 No local media                         Controller local program on
C 1000                                                        RAM may be corrupt
C 1001
C 1002              48 Illegal Unit #                         Unit went offline
C 1003
C 1004              49 Illegal relative or physical BLK #       see # 31
C 1005
C 1006              50 Device Error                           Possible write protected
C 1007
C 1008              51 Zero lenght message                    see # 31
C 1009
C 1010  Status errors
C 1011              52 Invalid Command                        see # 31
C 1012
C 1013              53 No region available                    see # 31
C 1014
C 1015              54 No region suitable                     see # 31
C 1016
C 1017              55 Program not known                      see # 47
C 1018
C 1019              56 Load failure                           ---
C 1020
C 1021              57 Stand alone type program               see # 31
C 1022
C 1023              58 DUP unkown status code                 see # 31
```

```
:       C 1024
:       C 1025
:       C 1026          SOFT ERRORS
:       C 1027          ----------
:       C 1028
:       C 1029          60  Controller error.                 See  error-log  packet for
:       C 1030                                                  details as the exact cause
:       C 1031                                                  may not be evident.
:       C 1032
:       C 1033          61  Host memory access error.         See #50.
:       C 1034
:       C 1035          62  Disk transfer error.              See #50.
:       C 1036
:       C 1037          63  'Standard    Disk    Inteconnect'  See #50.
:       C 1038              error.
:       C 1039
:       C 1040          64  'Small Disk' error.               See #50.
```

7.0 DATA PATTERNS

	HEX	OCTAL	BINARY
Pattern 1		RANDOM NUMBERS	
Pattern 2	0000	000000	0 000 000 000 000 000
Pattern 3	FFFF	177777	1 111 111 111 111 111
Pattern 4	8B8B	105613	1 000 101 110 001 011
Pattern 5	3333	031463	0 011 001 100 110 011
Pattern 6	3091	030221	0 011 000 010 010 001
Pattern 7	0001	000001	0 000 000 000 000 001
	0003	000003	0 000 000 000 000 011
	0007	000007	0 000 000 000 000 111
	000F	000017	0 000 000 000 001 111
	001F	000037	0 000 000 000 011 111
	003F	000077	0 000 000 000 111 111
	007F	000177	0 000 000 001 111 111
	00FF	000377	0 000 000 011 111 111
	01FF	000777	0 000 000 111 111 111
	03FF	001777	0 000 001 111 111 111
	07FF	003777	0 000 011 111 111 111
	0FFF	007777	0 000 111 111 111 111
	1FFF	017777	0 001 111 111 111 111
	3FFF	037777	0 011 111 111 111 111
	7FFF	077777	0 111 111 111 111 111
	FFFF	177777	1 111 111 111 111 111
Pattern 8	FFFE	177776	1 111 111 111 111 110
	FFFC	177774	1 111 111 111 111 100
	FFF8	177770	1 111 111 111 111 000
	FFF0	177760	1 111 111 111 110 000
	FFE0	177740	1 111 111 111 100 000
	FFC0	177700	1 111 111 111 000 000
	FF80	177600	1 111 111 110 000 000
	FF00	177400	1 111 111 100 000 000
	FE00	177000	1 111 111 000 000 000
	FC00	176000	1 111 110 000 000 000
	F800	174000	1 111 100 000 000 000
	F000	170000	1 111 000 000 000 000
	E000	160000	1 110 000 000 000 000
	C000	140000	1 100 000 000 000 000

```
;       C 1094                  8000    100000          1 000 000 000 000 000
;       C 1095                  0000    000000          0 000 000 000 000 000
```

```
;       C 1096
;       C 1097
;       C 1098          Pattern 9       0000    000000      0 000 000 000 000 000
;       C 1099                          0000    000000      0 000 000 000 000 000
;       C 1100                          0000    000000      0 000 000 000 000 000
;       C 1101                          FFFF    177777      1 111 111 111 111 111
;       C 1102                          FFFF    177777      1 111 111 111 111 111
;       C 1103                          FFFF    177777      1 111 111 111 111 111
;       C 1104                          0000    000000      0 000 000 000 000 000
;       C 1105                          0000    000000      0 000 000 000 000 000
;       C 1106                          FFFF    177777      1 111 111 111 111 111
;       C 1107                          FFFF    177777      1 111 111 111 111 111
;       C 1108                          0000    000000      0 000 000 000 000 000
;       C 1109                          FFFF    177777      1 111 111 111 111 111
;       C 1110                          0000    000000      0 000 000 000 000 000
;       C 1111                          FFFF    177777      1 111 111 111 111 111
;       C 1112                          0000    000000      0 000 000 000 000 000
;       C 1113                          FFFF    177777      1 111 111 111 111 111
;       C 1114
;       C 1115          Pattern 10      B6D9    133331      1 011 011 011 011 001
;       C 1116
;       C 1117          Pattern 11      5555    052525      0 101 010 101 010 101
;       C 1118                          5555    052525      0 101 010 101 010 101
;       C 1119                          5555    052525      0 101 010 101 010 101
;       C 1120                          AAAA    125252      1 010 101 010 101 010
;       C 1121                          AAAA    125252      1 010 101 010 101 010
;       C 1122                          AAAA    125252      1 010 101 010 101 010
;       C 1123                          5555    052525      0 101 010 101 010 101
;       C 1124                          5555    052525      0 101 010 101 010 101
;       C 1125                          AAAA    125252      1 010 101 010 101 010
;       C 1126                          AAAA    125252      1 010 101 010 101 010
;       C 1127                          5555    052525      0 101 010 101 010 101
;       C 1128                          AAAA    125252      1 010 101 010 101 010
;       C 1129                          5555    052525      0 101 010 101 010 101
;       C 1130                          AAAA    125252      1 010 101 010 101 010
;       C 1131                          5555    052525      0 101 010 101 010 101
;       C 1132                          AAAA    125252      1 010 101 010 101 010
;       C 1133
;       C 1134          Pattern 12      2D2D    026455      0 010 110 100 101 101
;       C 1135                          2D2D    026455      0 010 110 100 101 101
;       C 1136                          2D2D    026455      0 010 110 100 101 101
;       C 1137                          D2D2    151322      1 101 001 011 010 010
;       C 1138                          D2D2    151322      1 101 001 011 010 010
;       C 1139                          D2D2    151322      1 101 001 011 010 010
;       C 1140                          2D2D    026455      0 010 110 100 101 101
;       C 1141                          2D2D    026455      0 010 110 100 101 101
;       C 1142                          D2D2    151322      1 101 001 011 010 010
;       C 1143                          D2D2    151322      1 101 001 011 010 010
;       C 1144                          2D2D    026455      0 010 110 100 101 101
;       C 1145                          2D2D    026455      0 010 110 100 101 101
;       C 1146                          D2D2    151322      1 101 001 011 010 010
;       C 1147                          2D2D    026455      0 010 110 100 101 101
;       C 1148                          D2D2    151322      1 101 001 011 010 010
```

```
;       C 1149                  2D2D    026455      0 010 110 100 101 101
;       C 1150                  D2D2    151322      1 101 001 011 010 010
;       C 1151                  2D2D    026455      0 010 110 100 101 101
;       C 1152                  D2D2    151322      1 101 001 011 010 010
;       C 1153                  2D2D    026455      0 010 110 100 101 101
```

```
;       C 1154
;       C 1155
;       C 1156          Pattern 13      6DB6    066666          0 110 110 110 110 110
;       C 1157
;       C 1158          Pattern 14      0001    000001          0 000 000 000 000 001
;       C 1159                          0002    000002          0 000 000 000 000 010
;       C 1160                          0004    000004          0 000 000 000 000 100
;       C 1161                          0008    000010          0 000 000 000 001 000
;       C 1162                          0010    000020          0 000 000 000 010 000
;       C 1163                          0020    000040          0 000 000 000 100 000
;       C 1164                          0040    000100          0 000 000 001 000 000
;       C 1165                          0080    000200          0 000 000 010 000 000
;       C 1166                          0100    000400          0 000 000 100 000 000
;       C 1167                          0200    001000          0 000 001 000 000 000
;       C 1168                          0400    002000          0 000 010 000 000 000
;       C 1169                          0800    004000          0 000 100 000 000 000
;       C 1170                          1000    010000          0 001 000 000 000 000
;       C 1171                          2000    020000          0 010 000 000 000 000
;       C 1172                          4000    040000          0 100 000 000 000 000
;       C 1173                          8000    100000          1 000 000 000 000 000
;       C 1174
;       C 1175          Pattern 15      FFFE    177776          1 111 111 111 111 110
;       C 1176                          FFFD    177775          1 111 111 111 111 101
;       C 1177                          FFFB    177773          1 111 111 111 111 011
;       C 1178                          FFF7    177767          1 111 111 111 110 111
;       C 1179                          FFEF    177757          1 111 111 111 101 111
;       C 1180                          FFDF    177737          1 111 111 111 011 111
;       C 1181                          FFBF    177677          1 111 111 110 111 111
;       C 1182                          FF7F    177577          1 111 111 101 111 111
;       C 1183                          FEFF    177377          1 111 111 011 111 111
;       C 1184                          FDFF    176777          1 111 110 111 111 111
;       C 1185                          FBFF    175777          1 111 101 111 111 111
;       C 1186                          F7FF    173777          1 111 011 111 111 111
;       C 1187                          EFFF    167777          1 110 111 111 111 111
;       C 1188                          DFFF    157777          1 101 111 111 111 111
;       C 1189                          BFFF    137777          1 011 111 111 111 111
;       C 1190                          7FFF    077777          0 111 111 111 111 111
;       C 1191
;       C 1192          Pattern 16      B6D9    133331          1 011 011 011 011 001
;       C 1193                          B6D9    133331          1 011 011 011 011 001
;       C 1194                          B6D9    133331          1 011 011 011 011 001
;       C 1195                          DB6C    155554          1 101 101 101 101 100
;       C 1196                          DB6C    155554          1 101 101 101 101 100
;       C 1197                          DB6C    155554          1 101 101 101 101 100
;       C 1198                          B6D9    133331          1 011 011 011 011 001
;       C 1199                          B6D9    133331          1 011 011 011 011 001
;       C 1200                          DB6C    155554          1 101 101 101 101 100
;       C 1201                          DB6C    155554          1 101 101 101 101 100
;       C 1202                          B6D9    133331          1 011 011 011 011 001
;       C 1203                          DB6C    155554          1 101 101 101 101 100
;       C 1204                          B6D9    133331          1 011 011 011 011 001
;       C 1205                          DB6C    155554          1 101 101 101 101 100
;       C 1206                          B6D9    133331          1 011 011 011 011 001
```

```
;       C 1207                  DB6C    155554          1 101 101 101 101 100
```

```
;       C 1208
;       C 1209
;       C 1210          Pattern 17      LBN     LBN             LBN
;       C 1211                          8D36    106466  1 000 110 100 110 110
;       C 1212                          8D36    106466  1 000 110 100 110 110
;       C 1213                          72C9    071311  0 111 001 011 001 001
;       C 1214                          72C9    071311  0 111 001 011 001 001
;       C 1215                          72C9    071311  0 111 001 011 001 001
;       C 1216                          8D36    106466  1 000 110 100 110 110
;       C 1217                          8D36    106466  1 000 110 100 110 110
;       C 1218                          8D36    106466  1 000 110 100 110 110
;       C 1219                          8D36    106466  1 000 110 100 110 110
;       C 1220                          72C9    071311  0 111 001 011 001 001
;       C 1221                          72C9    071311  0 111 001 011 001 001
;       C 1222                          72C9    071311  0 111 001 011 001 001
;       C 1223                          72C9    071311  0 111 001 011 001 001
;       C 1224                          72C9    071311  0 111 001 011 001 001
;       C 1225                          8D36    106466  1 000 110 100 110 110
;       C 1226                          8D36    106466  1 000 110 100 110 110
;       C 1227                          8D36    106466  1 000 110 100 110 110
;       C 1228                          8D36    106466  1 000 110 100 110 110
;       C 1229                          8D36    106466  1 000 110 100 110 110
;       C 1230                          8D36    106466  1 000 110 100 110 110
;       C 1231
;       C 1232          Pattern 18      8D36    106466  1 000 110 100 110 110
;       C 1233                          LBN     LBN             LBN
;       C 1234                          72C9    071311  0 111 001 011 001 001
;       C 1235                          8D36    106466  1 000 110 100 110 110
;       C 1236                          8D36    106466  1 000 110 100 110 110
;       C 1237                          8D36    106466  1 000 110 100 110 110
;       C 1238                          72C9    071311  0 111 001 011 001 001
;       C 1239                          72C9    071311  0 111 001 011 001 001
;       C 1240                          72C9    071311  0 111 001 011 001 001
;       C 1241                          72C9    071311  0 111 001 011 001 001
;       C 1242                          8D36    106466  1 000 110 100 110 110
;       C 1243                          8D36    106466  1 000 110 100 110 110
;       C 1244                          8D36    106466  1 000 110 100 110 110
;       C 1245                          8D36    106466  1 000 110 100 110 110
;       C 1246                          8D36    106466  1 000 110 100 110 110
;       C 1247                          72C9    071311  0 111 001 011 001 001
;       C 1248                          72C9    071311  0 111 001 011 001 001
;       C 1249                          72C9    071311  0 111 001 011 001 001
;       C 1250                          72C9    071311  0 111 001 011 001 001
;       C 1251                          72C9    071311  0 111 001 011 001 001
;       C 1252                          72C9    071311  0 111 001 011 001 001
```

```
;       C 1253
;       C 1254
;       C 1255          Pattern 19      LBN     LBN             LBN
;       C 1256                          B999    134631  1 011 100 110 011 001
;       C 1257                          B999    134631  1 011 100 110 011 001
;       C 1258                          4666    043146  0 100 011 001 100 110
;       C 1259                          4666    043146  0 100 011 001 100 110
;       C 1260                          4666    043146  0 100 011 001 100 110
;       C 1261                          B999    134631  1 011 100 110 011 001
;       C 1262                          B999    134631  1 011 100 110 011 001
;       C 1263                          B999    134631  1 011 100 110 011 001
;       C 1264                          B999    134631  1 011 100 110 011 001
;       C 1265                          4666    043146  0 100 011 001 100 110
;       C 1266                          4666    043146  0 100 011 001 100 110
;       C 1267                          4666    043146  0 100 011 001 100 110
;       C 1268                          4666    043146  0 100 011 001 100 110
;       C 1269                          4666    043146  0 100 011 001 100 110
;       C 1270                          B999    134631  1 011 100 110 011 001
;       C 1271                          B999    134631  1 011 100 110 011 001
;       C 1272                          B999    134631  1 011 100 110 011 001
;       C 1273                          B999    134631  1 011 100 110 011 001
;       C 1274                          B999    134631  1 011 100 110 011 001
;       C 1275                          B999    134631  1 011 100 110 011 001
;       C 1276
;       C 1277          Pattern 20      B999    134631  1 011 100 110 011 001
;       C 1278                          LBN     LBN             LBN
;       C 1279                          4666    043146  0 100 011 001 100 110
;       C 1280                          B999    134631  1 011 100 110 011 001
;       C 1281                          B999    134631  1 011 100 110 011 001
;       C 1282                          B999    134631  1 011 100 110 011 001
;       C 1283                          4666    043146  0 100 011 001 100 110
;       C 1284                          4666    043146  0 100 011 001 100 110
;       C 1285                          4666    043146  0 100 011 001 100 110
;       C 1286                          4666    043146  0 100 011 001 100 110
;       C 1287                          B999    134631  1 011 100 110 011 001
;       C 1288                          B999    134631  1 011 100 110 011 001
;       C 1289                          B999    134631  1 011 100 110 011 001
;       C 1290                          B999    134631  1 011 100 110 011 001
;       C 1291                          B999    134631  1 011 100 110 011 001
;       C 1292                          4666    043146  0 100 011 001 100 110
;       C 1293                          4666    043146  0 100 011 001 100 110
;       C 1294                          4666    043146  0 100 011 001 100 110
;       C 1295                          4666    043146  0 100 011 001 100 110
;       C 1296                          4666    043146  0 100 011 001 100 110
;       C 1297                          4666    043146  0 100 011 001 100 110
;       C 1298
;       C 1299          Pattern 21      LBN     LBN             LBN
;       C 1300
;          1301  )%
```

```
;       1302  %sbttl 'PROGRAM HEADER'
;       1303
;       1304  library 'ZRQAB0.L16';                    ! RDRX EXERCISER GLOBAL LIBRARY
;       1305
;       1306  require 'BLSMAC.REQ';                    ! DIAGNOSTIC SUPERVISOR LIBRARY
;       2797
;       2798  literal
;       2799      DS$NBR_OF_TESTS = 1;                 ! NUMBER OF TESTS IN THIS DIAGNOSTIC
;       2800
;       2801  EQUALS;
;       2802
;       2803  POINTER (ALL);
;       2804
;       2805  !+
;       2806  ! THE PROGRAM HEADER IS THE INTERFACE BETWEEN
;       2807  ! THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
;       2808  !-
;       2809
;       2810  HEADER (%ascii'ZRQA', %ascii'B', %ascii'0', 32767, 1, PRI00);
```

```
;          2811  %sbttl 'DISPATCH TABLE'
;          2812
;          2813  !+
;          2814  ! THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
;          2815  ! IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
;          2816  !-
;          2817
;          2818  DISPATCH (DS$NBR_OF_TESTS);
```

```
;       2819  %sbttl 'GLOBAL DATA SECTION'
;       2820
;       2821  !+
;       2822  ! THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
;       2823  ! IN MORE THAN ONE TEST.
;       2824  !-
;       2825
;       2826  psect
;       2827      global = $FFF$ (read, nowrite, execute, local, concatenate);
;       2828
;       2829  global
;       2830      CST : blockvector [MAX_CTLR, CST_LEN, word] field (CST_FIELDS),
;       2831                        ! RUN-TIME CONTROLLER STATUS TABLES
;       2832      CST_ADDR : ref block [CST_LEN, word] field (CST_FIELDS),
;       2833                        ! CONTROLLER STATUS TABLE ADDRESS OF "CURRENT" CONTROLLER
;       2834      DCT : blockvector [MAX_CTLR, DCT_LEN, word] field (DCT_FIELDS),
;       2835                        ! DRIVER CONTROLLER TABLES
;       2836      DCT_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS),
;       2837                        ! ADDRESS OF "CURRENT" DRIVER CONTROLLER TABLE
;       2838      RDRX_ADDR : ref rdrx field (RC_REG),
;       2839                        ! DEVICE ADDRESS OF "CURRENT" CONTROLLER
;       2840      IRDRX_ADDR : ref rdrx field (RC_REG),
;       2841                        ! DEVICE ADDRESS OF INTERRUPTING CONTROLLER
;       2842      DUPPKT : BLOCK [257, WORD] field (DP_FIELDS),
;       2843                        ! BUFFER CONTAINING DUP INFORMATION FROM RECEIVE AND SEND COMMANDS
;       2844      TALLY : vector [MAX_UNITS * TALLY_LEN, word] field (T_FIELDS),
;       2845                        ! STATISTICS TABLES
;       2846      T_ADDR : ref block [TALLY_LEN, word] field (T_FIELDS),
;       2847                        ! ADDRESS OF STATISTICS TABLE (TALLY) FOR CURRENT UNIT
;       2848      C_ERR_TBL : blockvector [MAX_CTLR, C_ERR_LEN, word] field (C_ERR_FIELDS),
;       2849                        ! STATISTICS TABLE FOR CONTROLLER ERRORS
;       2850      MSCP_PKT : blockvector [PKT_CNT, PKT_LEN, word] field (PKT_FIELDS),
;       2851                        ! MSCP PACKET POOL
;       2852      IPKT_ADDR : ref block [PKT_LEN, word] field (PKT_FIELDS),
;       2853                        ! ADDRESS OF AN MSCP PACKET (INTERUPT PROCESSING)
;       2854      PKT_USE : vector [PKT_CNT, byte, signed],
;       2855                        ! MSCP PACKET POOL ALLOCATION TABLE
;       2856      RETPKT : blockvector [RP_CNT, RP_LEN, word] field (RP_FIELDS),
;       2857                        ! RETURN PACKET POOL
;       2858      RP_USE : vector [RP_CNT, byte, signed],
;       2859                        ! RETURN PACKET POOL ALLOCATION TABLE
;       2860      RP_INDX : word,   ! CURRENT RETURN PACKET INDEX
;       2861      RP_ADDR : ref block [RP_LEN, word] field (RP_FIELDS),
;       2862                        ! CURRENT RETURN PACKET ADDRESS
;       2863      ELOG_PKT : blockvector [EP_CNT, EP_LEN, word] field (EP_FIELDS),
;       2864                        ! ERROR-LOG PACKET SAVE AREA
;       2865      BUFF_ADDR : vector [MAX_BUF_CNT],                   ! TABLE OF I/O BUFFER DESCRIPTORS
;       2866      BUFF_OWN : vector [MAX_BUF_CNT, byte, signed],      ! I/O BUFFER OWNERSHIP (CONTROLLER NUMBER)
;       2867      IODQ : vector [IODQ_LEN, byte],
;       2868                                    ! I/O DONE QUEUE - CIRCULAR QUEUE OF RETPKT INDECES
;       2869      IODQ_IN : word,               ! I/O DONE QUEUE IN POINTER
;       2870      IODQ_OUT : word,              ! I/O DONE QUEUE OUT POINTER
;       2871      ENTRY_REASON : byte,          ! CURRENT OPERATOR COMMAND
```

```
;        2872      EOP_FLAG : byte,                    ! END-OF-PASS FLAG
;        2873      DUP_FLAGS : WORD,                   ! DUP FLAGS
;        2874      CCTLR : word,                       ! NUMBER OF "CURRENT" CONTROLLER
;        2875      CDISK : word,                       ! CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;        2876      CUOFF : word,                       ! CURRENT UNIT CST OFFSET
;        2877      CTLR_CNT : word,                    ! TOTAL NUMBER OF CONFIGURED CONTROLLERS
;        2878      DUR : vector [MAX_UNITS, byte],     ! DROP UNIT REASON
;        2879      QIO : vector [MAX_CTLR, byte],      ! NUMBER OF OUTSTANDING QIOS PER CONTROLLER
;        2880      FREE_MEM_ADDR,                      ! START OF FREE MEMORY
;        2881      BYTS_PER_QIO : word,                ! SIZE (BYTES) OF AN I/O BUFFER
;        2882      ST_CODE : word,                     ! CURRENT STATUS CODE
;        2883      SB_CODE : word,                     ! CURRENT SUB-CODE
;        2884      STEP : word,                        ! CURRENT STEP IN HARD_INIT
;        2885      OF_RC : signed word,                ! OFFSET (0 OR 2) TO READ IP OR SA
;        2886      SA_REG : word,                      ! STORAGE FOR SA REGISTER READS AND WRITES
;        2887      CMD_TIME : word,                    ! COMMAND TIMEOUT VALUE (IN SECONDS)
;        2888      NEX : word,                         ! NON-EXISTENT MEMORY TRAP INDICATOR
;        2889      CRN_LOW : word,                     ! COMMAND REF NUMBER OF LAST COMMAND SENT
;        2890      CRN_HIGH : word,                    ! COMMAND REF NUMBER (HI ORDER)
;        2891      P_INDEX : signed word,      ! CURRENT message PACKET INDEX
;        2892      S_DUPPKT : WORD,            ! DBN BYTE COUNTER
;        2893      S_PATTERN : WORD,           ! THE PATTERN WRITTEN TO DBN'S
;        2894      CREDIT_BAL : word,                  ! CREDIT BALANCE
;        2895      NEXT_PKT_USE : byte;                ! POINTER TO NEXT ENTRY IN PKT_USE TABLE
;        2896
;        2897  ERRTBL;
```

```
;       2898  %sbttl 'GLOBAL TEXT SECTION'
;       2899
;       2900  !+
;       2901  ! THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
;       2902  ! MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
;       2903  ! MORE THAN ONE TEST.
;       2904  !-
;       2905
;       2906  global bind
;       2907          !
;       2908          ! HARDWARE DIALOG
;       2909          !
;       2910              HWQ1 = uplit (%asciz'IP ADDRESS'),
;       2911              HWQ2 = uplit (%asciz'VECTOR'),
;       2912              HWQ3 = uplit (%asciz'BR LEVEL'),
;       2913              HWQ4 = uplit (%asciz'RD/RX DRIVE NUMBER'),
;       2914              HWQ5 = uplit (%asciz'IS DISK AN RD51 WINCHESTER (NO, IMPLIES AN RX50 FLOPPY)'),
;       2915              HWQ6 = uplit (%asciz'STARTING LBN'),
;       2916              HWQ7 = uplit (%asciz'ENDING LBN (MAXIMUM -  RX50: 799, RD51: 21599)'),
;       2917              HWQ8 = uplit (%asciz'WRITE ON CUSTOMER DATA AREA ON THIS DISK'),
;       2918              HWQ9 = uplit (%asciz'** WARNING - CUSTOMER DATA AREA MAY BE OVERWRITTEN! ... CONFIRM'),
;       2919              HWQ10 = uplit (%asciz'ALSO EXERCISE ON DIAGNOSTIC AREA (-NON-CUSTOMER AREA)'),
;       2920              HWQ11 = uplit (%asciz'WRITE ON DIAGNOSTIC AREA'),
;       2921
;       2922          !
;       2923          ! SOFTWARE DIALOG
;       2924          !
;       2925              SWQ1  = uplit (%asciz'HARD ERROR LIMIT'),
;       2926              SWQ2  = uplit (%asciz'TRANSFER LIMIT IN MEGABYTES (0 FOR ''QUICK PASS'')'),
;       2927              SWQ4  = uplit (%asciz'RANDOM SEEK MODE'),
;       2928              SWQ7  = uplit (%asciz'READ-COMPARES PERFORMED AT THE CONTROLLER'),
;       2929              SWQ9  = uplit (%asciz'WRITE-COMPARES PERFORMED AT THE CONTROLLER'),
;       2930              SWQ10 = uplit (%asciz'CHECK ALL WRITES AT HOST BY READING'),
;       2931              SWQ11 = uplit (%asciz'USER-DEFINED DATA PATTERN'),
;       2932              SWQ12 = uplit (%asciz'SELECT PRE-DEFINED DATA PATTERN (0 FOR SEQUENTIAL SELECTION)'),
;       2933              SWQ13 = uplit (%asciz'NUMBER OF WORDS IN DATA PATTERN (16 MAXIMUM)'),
;       2934              SWQ14 = uplit (%asciz'PATTERN VALUE'),
;       2935              SWQ15 = uplit (%asciz'CLEAR STATISTICAL TABLES AFTER PRINTING'),
;       2936              SWQ17 = uplit (%asciz'PERCENTAGE OF RD51 OPERATIONS OUT OF TOTAL OPERATIONS'),
;       2937              SWQ19 = uplit (%asciz'UNITS TO BE SELECTED AT RANDOM (NO, IMPLIES SEQUENTIAL)'),
;       2938  !           SWQ20 = uplit (%asciz'WANT TO REWRITE BLOCKS WHEN ''FORCED ERROR'' DETECTED ON READS'),
;       2939  !           SWQ21 = uplit (%asciz'IF ''HALT ON ERROR'' FLAG SET, WANT TO HALT ON HARD/SOFT ERRORS ALSO'),
;       2940              SWQ22 = uplit (%asciz'NUMBER OF DBNs READ AT ONE TIME (effects RD51s only)'),
;       2941              SWM1  = uplit (%asciz'THE REMAINING QUESTIONS ONLY APPLY TO UNPROTECTED DISKS'),
;       2942              NULL  = uplit (%asciz''),
;       2943
;       2944  !++
;       2945  !  THE FOLLOWING DBMs ARE DEBUG MESSAGES, AND SHOULD BE REMOVED BEFORE
;       2946  !  RELEASING THE PROGRAM.  THEY INCLUDE THE NAMES OF EACH ROUTINE, PLUS
;       2947  !  FORMAT STATEMENTS FOR PRINTING OUT OTHER INFORMATION.
;       2948  !--
;       2949
;       2950      DBM5   = uplit (%asciz'%N%A** DROP UNIT %D2'),
```

```
2951        DBM12  = uplit (%asciz'%N%A** PROC_RETPKT: CONN ID = %D5%A RECEIVED'),
2952    !    DBM15  = uplit (%asciz'%N%A** MULTI-DRIVE TEST'),
2953        DBM18  = uplit (%asciz'%N%A** FATAL_ERROR: RETPKT NOT AVAILABLE'),
2954        DBM19  = uplit (%asciz'%N%A** FSET_DPAR: CAN''T FIND DISK %D3%A IN CST %D1'),
2955        DBM20  = uplit (%asciz'%N%A** BAD CONN ID = %D5%A RECEIVED FROM %O6'),
2956        DBM21  = uplit (%asciz'%N%A** MESSAGE TYPE %D2%A RECEIVED IN MSCP PACKET'),
2957    !    DBM22  = uplit (%asciz'%N%A** SEQUEN: RETPKT NOT AVAILABLE'),
2958        DBM23  = uplit (%asciz'%N%A** ERROR IN SET_CTLR_CHAR'),
2959        DBM25  = uplit (%asciz'%N%A** CTLR TIMEOUT = %D3%A SECONDS'),
2960        DBM26  = uplit (%asciz'%N%A** ERROR IN UNIT_INIT'),
2961    !    DBM27  = uplit (%asciz'%N%A** UNIT_INIT: RETPKT HAS BAD ENDCODE'),
2962    !    DBM28A = uplit (%asciz'%N%A** UNIT SIZE (LO) = %D5'),
2963    !    DBM28B = uplit (%asciz'%N%A** UNIT SIZE (HI) = %D5'),
2964        DBM29  = uplit (%asciz'%N%A** ACCESS: RETPKT HAS BAD ENDCODE'),
2965    !    DBM32  = uplit (%asciz'%N%A** QIO_UNIT: CST %D1%A NO UNIT SELECTED'),
2966    !    DBM101 = uplit (%asciz'%N%A** UNIT # IS: %O6'),
2967    !    DBM104 = uplit (%asciz'%N%A** RX50 IS SELECTED'),
2968    !    DBM105 = uplit (%asciz'%N%A** RD51 IS SELECTED'),
2969        DBM107 = uplit (%asciz'%N%A** ILLEGAL FUNCTION: %O6'),
2970        DBM108 = uplit (%asciz'%N%A** COMMAND REF # %D5%A. NOT SENT BY HOST'),
2971        DBM109 = uplit (%asciz'%N%A** UNKNOWN ERROR LOG FORMAT %D3%A. RECEIVED'),
2972        DBM110 = uplit (%asciz'%N%A** ERROR-LOG SAVE AREA FULL'),
2973    !    DBM111 = uplit (%asciz'%N%A** REINIT MSCP EXERCISER'),
2974        DBM112  = uplit (%asciz'%N%A** DUP_RSP: RETPKT NOT AVAILABLE'),
2975    !
2976    ! DROP UNIT MESSAGES
2977    !
2978        DU_MSG = uplit (%asciz'%N%AUNIT%D2%A DROPPED - '),
2979        DU_RSN = uplit (
2980                  uplit (%asciz'%AUSER COMMAND%N'),
2981                  uplit (%asciz'%ACONFIGURATION ERROR%N'),
2982                  uplit (%asciz'%AINIT ERROR%N'),
2983                  uplit (%asciz'%ATRANSFER LIMIT REACHED%N'),
2984                  uplit (%asciz'%AERROR LIMIT REACHED%N'),
2985                  uplit (%asciz'%AUNRECOVERABLE DEVICE ERROR%N'),
2986                  uplit (%asciz'%AUNRECOVERABLE CONTROLLER ERROR%N'),
2987                  uplit (%asciz'%AFAILED TO COME ONLINE%N'),
2988                  uplit (%asciz'%AFAILED TO ACCESS LAST TRACK DURING INIT%N'),
2989                  uplit (%asciz'%ADISK WRITE PROTECTED%N'),
2990                  uplit (%asciz'%ACOMMAND TIME OUT%N'),
2991                  uplit (%asciz'%AUNIT WENT TO AVAILABLE STATE%N')) : vector [11],
2992    !
2993    ! SYSTEM MESSAGES (PRINTF)
2994    !
2995        MSG_01 = uplit (%asciz'%N%APOWER DELAY - WAITING'),
2996        MSG_02 = uplit (%asciz'%N%AFUNCTIONAL TEST STARTED'),
2997        MSG_03 = uplit (%asciz'%N%N%AEXERCISER STARTED%N'),
2998    !
2999    ! REPORT MESSAGES (PRINTS)
3000    !
3001        RPT1  = uplit (%asciz'%N%N%AUNT DSK%S8%A# OF    # BYTES    # OF     # BYTES'),
3002        RPT2  = uplit (%asciz'%A  --HARD ERRORS-- --SOFT ERRORS--'),
3003        RPT3  = uplit (%asciz'%N%A #    #  TYPE  READS      READ   WRITES   WRITTEN'),
```

```
3004        RPT4  = uplit (%asciz'%A  SEK DAT DRV HST SEK DAT DRV HST'),
3005        RPT5  = uplit (%asciz'%N%A--- --- ----  -----  ---------.------  ---------'),
3006        RPT6  = uplit (%asciz'%A --- --- --- --- --- --- --- ---'),
3007        RPT7  = uplit (%asciz'%N%D2%D4%A  RX50'),
3008        RPT8  = uplit (%asciz'%N%D2%D4%A  RD51'),
3009        RPT9  = uplit (%asciz'%D4%Z3%D3%A,%Z3%A,%Z3'),
3010        RPT10 = uplit (%asciz'%D4%D4%D4%D4%D4%D4%D4%D4%D4'),
3011        RPT11 = uplit (%asciz'%N%A .   .  CNTR      :  .........    .   .  .........'),
3012        RPT12 = uplit (%asciz'%A   .   .%D4%A    .   .   .%D4%A    .'),
3013        RPT13 = UPLIT(%ASCIZ'%N%N%AUNIT  DISK          # OF   # BLKS      # OF    # BLKS '),
3014        RPT14 = UPLIT(%ASCIZ'%N%A  #     #     TYPE   READS     READ      WRITES  WRITTEN '),
3015        RPT15 = UPLIT(%ASCIZ'%N%A----  ----  -------  ------  ------     ------  ------  '),
3016        RPT16 = UPLIT(%ASCIZ'%N%S1%D2%S4%D2%A   DBNRD51  %D6%S3%D6%S5%D6%S3%D6'),
3017
3018    !
3019    ! GENERAL ERROR MESSAGES
3020    !
3021    !       SYSTEM FATAL (ERRSF)
3022    !
3023        EGS_01 = uplit (%asciz'TOO MANY UNITS'),
3024        EGS_02 = uplit (%asciz'NOT ENOUGH FREE MEMORY FOR ALLOCATING READ/WRITE BUFFERS'),
3025    !
3026    !       DEVICE FATAL (ERRDF)
3027    !
3028        EGD_10 = uplit (%asciz'REGISTER EXISTENCE TEST FAILED'),
3029        EGD_11 = uplit (%asciz'VECTOR TEST FAILED'),
3030        EGD_12 = uplit (%asciz'BR LEVEL TEST FAILED'),
3031        EGD_13 = uplit (%asciz'INIT SEQUENCE FAILED'),
3032        EGD_14 = uplit (%asciz'FATAL CONTROLLER ERROR'),
3033        EGD_15 = uplit (%asciz'ONLINE FAILED'),
3034        EGD_16 = uplit (%asciz'WRITE-PROTECT CONFLICT'),
3035        EGD_17 = uplit (%asciz'ACCESS FAILED'),
3036        EGD_18 = uplit (%asciz'FATAL I/O ERROR'),
3037    !   EGD_19 = uplit (%asciz'CONTROLLER TIMEOUT'),
3038        EGD_20 = uplit (%asciz'FAILED TO SEND SET-CONTROLLER-CHARACTERISTICS COMMAND'),
3039        EGD_21 = uplit (%asciz'SET-CONTROLLER-CHARACTERISTICS RESPONSE HAS BAD ENDCODE OR FLAGS IN ERROR'),
3040        EGD_22 = uplit (%asciz'FAILED TO SEND ON-LINE COMMAND'),
3041        EGD_23 = uplit (%asciz'ON-LINE RESPONSE HAS BAD ENDCODE'),
3042    !
3043    !       HARD or SOFT (ERRHRD or ERRSOFT)
3044    !
3045        EGH_30 = uplit (%asciz'I/O REQUEST FAILED'),
3046    !
3047    ! BASIC ERROR MESSAGES (PRINTB)
3048    !
3049    !       SYSTEM FATAL (ERRSF)
3050    !
3051        EBS_01 = uplit (%asciz'%AMORE THAN %D2%A UNITS SPECIFIED'),
3052    !
3053    !       DEVICE FATAL (ERRDF)
3054    !
3055        EBD_10 = uplit (%asciz'%A* NO RESPONSE AT ADDRESS %O6'),
3056        EBD_12 = uplit (%asciz'%A* INCORRECT BR LEVEL FOR DEVICE %O6'),
```

```
3057      EBD_13 = uplit (%asciz'%A* STEP %D1%A READ ERROR'),
3058      EBD_14 = uplit (%asciz'%A* BAD SA CODE FROM DEVICE %O6'),
3059      EBD_18 = uplit (%asciz'%A* DISK%D2%A WENT OFFLINE'),
3060      EBD_19 = uplit (%asciz'%A* DEVICE %O6%A NOT PROCESSING COMMAND PACKETS'),
3061  !
3062  !       HARD or SOFT (ERRHRD or ERRSOFT)
3063  !
3064      EH_0 = UPLIT (%ASCIZ' - unrecognized MESSAGE TYPE'),
3065      EH_1 = UPLIT (%ASCIZ' - unrecognized connection id'),
3066      EH_2 = UPLIT (%ASCIZ' - unrecognized RETURN message'),
3067      EH_3  = UPLIT (%ASCIZ' - unrecognized RETURN PACKET'),
3068      EH_4 = UPLIT (%ASCIZ' - unrecognized CRN'),
3069      EH_5 = UPLIT (%ASCIZ' - UNRECOGNIZED OPCODE'),
3070      EH_6 = UPLIT (%ASCIZ' - MSCP STATUS CODE ERROR'),
3071      EH_7 = UPLIT (%ASCIZ' - DUP STATUS CODE ERROR'),
3072      EH_8 = UPLIT (%ASCIZ' - unrecognized STATUS CODE'),
3073      EH_9 = UPLIT (%ASCIZ' - LBN HOST COMPARE ERROR'),!
3074      EH_10 = UPLIT (%ASCIZ' - DBN HOST COMPARE ERROR'),
3075      EH_12 = UPLIT (%ASCIZ' - UNABLE TO LOAD DUP MEDIA '),
3076      EH_13 = UPLIT (%ASCIZ' - ERROR IN DUP-PKT WHEN USING CTLR LC PRG'),
3077      ERR_COD = uplit (
3078                uplit (%asciz'%AINVALID COMMAND'),
3079                uplit (%asciz'%ACOMMAND ABORTED'),
3080                uplit (%asciz'%AUNIT OFFLINE'),
3081                uplit (%asciz'%ATRANSITION TO AVAILABLE STATE'),
3082                uplit (%asciz'%AMEDIA FORMAT ERROR'),
3083                uplit (%asciz'%AWRITE-PROTECTED'),
3084                uplit (%asciz'%ADEVICE COMPARE ERROR'),
3085                uplit (%asciz'%ADATA ERROR'),
3086                uplit (%asciz'%AHOST BUFFER ACCESS ERROR'),
3087                uplit (%asciz'%ACONTROLLER ERROR'),
3088                uplit (%asciz'%ADRIVE ERROR'),
3089                uplit (%asciz'%AMESSAGE FROM INTERNAL DIAGNOSTICS'),
3090                uplit (%asciz'%AHOST COMPARE ERROR'),
3091                uplit (%asciz'%ACOMMAND TIMEOUT')) : vector [14],
3092  !
3093  !       ERROR LOG MESSAGE (ERRSOFT)
3094  !
3095      ELG_00  = uplit (%asciz'%N%AERROR LOG MESSAGE RECEIVED:%N'),
3096      ELG_FMT = uplit (
3097                uplit (%asciz'%A* CONTROLLER ERROR%N'),
3098                uplit (%asciz'%A* HOST MEMORY ACCESS ERROR%N'),
3099                uplit (%asciz'%A* DISK%D2%A - DISK TRANSFER ERROR%N'),
3100                uplit (%asciz'%A* DISK%D2%A - ''STANDARD DISK INTERCONNECT'' ERROR%N'),
3101                uplit (%asciz'%A* DISK%D2%A - ''SMALL DISK'' ERROR%N')) : vector [5],
3102  !
3103  ! EXTENDED ERROR MESSAGES (PRINTX)
3104  !
3105
3106      EX_BDR = uplit (%asciz'%N%AI/O BUFFER ADDRESS FOR READ (32 BITS): %O6%A %O6%N'),
3107      EX_BDW = uplit (%asciz'%N%AI/O BUFFER ADDRESS FOR WRITE (32 BITS): %O6%A %O6'),
3108      EX_LBR = uplit (%asciz'%N%ALBN: (READ) %D5%A. (OCT %O6%A)'),
3109      EX_LBW = uplit (%asciz'%N%ALBN: (WRITE) %D5%A. (OCT %O6%A)'),
```

```
3110        EX_RBN = uplit (%asciz'%N%AREPLACEMENT BLOCK NO. %D5%A. (OCT %O6%A)'),
3111        EX_CBR = uplit (%asciz'%N%ABYTE COUNT IN READ COMMAND: %D5%A.'),
3112        EX_CBW = uplit (%asciz'%N%ABYTE COUNT IN WRITE COMMAND: %D5%A.'),
3113
3114    XX13 = UPLIT (%ASCIZ'%N%A * DISK : %D2'),
3115    XX14 = uplit (%asciz'%N%ASA: %O6'),
3116    XX15 = uplit (%asciz'%N%ASTATUS CODE: '),
3117    XX16 = uplit (%asciz'%N%ASTATUS SUB-CODE: '),
3118    XX17 = uplit (%asciz'%N%ACOMMAND: '),
3119    XX18 = uplit (%asciz'%A-DUP-'),
3120    XX19 = uplit (%asciz'%A-MSCP-'),
3121    XX20 = uplit (%asciz'%A-COMPARE'),
3122    XX21 = uplit (%asciz'%N%ABAD BLOCK REPORTED: %O5%A.'),
3123    XX22 = uplit (%asciz'%N%ALBN: %D5%A. (OCT %O6%A)'),
3124    XX23 = UPLIT (%ASCIZ'%N%ADBN: %D5%A. (OCT %O6%A)'),
3125    XX24 = uplit (%asciz'%N%ABYTE COUNT IN COMMAND: %D5'),
3126    XX25 = uplit (%asciz'%N%AACTUAL # OF BYTES TRANSFERRED: %D5'),
3127    XX26 = uplit (%asciz'%N%AI/O BUFFER ADDRESS (32 BITS): %O6%A %O6'),
3128    XX27 = uplit (%asciz'%N%ACONTENTS OF RETURN PACKET:%N'),
3129    XX29 = UPLIT (%ASCIZ'%N%AMESSAGE TYPE: '),
3130    XX30 = UPLIT (%ASCIZ'%N%AMESSAGE NUMBERS: '),
3131    XX31 = UPLIT (%ASCIZ'%N%AMESSAGE ERROR CODES: '),
3132    XX32 = UPLIT (%ASCIZ'%N%ABYTE NUMBER: %D3'),
3133    XX33 = UPLIT (%ASCIZ'%N%ARANDOM WRITTEN WORD :%B16'),
3134    XX34 = UPLIT (%ASCIZ'%N%ARANDOM READ WORD bin:%B16%A oct:%O6'),
3135    XX35 = UPLIT (%ASCIZ'%N%ACRN : %O6'),
3136    !XX36 = UPLIT (%ASCIZ'%N%ATHE EXPECTED CRN : %O6'),
3137    XX37 = UPLIT (%ASCIZ'%A - UNKNOWN : %D2'),
3138    XX38 = UPLIT (%ASCIZ'%A - UNKNOWN CONNECTION ID: %D3%A -'),
3139    XX39 = UPLIT (%ASCIZ'%N%ACONTROLLER FLAGS:'),
3140    XX40 = UPLIT (%ASCIZ'%N%AUNIT FLAGS:'),
3141    XX41 = UPLIT (%ASCIZ'%N%AEND MESSAGE FLAGS:'),
3142
3143
3144    !
3145    ! UNKNOWN RETURN MESSAGES
3146    !
3147          EB_DCT = UPLIT (%ASCIZ'%N%A DRIVER CONTROLLER TABLE = ADDR: %D6%N'),
3148          EB_COMM = UPLIT (%ASCIZ'%N%A CMD INT, RSP INT, COMMAND RING = ADDR: %D6%N'),
3149          EB_PKT = UPLIT (%ASCIZ'%N%A ALL PACKETS IN MESSAGE AREA'),
3150          EB_RAL = UPLIT (%ASCIZ'%N%A ALL RETURN PACKETS IN AREA'),
3151          EB_ADDR = UPLIT (%ASCIZ'%N%A ADDR: %D6%A     PACKET = %N'),
3152          EBNEX1 = UPLIT (%ASCIZ'%N%A ADDR OF RESPONSE RING TO BE POLLED %D6'),
3153          EB_NEX2 = UPLIT (%ASCIZ'%N%A ADDR OF MESSAGE PACKET RESPONSE RING SLOT POINTS TO %D6'),
3154          EBNEX3 = UPLIT (%ASCIZ'%N%A ADDR OF MESSAGE PACKET COMMAND RING SLOT POINTS TO %D6'),
3155    !
3156    ! CONFIGURATION ERROR MESSAGES (PRINTF)
3157    !
3158       CER_01 = uplit (%asciz'%N%ADUPLICATE UNIT:%D2%A AT IP: %O6'),
3159       CER_02 = uplit (%asciz'%N%AMORE THAN %D1%A DIFFERENT IP ADDRESSES'),
3160
3161    !
3162    ! MESSAGE TYPES
```

```
;       3163  !
;       3164      EX_SEQ = UPLIT (%ASCIZ'%A- SEQUENTIAL'),
;       3165      EX_CRD = UPLIT (%ASCIZ'%A- CREDIT NOTIFICATION'),
;       3166      EX_MTN = UPLIT (%ASCIZ'%A- MAINTENANCE'),
;       3167      EX_DGM = UPLIT (%ASCIZ'%A- DATAGRAM'),
;       3168  !
;       3169  ! commands
;       3170  !mscp
;       3171      EX_RD = uplit (%asciz'%AREAD'),
;       3172      EX_WRT = uplit (%asciz'%AWRITE'),
;       3173      EX_ACC = uplit (%asciz'%AACCESS'),
;       3174      EX_ONL = uplit (%asciz'%AON LINE'),
;       3175      EX_SCC = uplit (%asciz'%ASET CONTROL CHAR.'),
;       3176  !dup
;       3177      EX_GDS = uplit (%asciz'%AGET DUST STATUS'),
;       3178      EX_ESP = uplit (%asciz'%AEXECUTE SUPPLIED PRG'),
;       3179      EX_ELP = uplit (%asciz'%AEXECUTE LOCAL PRG'),
;       3180      EX_SDD = uplit (%asciz'%ASEND DATA'),
;       3181      EX_RCD = uplit (%asciz'%ARECEIVE DATA'),
;       3182      EX_ABP = uplit (%asciz'%AABORT'),
;       3183  !
;       3184  ! ERROR/EVENT SUB CODES (PRINTX)
;       3185  !
;       3186      SC_SDI = uplit (%asciz'%ASPIN-DOWN IGNORED'),
;       3187      SC_CON = uplit (%asciz'%ASTILL CONNECTED'),
;       3188      SC_DUP = uplit (%asciz'%ADUPLICATE UNIT NUMBER'),
;       3189      SC_ONL = uplit (%asciz'%AALREADY ONLINE'),
;       3190      SC_SON = uplit (%asciz'%ASTILL ONLINE'),
;       3191      SC_UNK = uplit (%asciz'%AUNIT UNKNOWN OR ONLINE TO ANOTHER CONTROLLER'),
;       3192      SC_VOL = uplit (%asciz'%ANO VOLUME MOUNTED OR DRIVE DISABLED BY SWITCH'),
;       3193      SC_IOP = uplit (%asciz'%AUNIT INOPERATIVE (RD51 WRITE FAULT'),
;       3194      SC_DIS = uplit (%asciz'%AUNIT DISABLED BY FIELD SERVICE OR INTERNAL DIAGNOSTICS'),
;       3195      SC_FER = uplit (%asciz'%A''FORCED ERROR'' DETECTED WHILE ACCESSING FCT OR RCT'),
;       3196      SC_FE2 = uplit (%asciz'%ASECTOR WRITTEN WITH ''FORCED ERROR'' MODIFIER'),
;       3197      SC_ISH = uplit (%asciz'%AFCT OR RCT UNREADABLE - INVALID SECTOR HEADER'),
;       3198      SC_IS2 = uplit (%asciz'%AHEADER COMPARE ERROR (VALID HEADER NOT FOUND)'),
;       3199      SC_DST = uplit (%asciz'%AFCT OR RCT UNREADABLE - DATA SYNC TIMEOUT'),
;       3200      SC_DS2 = uplit (%asciz'%ADATA SYNC NOT FOUND (DATA SYNC TIMEOUT)'),
;       3201      SC_ECC = uplit (%asciz'%AFCT OR RCT UNREADABLE - UNCORRECTABLE ECC ERROR'),
;       3202      SC_ECD = uplit (%asciz'%AUNCORRECTABLE ECC ERROR'),
;       3203      SC_RCT = uplit (%asciz'%ARCT CORRUPTED'),
;       3204      SC_FUL = uplit (%asciz'%ANO REPLACEMENT BLOCK AVAILABLE (RCT FULL)'),
;       3205      SC_576 = uplit (%asciz'%ADISK NOT FORMATTED WITH 512 BYTE SECTORS'),
;       3206      SC_FCT = uplit (%asciz'%ADISK NOT FORMATTED OR FCT CORRUPTED'),
;       3207      SC_EC1 = uplit (%asciz'%AONE SYMBOL ECC ERROR'),
;       3208      SC_EC2 = uplit (%asciz'%ATWO SYMBOL ECC ERROR'),
;       3209      SC_EC3 = uplit (%asciz'%ATHREE SYMBOL ECC ERROR'),
;       3210      SC_EC4 = uplit (%asciz'%AFOUR SYMBOL ECC ERROR'),
;       3211      SC_EC5 = uplit (%asciz'%AFIVE SYMBOL ECC ERROR'),
;       3212      SC_EC6 = uplit (%asciz'%ASIX SYMBOL ECC ERROR'),
;       3213      SC_EC7 = uplit (%asciz'%ASEVEN SYMBOL ECC ERROR'),
;       3214      SC_EC8 = uplit (%asciz'%AEIGHT SYMBOL ECC ERROR'),
;       3215      SC_EC9 = uplit (%asciz'%ACORRECTABLE ERROR IN ECC FIELD'),
```

```
;          3216      SC_SWP = uplit (%asciz'%AUNIT SOFTWARE WRITE PROTECTED'),
;          3217      SC_HWP = uplit (%asciz'%AUNIT HARDWARE WRITE PROTECTED'),
;          3218      SC_ODA = uplit (%asciz'%AODD TRANSFER ADDRESS'),
;          3219      SC_ODB = uplit (%asciz'%AODD BYTE COUNT'),
;          3220      SC_NXM = uplit (%asciz'%ANON-EXISTENT HOST MEMORY'),
;          3221      SC_PAR = uplit (%asciz'%AHOST MEMORY PARITY ERROR'),
;          3222      SC_CTO = uplit (%asciz'%ACOMMAND TIMOUT OR RETRY LIMIT EXCEEDED'),
;          3223      SC_SDS = uplit (%asciz'%ASERIALIZER/DESERIALIZER OVERRUN OR UNDERRUN'),
;          3224      SC_EDC = uplit (%asciz'%AEDC ERROR'),
;          3225      SC_IDS = uplit (%asciz'%AINCONSISTENT INTERNAL DATA STRUCTURE'),
;          3226      SC_SRT = uplit (%asciz'%ADRIVE COMMAND TIMEOUT (NO RESPONSE OR SEEK INCOMPLETE)'),
;          3227      SC_SRI = uplit (%asciz'%ACONTROLLER DETECTED TRANSMISSION OR PROTOCOL ERROR'),
;          3228      SC_POE = uplit (%asciz'%APOSITION ERROR (MIS-SEEK)'),
;          3229      SC_RDY = uplit (%asciz'%ALOST READ/WRITE READY DURING/BETWEEN TRANSFERS'),
;          3230      SC_CLK = uplit (%asciz'%ADRIVE CLOCK DROPOUT'),
;          3231      SC_RSP = uplit (%asciz'%ALOST RECEIVER READY BETWEEN SECTORS'),
;          3232      SC_SUR = uplit (%asciz'%ADRIVE DETECTED ERROR'),
;          3233      SC_PSP = uplit (%asciz'%ACONTROLLER DETECTED PULSE OR STATE PARITY ERROR'),
;          3234  !
;          3235  ! MSCP END MESSAGE FLAGS
;          3236  !
;          3237  F_0 = uplit (%asciz'%N%A! Bad Block Reported'),
;          3238  F_1 = uplit (%asciz'%N%A ! Bad Block Unreported'),
;          3239  F_2 = uplit (%asciz'%N%A ! Error Log Generated'),
;          3240  F_3 = uplit (%asciz'%N%A ! Serious Exception'),
;          3241  ! MSCP Controller Flags
;          3242  F_4 = uplit (%asciz'%N%A! Enable Attention Messages'),
;          3243  F_5 = uplit (%asciz'%N%A! Enable Miscellaneous Error Log Messages'),
;          3244  F_6 = uplit (%asciz'%N%A! Enable Other Hosts Error Log Messages'),
;          3245  F_7 = uplit (%asciz'%N%A! Enable This Hosts Error Log Messages'),
;          3246  F_8 = uplit (%asciz'%N%A! Controller Initiated Bad Block Rplcmnt'),
;          3247  F_9 = uplit (%asciz'%N%A! Shadowing'),
;          3248  F_10 = uplit (%asciz'%N%A! 576 Byte Sectors'),
;          3249  !
;          3250  ! MSCP UNIT FLAGS
;          3251  !
;          3252  F_11 = uplit (%asciz'%N%A! Compare Reads'),
;          3253  F_12 = uplit (%asciz'%N%A! Compare Writes'),
;          3254  F_13 = uplit (%asciz'%N%A! Controller Initiated Bad Block Rplcmnt'),
;          3255  F_14 = uplit (%asciz'%N%A! Inactive Shadow Set Unit'),
;          3256  F_15 = uplit (%asciz'%N%A! Removable Media'),
;          3257  F_16 = uplit (%asciz'%N%A! Suppress Caching (high speed)'),
;          3258  F_17 = uplit (%asciz'%N%A! Suppress Caching (low speed)'),
;          3259  F_18 = uplit (%asciz'%N%A! Write-back (non-volatile)'),
;          3260  F_19 = uplit (%asciz'%N%A! Write Protect (hardware)'),
;          3261  F_20 = uplit (%asciz'%N%A! Write Protect (software or volume)'),
;          3262  F_21 = uplit (%asciz'%N%A! 576 Byte Sectors'),
;          3263  !
;          3264  ! DUP RETURN PACKET MESSAGES
;          3265  ! STATUS CODE
;          3266     EBH_30 = uplit (%asciz'%A - SUCCESS'),
;          3267     EBH_44 = UPLIT (%ASCIZ'%A - INVALID COMMAND(SERVER nonIDLE or no media if EX-LC-PRG cmd)'),
;          3268     EBH_45 = UPLIT (%ASCIZ'%A - NO REGION AVAILABLE'),
```

```
3269        EBH_46 = UPLIT (%ASCIZ'%A - NO REGION SUITABLE'),
3270        EBH_47 = UPLIT (%ASCIZ'%A - PROGRAM NOT KNOWN (NO SUCH PROGRAM ON MEDIA)'),
3271        EBH_48 = UPLIT (%ASCIZ'%A - LOAD FAILURE (INPUT ERROR WHILE LOADING PROGRAM)'),
3272        EBH_49 = UPLIT (%ASCIZ'%A - STANDALONE (STANDALONE MODIFIER NOT SPECIFIED FOR STAND ALONE PRG.)'),
3273
3274    !
3275    ! DUP GET DUST STATUS FLAGS
3276        df_0 = uplit (%asciz'%N%A! One Server at a Time'),
3277        df_1 = uplit (%asciz'%N%A! Contains Local Media'),
3278        df_2 = uplit (%asciz'%N%A! Execute Local Prg cmd is UNSUPPORTED'),
3279        df_3 = uplit (%asciz'%N%A! Currently in Active State'),
3280    !
3281    ! DUP EXECUTE LOCAL PRG END FLAGS
3282        df_4 = uplit (%asciz'%N%A! Standalone Prg'),
3283        df_5 = uplit (%asciz'%N%A! Needs overlay'),
3284        df_6 = uplit (%asciz'%N%A! Needs Writeable/Readable Overlay'),
3285        df_7 = uplit (%asciz'%N%A! Uses Std Dup Dialog; REC/SEND/REC'),
3286
3287    ! DUP LOCAL PROGRAM PACKET MESSAGES
3288    !
3289        T_QUE = uplit (%asciz'%N%A  ** QUESTION'),
3290        T_DEF = uplit (%asciz'%N%A  ** DEFAULT QUESTION'),
3291        T_INF = uplit (%asciz'%N%A  ** INFORMATION'),
3292        T_TER = uplit (%asciz'%N%A  ** TERMINATION'),
3293        T_FAT = uplit (%asciz'%N%A  ** FATAL ERROR'),
3294        T_SPL = uplit (%asciz'%N%A  ** SPECIAL'),
3295    !    E_SUC = UPLIT (%ASCIZ'%N%A - SUCCESS'),
3296        E_UNT = UPLIT (%ASCIZ'%N%A - ILLEGAL UNIT NUMBER'),
3297        E_BLK = UPLIT (%ASCIZ'%N%A - ILLEGAL PHYSICAL OR RELATIVE BLOCK NUMBER'),
3298        E_DEV = UPLIT (%ASCIZ'%N%A - DEVICE ERROR'),
3299        E_ZER = UPLIT (%ASCIZ'%N%A - ZERO LENGHT MESSAGE'),
3300        M_ASC = UPLIT (%ASCIZ'%N%A -- ASCII INFORMATION '),
3301        M_BIN = UPLIT (%ASCIZ'%N%A -- NON-ACSII INFORMATION'),
3302        M_TER = UPLIT (%ASCIZ'%N%A -- TERMINATION MESSAGE'),
3303        M_COD = UPLIT (%ASCIZ'%N%A -- SUCCESS/FAILURE CODE'),
3304        M_DAT = UPLIT (%ASCIZ'%N%A -- SEND BINARY DATA'),
3305        M_UR = UPLIT (%ASCIZ'%N%A -- SEND UNIT NUMBER, RELATIVE DBN'),
3306        M_URP = UPLIT (%ASCIZ'%N%A -- SEND UNIT NUMBER, RELATIVE DBN, WRITE PATTERN'),
3307        M_UP = UPLIT (%ASCIZ'%N%A -- SEND UNIT NUMBER, PHYSICAL BLOCK NUMBER'),
3308        M_UL = UPLIT (%ASCIZ'%N%A -- SEND UNIT NUMBER, LOGICAL  BLOCK NUMBER '),
3309    !
3310    ! CONTROLLER GENERIC ERROR CODES
3311    !
3312        CNTR_ERR = uplit (
3313                   uplit (%asciz'%ACONTROLLER TIMEOUT'),
3314                   uplit (%asciz'%AENVELOPE/PACKET READ ERROR (PARITY OR TIMEOUT)'),
3315                   uplit (%asciz'%AENVELOPE/PACKET WRITE ERROR (PARITY OR TIMEOUT)'),
3316                   uplit (%asciz'%ACONTROLLER ROM AND RAM PARITY ERROR'),
3317                   uplit (%asciz'%ACONTROLLER RAM PARITY ERROR'),
3318                   uplit (%asciz'%ACONTROLLER ROM PARITY ERROR'),
3319                   uplit (%asciz'%ARING READ ERROR (PARITY OR TIMEOUT)'),
3320                   uplit (%asciz'%ARING WRITE ERROR (PARITY OR TIMEOUT)'),
3321                   uplit (%asciz'INTERRUPT MASTER FAILURE'),
```

```
3322                    uplit (%asciz'%AHOST ACCESS TIMEOUT (HIGHER LEVEL PROTOCOL DEPENDENT)'),
3323                    uplit (%asciz'%ACREDIT LIMIT EXCEEDED'),
3324                    uplit (%asciz'%AQ-BUS MASTER ERROR'),
3325                    uplit (%asciz'%ACONTROLLER FATAL ERROR'),
3326                    uplit (%asciz'%AINSTRUCTION LOOP TIMEOUT'),
3327                    uplit (%asciz'%AILLEGAL VIRTUAL CIRCUIT ID'),
3328                    uplit (%asciz'%AINTERRUPT VECTOR ILLEGAL'),
3329                    uplit (%asciz'%AMAINTENANCE READ/WRITE INVALID REGION IDENTIFIER'),
3330                    uplit (%asciz'%AMAINTENANCE WRITE LOAD TO NON-LOADABLE CONTROLLER'),
3331                    uplit (%asciz'%ACONTROLLER RAM ERROR (NON-PARITY)'),
3332                    uplit (%asciz'%AINIT SEQUENCE ERROR'),
3333                    uplit (%asciz'%AHIGHER LEVEL PROTOCOL INCOMPATIBILITY ERROR'),
3334                    uplit (%asciz'%APURGE/POLL HARDWARE FAILURE'),
3335                    uplit (%asciz'%AMAPPING REGISTER READ FAILURE (PARITY OR TIMEOUT)')) : vector [23],
3336    !
3337    ! RD/RX CONTROLLER DEPENDENT ERRORS CODES
3338    !
3339        RDRX_ERR = uplit (
3340                    uplit (%asciz'%AT11 CPU FAILURE'),
3341                    uplit (%asciz'%ANON-PARITY RAM ERROR'),
3342                    uplit (%asciz'%ASTATE MACHINE FAILURE - T11 ADDRESS REGISTER'),
3343                    uplit (%asciz'%ASTATE MACHINE FAILURE - Q-BUS ADDRESS REGISTER'),
3344                    uplit (%asciz'%ASTATE MACHINE FAILURE - CRC REGISTER'),
3345                    uplit (%asciz'%ASTATE MACHINE FAILURE - SERIALIZER/DESERIALIZER REGISTER'),
3346                    uplit (%asciz'%ASTATE MACHINE FAILURE - WRONG HARDWARE VERSION')) : vector [7],
3347    !
3348    ! MISCELLANEOUS
3349    !
3350        EX_WRD = uplit (%asciz'%A %O6'),
3351        EX_OP = uplit (%asciz'%Aoct %O4'),
3352        SPACE4    = uplit (%asciz'%S4'),
3353        CRLF      = uplit (%asciz'%N'),
3354        DASH      = uplit (%asciz'%A - '),
3355        ASTERISK = uplit (%asciz'%A* ');
```

```
;        3356  %sbttl 'DEFAULT HARDWARE P-TABLE'
;        3357
;        3358  !+
;        3359  ! THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
;        3360  ! THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
;        3361  ! IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P-TABLES,
;        3362  ! AND IS USED AS A "TEMPLATE" FOR BUILDING THE P-TABLES.
;        3363  !-
;        3364
;        3365  BGNHW (DFPTBL);
;        3366
;        3367  global
;        3368      HWPT_IP_ADDR : word initial (INIT_IP_ADDR), ! IP ADDRESS
;        3369      HWPT_VECTOR : word initial (INIT_INTR_VECT),! VECTOR ADDRESS
;        3370      HWPT_BR_LEVEL : word initial (INIT_BR_LEVEL),!BR LEVEL
;        3371      HWPT_DISK : word initial (%o'100034'),      ! DISK NUMBER, TYPE, PROTECTON BIT
;        3372      HWPT_S_TRK : word initial (0),              ! STARTING TRACK
;        3373      HWPT_E_TRK : word initial (RD_MAX_LBN);     ! ENDING TRACK
;        3374
;        3375  ENDHW;
```

```
;          3376  %sbttl 'SOFTWARE P-TABLE'
;          3377
;          3378  !+
;          3379  ! THE SOFTWARE TABLE CONTAINS VARIOUS DATA USED BY THE
;          3380  ! PROGRAM AS OPERATIONAL PARAMETERS.  THESE PARAMETERS ARE
;          3381  ! SET UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR
;          3382  ! AT RUN TIME.
;          3383  !-
;          3384
;          3385  BGNSW (SFPTBL);
;          3386
;          3387  global
;          3388      SWP_ERROR : word initial (32),          ! HARD ERROR LIMIT FOR DROPPING UNIT
;          3389      SWP_XFER : word initial (20),           ! TRANSFER LIMIT FOR DROPPING UNIT
;          3390      SWP_FLAGS : word initial (%o'202'),     ! FLAGS (SEE DOCUMENTATION)
;          3391      SWP_DPAT : word initial (0),            ! DATA PATTERN NUMBER
;          3392      SWP_UCNT : word initial (MAX_UDP_CNT),  ! USER DATA PATTERN COUNT
;          3393      SWP_RAT : word initial (99),            ! RD51 OPERATION RATIO
;          3394      dupround : word initial (18),           ! NUMBER OF DBN'S WRITTEN AT ONE TIME


;          3395      SWP_UDPAT : vector [MAX_UDP_CNT, word];     ! USER DATA PATTERN
;          3396
;          3397  ENDSW;
```

```
;        3398  %sbttl 'PROTECTION TABLE'
;        3399
;        3400  !+
;        3401  ! THIS TABLE IS USED BY THE RUNTIME SERVICES
;        3402  ! TO PROTECT THE LOAD MEDIA.
;        3403  !-
;        3404
;        3405  BGNPROT (0, -1, 6);
;        3406
;        3407  !1ST ARG =      OFFSET INTO P-TABLE FOR CSR ADDRESS
;        3408  !2ND ARG =      OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
;        3409  !3RD ARG =      OFFSET INTO P-TABLE FOR DRIVE NUMBER
;        3410
;        3411  ENDPROT;
;        3412  end
;        3413
;        3414  eludom

                                        .TITLE  ZRQAM1 RD/RX EXERCISER
                                        .IDENT  /V01.2/
                                        .ENABL  AMA


000000                                  .PSECT  $CODE$,  RO
000000  132     122     121     L$NAME::.ASCII  /ZRQ/
000003  101                             .ASCII  /A/
000004  000                             .BYTE   0
000005  000                             .BYTE   0
000006  000                             .BYTE   0
000007  000                             .BYTE   0
000010                          L$REV::
000010  102                             .ASCII  /B/
000011  060                             .ASCII  /0/
000012  000000G                 L$UNIT::.WORD   T$PTHV
000014  077777                  L$TIML::.WORD   77777
000016  000000G                 L$HPCP::.WORD   L$HARD
000020  000000G                 L$SPCP::.WORD   L$SOFT
000022  024756'                  L$HPTP::.WORD   L$HW
000024  024776'                  L$SPTP::.WORD   L$SW
000026  000000G                 L$LADP::.WORD   L$LAST
000030  000000                  L$STA:: .WORD   0
000032  000000                  L$CO::  .WORD   0
000034  000001                  L$DTYP::.WORD   1
000036  000000                  L$APT:: .WORD   0
000040  000124'                 L$DTP:: .WORD   L$DISPATCH
000042  000000                  L$PRIO::.WORD   0
000044  000000                  L$ENVI::.WORD   0
000046  000000                  L$EXP1::.WORD   0
000050                          L$MREV::
000050  003                             .BYTE   3
000051  003                             .BYTE   3
000052  000000          L$EF::  .WORD   0
```

```
000054  000000                          .WORD   0
000056  000000                  L$SPC:: .WORD   0
000060  000000G                 L$DEVP::.WORD   L$DVTYP
000062  000000G                 L$REPP::.WORD   L$RPT
000064  000000                  L$EXP4::.WORD   0
000066  000000                  L$EXP5::.WORD   0
000070  000000G                 L$AUT:: .WORD   L$AU
000072  000000G                 L$DUT:: .WORD   L$DU
000074  000000                  L$LUN:: .WORD   0
000076  000000G                 L$DESP::.WORD   L$DESC
000100  104035                  L$LOAD::.WORD   -73743
000102  000126'                 L$ETP:: .WORD   L$ERRTBL
000104  000000G                 L$ICP:: .WORD   L$INIT
000106  000000G                 L$CCP:: .WORD   L$CLEAN
000110  000000G                 L$ACP:: .WORD   L$AUTO
000112  025056'                 L$PRT:: .WORD   L$PROT
000114  000000                  L$TEST::.WORD   0
000116  000000                  L$DLY:: .WORD   0
000120  000000                  L$HIME::.WORD   0
000122  000001                  D$PCNT::.WORD   1
000124  000000G                 L$DISPATCH::
                                        .WORD   T1
000126                          ERRTYP::.BLKW   1
000130                          ERRNBR::.BLKW   1
000132                          ERRMSG::.BLKW   1
000134                          ERRBLK::.BLKW   1
000136  111     120     040     P.AAA:  .ASCII  /IP /
000141  101     104     104             .ASCII  /ADD/
000144  122     105     123             .ASCII  /RES/
000147  123     000     000             .ASCII  /S/<00><00>
000152  126     105     103     P.AAB:  .ASCII  /VEC/
000155  124     117     122             .ASCII  /TOR/
000160  000     000                     .ASCII  <00><00>
000162  102     122     040     P.AAC:  .ASCII  /BR /
000165  114     105     126             .ASCII  /LEV/
000170  105     114     000             .ASCII  /EL/<00>
000173  000                             .ASCII  <00>
000174  122     104     057     P.AAD:  .ASCII  /RD/<57>
000177  122     130     040             .ASCII  /RX /
000202  104     122     111             .ASCII  /DRI/
000205  126     105     040             .ASCII  /VE /
000210  116     125     115             .ASCII  /NUM/
000213  102     105     122             .ASCII  /BER/
000216  000     000                     .ASCII  <00><00>
000220  111     123     040     P.AAE:  .ASCII  /IS /
000223  104     111     123             .ASCII  /DIS/
000226  113     040     101             .ASCII  /K A/
000231  116     040     122             .ASCII  /N R/
000234  104     065     061             .ASCII  /D51/
000237  040     127     111             .ASCII  / WI/
000242  116     103     110             .ASCII  /NCH/
000245  105     123     124             .ASCII  /EST/
000250  105     122     040             .ASCII  /ER /
```

```
000253  050  116  117                .ASCII  /(NO/
000256  054  040  111                .ASCII  /, I/
000261  115  120  114                .ASCII  /MPL/
000264  111  105  123                .ASCII  /IES/
000267  040  101  116                .ASCII  / AN/
000272  040  122  130                .ASCII  / RX/
000275  065  060  040                .ASCII  /50 /
000300  106  114  117                .ASCII  /FLO/
000303  120  120  131                .ASCII  /PPY/
000306  051  000                    .ASCII  /)/<00>
000310  123  124  101     P.AAF:     .ASCII  /STA/
000313  122  124  111                .ASCII  /RTI/
000316  116  107  040                .ASCII  /NG /
000321  114  102  116                .ASCII  /LBN/
000324  000  000                    .ASCII  <00><00>
000326  105  116  104     P.AAG:     .ASCII  /END/
000331  111  116  107                .ASCII  /ING/
000334  040  114  102                .ASCII  / LB/
000337  116  040  050                .ASCII  /N (/
000342  115  101  130                .ASCII  /MAX/
000345  111  115  125                .ASCII  /IMU/
000350  115  040  055                .ASCII  /M -/
000353  040  040  122                .ASCII  /  R/
000356  130  065  060                .ASCII  /X50/
000361  072  040  067                .ASCII  /: 7/
000364  071  071  054                .ASCII  /99,/
000367  040  122  104                .ASCII  / RD/
000372  065  061  072                .ASCII  /51:/
000375  040  062  061                .ASCII  / 21/
000400  065  071  071                .ASCII  /599/
000403  051  000  000                .ASCII  /)/<00><00>
000406  127  122  111     P.AAH:     .ASCII  /WRI/
000411  124  105  040                .ASCII  /TE /
000414  117  116  040                .ASCII  /ON /
000417  103  125  123                .ASCII  /CUS/
000422  124  117  115                .ASCII  /TOM/
000425  105  122  040                .ASCII  /ER /
000430  104  101  124                .ASCII  /DAT/
000433  101  040  101                .ASCII  /A A/
000436  122  105  101                .ASCII  /REA/
000441  040  117  116                .ASCII  / ON/
000444  040  124  110                .ASCII  / TH/
000447  111  123  040                .ASCII  /IS /
000452  104  111  123                .ASCII  /DIS/
000455  113  000  000                .ASCII  /K/<00><00>
000460  052  052  040     P.AAI:     .ASCII  /** /
000463  127  101  122                .ASCII  /WAR/
000466  116  111  116                .ASCII  /NIN/
000471  107  040  055                .ASCII  /G -/
000474  040  103  125                .ASCII  / CU/
000477  123  124  117                .ASCII  /STO/
000502  115  105  122                .ASCII  /MER/
000505  040  104  101                .ASCII  / DA/
```

```
000510  124  101  040                .ASCII  /TA /
000513  101  122  105                .ASCII  /ARE/
000516  101  040  115                .ASCII  /A M/
000521  101  131  040                .ASCII  /AY /
000524  102  105  040                .ASCII  /BE /
000527  117  126  105                .ASCII  /OVE/
000532  122  127  122                .ASCII  /RWR/
000535  111  124  124                .ASCII  /ITT/
000540  105  116  041                .ASCII  /EN!/
000543  040  056  056                .ASCII  / ../
000546  056  040  103                .ASCII  /. C/
000551  117  116  106                .ASCII  /ONF/
000554  111  122  115                .ASCII  /IRM/
000557  000                         .ASCII  <00>
000560  101  114  123      P.AAJ:    .ASCII  /ALS/
000563  117  040  105                .ASCII  /O E/
000566  130  105  122                .ASCII  /XER/
000571  103  111  123                .ASCII  /CIS/
000574  105  040  117                .ASCII  /E O/
000577  116  040  104                .ASCII  /N D/
000602  111  101  107                .ASCII  /IAG/
000605  116  117  123                .ASCII  /NOS/
000610  124  111  103                .ASCII  /TIC/
000613  040  101  122                .ASCII  / AR/
000616  105  101  040                .ASCII  /EA /
000621  050  055  116                .ASCII  /(-N/
000624  117  116  055                .ASCII  /ON-/
000627  103  125  123                .ASCII  /CUS/
000632  124  117  115                .ASCII  /TOM/
000635  105  122  040                .ASCII  /ER /
000640  101  122  105                .ASCII  /ARE/
000643  101  051  000                .ASCII  /A)/<00>
000646  127  122  111      P.AAK:    .ASCII  /WRI/
000651  124  105  040                .ASCII  /TE /
000654  117  116  040                .ASCII  /ON /
000657  104  111  101                .ASCII  /DIA/
000662  107  116  117                .ASCII  /GNO/
000665  123  124  111                .ASCII  /STI/
000670  103  040  101                .ASCII  /C A/
000673  122  105  101                .ASCII  /REA/
000676  000  000                     .ASCII  <00><00>
000700  110  101  122      P.AAL:    .ASCII  /HAR/
000703  104  040  105                .ASCII  /D E/
000706  122  122  117                .ASCII  /RRO/
000711  122  040  114                .ASCII  /R L/
000714  111  115  111                .ASCII  /IMI/
000717  124  000  000                .ASCII  /T/<00><00>
000722  124  122  101      P.AAM:    .ASCII  /TRA/
000725  116  123  106                .ASCII  /NSF/
000730  105  122  040                .ASCII  /ER /
000733  114  111  115                .ASCII  /LIM/
000736  111  124  040                .ASCII  /IT /
000741  111  116  040                .ASCII  /IN /
```

```
000744  115  105  107                .ASCII  /MEG/
000747  101  102  131                .ASCII  /ABY/
000752  124  105  123                .ASCII  /TES/
000755  040  050  060                .ASCII  / (0/
000760  040  106  117                .ASCII  / FO/
000763  122  040  042                .ASCII  /R "/
000766  121  125  111                .ASCII  /QUI/
000771  103  113  040                .ASCII  /CK /
000774  120  101  123                .ASCII  /PAS/
000777  123  042  051                .ASCII  /S")/
001002  000  000                    .ASCII  <00><00>
001004  122  101  116      P.AAN:    .ASCII  /RAN/
001007  104  117  115                .ASCII  /DOM/
001012  040  123  105                .ASCII  / SE/
001015  105  113  040                .ASCII  /EK /
001020  115  117  104                .ASCII  /MOD/
001023  105  000  000                .ASCII  /E/<00><00>
001026  122  105  101      P.AAO:    .ASCII  /REA/
001031  104  055  103                .ASCII  /D-C/
001034  117  115  120                .ASCII  /OMP/
001037  101  122  105                .ASCII  /ARE/
001042  123  040  120                .ASCII  /S P/
001045  105  122  106                .ASCII  /ERF/
001050  117  122  115                .ASCII  /ORM/
001053  105  104  040                .ASCII  /ED /
001056  101  124  040                .ASCII  /AT /
001061  124  110  105                .ASCII  /THE/
001064  040  103  117                .ASCII  / CO/
001067  116  124  122                .ASCII  /NTR/
001072  117  114  114                .ASCII  /OLL/
001075  105  122  000                .ASCII  /ER/<00>
001100  127  122  111      P.AAP:    .ASCII  /WRI/
001103  124  105  055                .ASCII  /TE-/
001106  103  117  115                .ASCII  /COM/
001111  120  101  122                .ASCII  /PAR/
001114  105  123  040                .ASCII  /ES /
001117  120  105  122                .ASCII  /PER/
001122  106  117  122                .ASCII  /FOR/
001125  115  105  104                .ASCII  /MED/
001130  040  101  124                .ASCII  / AT/
001133  040  124  110                .ASCII  / TH/
001136  105  040  103                .ASCII  /E C/
001141  117  116  124                .ASCII  /ONT/
001144  122  117  114                .ASCII  /ROL/
001147  114  105  122                .ASCII  /LER/
001152  000  000                    .ASCII  <00><00>
001154  103  110  105      P.AAQ:    .ASCII  /CHE/
001157  103  113  040                .ASCII  /CK /
001162  101  114  114                .ASCII  /ALL/
001165  040  127  122                .ASCII  / WR/
001170  111  124  105                .ASCII  /ITE/
001173  123  040  101                .ASCII  /S A/
001176  124  040  110                .ASCII  /T H/
```

```
001201  117  123  124          .ASCII  /OST/
001204  040  102  131          .ASCII  / BY/
001207  040  122  105          .ASCII  / RE/
001212  101  104  111          .ASCII  /ADI/
001215  116  107  000          .ASCII  /NG/<00>
001220  125  123  105  P.AAR:   .ASCII  /USE/
001223  122  055  104          .ASCII  /R-D/
001226  105  106  111          .ASCII  /EFI/
001231  116  105  104          .ASCII  /NED/
001234  040  104  101          .ASCII  / DA/
001237  124  101  040          .ASCII  /TA /
001242  120  101  124          .ASCII  /PAT/
001245  124  105  122          .ASCII  /TER/
001250  116  000               .ASCII  /N/<00>
001252  123  105  114  P.AAS:   .ASCII  /SEL/
001255  105  103  124          .ASCII  /ECT/
001260  040  120  122          .ASCII  / PR/
001263  105  055  104          .ASCII  /E-D/
001266  105  106  111          .ASCII  /EFI/
001271  116  105  104          .ASCII  /NED/
001274  040  104  101          .ASCII  / DA/
001277  124  101  040          .ASCII  /TA /
001302  120  101  124          .ASCII  /PAT/
001305  124  105  122          .ASCII  /TER/
001310  116  040  050          .ASCII  /N (/
001313  060  040  106          .ASCII  /0 F/
001316  117  122  040          .ASCII  /OR /
001321  123  105  121          .ASCII  /SEQ/
001324  125  105  116          .ASCII  /UEN/
001327  124  111  101          .ASCII  /TIA/
001332  114  040  123          .ASCII  /L S/
001335  105  114  105          .ASCII  /ELE/
001340  103  124  111          .ASCII  /CTI/
001343  117  116  051          .ASCII  /ON)/
001346  000  000               .ASCII  <00><00>
001350  116  125  115  P.AAT:   .ASCII  /NUM/
001353  102  105  122          .ASCII  /BER/
001356  040  117  106          .ASCII  / OF/
001361  040  127  117          .ASCII  / WO/
001364  122  104  123          .ASCII  /RDS/
001367  040  111  116          .ASCII  / IN/
001372  040  104  101          .ASCII  / DA/
001375  124  101  040          .ASCII  /TA /
001400  120  101  124          .ASCII  /PAT/
001403  124  105  122          .ASCII  /TER/
001406  116  040  050          .ASCII  /N (/
001411  061  066  040          .ASCII  /16 /
001414  115  101  130          .ASCII  /MAX/
001417  111  115  125          .ASCII  /IMU/
001422  115  051  000          .ASCII  /M)/<00>
001425  000                    .ASCII  <00>
001426  120  101  124  P.AAU:   .ASCII  /PAT/
001431  124  105  122          .ASCII  /TER/
```

```
001434  116  040  126          .ASCII  /N V/
001437  101  114  125          .ASCII  /ALU/
001442  105  000               .ASCII  /E/<00>
001444  103  114  105  P.AAV:   .ASCII  /CLE/
001447  101  122  040          .ASCII  /AR /
001452  123  124  101          .ASCII  /STA/
001455  124  111  123          .ASCII  /TIS/
001460  124  111  103          .ASCII  /TIC/
001463  101  114  040          .ASCII  /AL /
001466  124  101  102          .ASCII  /TAB/
001471  114  105  123          .ASCII  /LES/
001474  040  101  106          .ASCII  / AF/
001477  124  105  122          .ASCII  /TER/
001502  040  120  122          .ASCII  / PR/
001505  111  116  124          .ASCII  /INT/
001510  111  116  107          .ASCII  /ING/
001513  000                    .ASCII  <00>
001514  120  105  122  P.AAW:   .ASCII  /PER/
001517  103  105  116          .ASCII  /CEN/
001522  124  101  107          .ASCII  /TAG/
001525  105  040  117          .ASCII  /E O/
001530  106  040  122          .ASCII  /F R/
001533  104  065  061          .ASCII  /D51/
001536  040  117  120          .ASCII  / OP/
001541  105  122  101          .ASCII  /ERA/
001544  124  111  117          .ASCII  /TIO/
001547  116  123  040          .ASCII  /NS /
001552  117  125  124          .ASCII  /OUT/
001555  040  117  106          .ASCII  / OF/
001560  040  124  117          .ASCII  / TO/
001563  124  101  114          .ASCII  /TAL/
001566  040  117  120          .ASCII  / OP/
001571  105  122  101          .ASCII  /ERA/
001574  124  111  117          .ASCII  /TIO/
001577  116  123  000          .ASCII  /NS/<00>
001602  125  116  111  P.AAX:   .ASCII  /UNI/
001605  124  123  040          .ASCII  /TS /
001610  124  117  040          .ASCII  /TO /
001613  102  105  040          .ASCII  /BE /
001616  123  105  114          .ASCII  /SEL/
001621  105  103  124          .ASCII  /ECT/
001624  105  104  040          .ASCII  /ED /
001627  101  124  040          .ASCII  /AT /
001632  122  101  116          .ASCII  /RAN/
001635  104  117  115          .ASCII  /DOM/
001640  040  050  116          .ASCII  / (N/
001643  117  054  040          .ASCII  /O, /
001646  111  115  120          .ASCII  /IMP/
001651  114  111  105          .ASCII  /LIE/
001654  123  040  123          .ASCII  /S S/
001657  105  121  125          .ASCII  /EQU/
001662  105  116  124          .ASCII  /ENT/
001665  111  101  114          .ASCII  /IAL/
```

```
001670  051  000                    .ASCII  /)/<00>
001672  116  125  115       P.AAY:  .ASCII  /NUM/
001675  102  105  122               .ASCII  /BER/
001700  040  117  106               .ASCII  / OF/
001703  040  104  102               .ASCII  / DB/
001706  116  163  040               .ASCII  /Ns /
001711  122  105  101               .ASCII  /REA/
001714  104  040  101               .ASCII  /D A/
001717  124  040  117               .ASCII  /T O/
001722  116  105  040               .ASCII  /NE /
001725  124  111  115               .ASCII  /TIM/
001730  105  040  050               .ASCII  /E (/
001733  145  146  146               .ASCII  /eff/
001736  145  143  164               .ASCII  /ect/
001741  163  040  122               .ASCII  /s R/
001744  104  065  061               .ASCII  /D51/
001747  163  040  157               .ASCII  /s o/
001752  156  154  171               .ASCII  /nly/
001755  051  000  000               .ASCII  /)/<00><00>
001760  124  110  105       P.AAZ:  .ASCII  /THE/
001763  040  122  105               .ASCII  / RE/
001766  115  101  111               .ASCII  /MAI/
001771  116  111  116               .ASCII  /NIN/
001774  107  040  121               .ASCII  /G Q/
001777  125  105  123               .ASCII  /UES/
002002  124  111  117               .ASCII  /TIO/
002005  116  123  040               .ASCII  /NS /
002010  117  116  114               .ASCII  /ONL/
002013  131  040  101               .ASCII  /Y A/
002016  120  120  114               .ASCII  /PPL/
002021  131  040  124               .ASCII  /Y T/
002024  117  040  125               .ASCII  /O U/
002027  116  120  122               .ASCII  /NPR/
002032  117  124  105               .ASCII  /OTE/
002035  103  124  105               .ASCII  /CTE/
002040  104  040  104               .ASCII  /D D/
002043  111  123  113               .ASCII  /ISK/
002046  123  000                    .ASCII  /S/<00>
002050  000  000            P.ABA:  .ASCII  <00><00>
002052  045  116  045       P.ABB:  .ASCII  /%N%/
002055  101  052  052               .ASCII  /A**/
002060  040  104  122               .ASCII  / DR/
002063  117  120  040               .ASCII  /OP /
002066  125  116  111               .ASCII  /UNI/
002071  124  040  045               .ASCII  /T %/
002074  104  062  000               .ASCII  /D2/<00>
002077  000                        .ASCII  <00>
002100  045  116  045       P.ABC:  .ASCII  /%N%/
002103  101  052  052               .ASCII  /A**/
002106  040  120  122               .ASCII  / PR/
002111  117  103  137               .ASCII  /OC_/
002114  122  105  124               .ASCII  /RET/
002117  120  113  124               .ASCII  /PKT/
```

```
002122  072     040     103             .ASCII  /: C/
002125  117     116     116             .ASCII  /ONN/
002130  040     111     104             .ASCII  / ID/
002133  040     075     040             .ASCII  / = /
002136  045     104     065             .ASCII  /%D5/
002141  045     101     040             .ASCII  /%A /
002144  122     105     103             .ASCII  /REC/
002147  105     111     126             .ASCII  /EIV/
002152  105     104     000             .ASCII  /ED/<00>
002155  000                             .ASCII  <00>
002156  045     116     045     P.ABD:  .ASCII  /%N%/
002161  101     052     052             .ASCII  /A**/
002164  040     106     101             .ASCII  / FA/
002167  124     101     114             .ASCII  /TAL/
002172  137     105     122             .ASCII  / ER/
002175  122     117     122             .ASCII  /ROR/
002200  072     040     122             .ASCII  /: R/
002203  105     124     120             .ASCII  /ETP/
002206  113     124     040             .ASCII  /KT /
002211  116     117     124             .ASCII  /NOT/
002214  040     101     126             .ASCII  / AV/
002217  101     111     114             .ASCII  /AIL/
002222  101     102     114             .ASCII  /ABL/
002225  105     000     000             .ASCII  /E/<00><00>
002230  045     116     045     P.ABE:  .ASCII  /%N%/
002233  101     052     052             .ASCII  /A**/
002236  040     106     123             .ASCII  / FS/
002241  105     124     137             .ASCII  /ET_/
002244  125     120     101             .ASCII  /UPA/
002247  122     072     040             .ASCII  /R: /
002252  103     101     116             .ASCII  /CAN/
002255  047     124     040             .ASCII  /'T /
002260  106     111     116             .ASCII  /FIN/
002263  104     040     104             .ASCII  /D D/
002266  111     123     113             .ASCII  /ISK/
002271  040     045     104             .ASCII  / %D/
002274  063     045     101             .ASCII  /3%A/
002277  040     111     116             .ASCII  / IN/
002302  040     103     123             .ASCII  / CS/
002305  124     040     045             .ASCII  /T %/
002310  104     061     000             .ASCII  /D1/<00>
002313  000                             .ASCII  <00>
002314  045     116     045     P.ABF:  .ASCII  /%N%/
002317  101     052     052             .ASCII  /A**/
002322  040     102     101             .ASCII  / BA/
002325  104     040     103             .ASCII  /D C/
002330  117     116     116             .ASCII  /ONN/
002333  040     111     104             .ASCII  / ID/
002336  040     075     040             .ASCII  / = /
002341  045     104     065             .ASCII  /%D5/
002344  045     101     040             .ASCII  /%A /
002347  122     105     103             .ASCII  /REC/
002352  105     111     126             .ASCII  /EIV/
```

```
002355  105  104  040                .ASCII  /ED /
002360  106  122  117                .ASCII  /FRO/
002363  115  040  045                .ASCII  /M %/
002366  117  066  000                .ASCII  /06/<00>
002371  000                         .ASCII  <00>
002372  045  116  045        P.ABG:  .ASCII  /%N%/
002375  101  052  052                .ASCII  /A**/
002400  040  115  105                .ASCII  / ME/
002403  123  123  101                .ASCII  /SSA/
002406  107  105  040                .ASCII  /GE /
002411  124  131  120                .ASCII  /TYP/
002414  105  040  045                .ASCII  /E %/
002417  104  062  045                .ASCII  /D2%/
002422  101  040  122                .ASCII  /A R/
002425  105  103  105                .ASCII  /ECE/
002430  111  126  105                .ASCII  /IVE/
002433  104  040  111                .ASCII  /D I/
002436  116  040  115                .ASCII  /N M/
002441  123  103  120                .ASCII  /SCP/
002444  040  120  101                .ASCII  / PA/
002447  103  113  105                .ASCII  /CKE/
002452  124  000                    .ASCII  /T/<00>
002454  045  116  045        P.ABH:  .ASCII  /%N%/
002457  101  052  052                .ASCII  /A**/
002462  040  105  122                .ASCII  / ER/
002465  122  117  122                .ASCII  /ROR/
002470  040  111  116                .ASCII  / IN/
002473  040  123  105                .ASCII  / SE/
002476  124  137  103                .ASCII  /T_C/
002501  124  114  122                .ASCII  /TLR/
002504  137  103  110                .ASCII  /_CH/
002507  101  122  000                .ASCII  /AR/<00>
002512  045  116  045        P.ABI:  .ASCII  /%N%/
002515  101  052  052                .ASCII  /A**/
002520  040  103  124                .ASCII  / CT/
002523  114  122  040                .ASCII  /LR /
002526  124  111  115                .ASCII  /TIM/
002531  105  117  125                .ASCII  /EOU/
002534  124  040  075                .ASCII  /T =/
002537  040  045  104                .ASCII  / %D/
002542  063  045  101                .ASCII  /3%A/
002545  040  123  105                .ASCII  / SE/
002550  103  117  116                .ASCII  /CON/
002553  104  123  000                .ASCII  /DS/<00>
002556  045  116  045        P.ABJ:  .ASCII  /%N%/
002561  101  052  052                .ASCII  /A**/
002564  040  105  122                .ASCII  / ER/
002567  122  117  122                .ASCII  /ROR/
002572  040  111  116                .ASCII  / IN/
002575  040  125  116                .ASCII  / UN/
002600  111  124  137                .ASCII  /IT_/
002603  111  116  111                .ASCII  /INI/
002606  124  000                    .ASCII  /T/<00>
```

```
002610  045  116  045        P.ABK:  .ASCII  /%N%/
002613  101  052  052                .ASCII  /A**/
002616  040  101  103                .ASCII  / AC/
002621  103  105  123                .ASCII  /CES/
002624  123  072  040                .ASCII  /S: /
002627  122  105  124                .ASCII  /RET/
002632  120  113  124                .ASCII  /PKT/
002635  040  110  101                .ASCII  / HA/
002640  123  040  102                .ASCII  /S B/
002643  101  104  040                .ASCII  /AD /
002646  105  116  104                .ASCII  /END/
002651  103  117  104                .ASCII  /COD/
002654  105  000                    .ASCII  /E/<00>
002656  045  116  045        P.ABL:  .ASCII  /%N%/
002661  101  052  052                .ASCII  /A**/
002664  040  111  114                .ASCII  / IL/
002667  114  105  107                .ASCII  /LEG/
002672  101  114  040                .ASCII  /AL /
002675  106  125  116                .ASCII  /FUN/
002700  103  124  111                .ASCII  /CTI/
002703  117  116  072                .ASCII  /ON:/
002706  040  045  117                .ASCII  / %O/
002711  066  000  000                .ASCII  /6/<00><00>
002714  045  116  045        P.ABM:  .ASCII  /%N%/
002717  101  052  052                .ASCII  /A**/
002722  040  103  117                .ASCII  / CO/
002725  115  115  101                .ASCII  /MMA/
002730  116  104  040                .ASCII  /ND /
002733  122  105  106                .ASCII  /REF/
002736  040  043  040                .ASCII  / # /
002741  045  104  065                .ASCII  /%D5/
002744  045  101  056                .ASCII  /%A./
002747  040  116  117                .ASCII  / NO/
002752  124  040  123                .ASCII  /T S/
002755  105  116  124                .ASCII  /ENT/
002760  040  102  131                .ASCII  / BY/
002763  040  110  117                .ASCII  / HO/
002766  123  124  000                .ASCII  /ST/<00>
002771  000                        .ASCII  <00>
002772  045  116  045        P.ABN:  .ASCII  /%N%/
002775  101  052  052                .ASCII  /A**/
003000  040  125  116                .ASCII  / UN/
003003  113  116  117                .ASCII  /KNO/
003006  127  116  040                .ASCII  /WN /
003011  105  122  122                .ASCII  /ERR/
003014  117  122  040                .ASCII  /OR /
003017  114  117  107                .ASCII  /LOG/
003022  040  106  117                .ASCII  / FO/
003025  122  115  101                .ASCII  /RMA/
003030  124  040  045                .ASCII  /T %/
003033  104  063  045                .ASCII  /D3%/
003036  101  056  040                .ASCII  /A. /
003041  122  105  103                .ASCII  /REC/
```

```
003044  105  111  126                .ASCII  /EIV/
003047  105  104  000                .ASCII  /ED/<00>
003052  045  116  045     P.AB0:     .ASCII  /%N%/
003055  101  052  052                .ASCII  /A**/
003060  040  105  122                .ASCII  / ER/
003063  122  117  122                .ASCII  /ROR/
003066  055  114  117                .ASCII  /-LO/
003071  107  040  123                .ASCII  /G S/
003074  101  126  105                .ASCII  /AVE/
003077  040  101  122                .ASCII  / AR/
003102  105  101  040                .ASCII  /EA /
003105  106  125  114                .ASCII  /FUL/
003110  114  000                    .ASCII  /L/<00>
003112  045  116  045     P.ABP:     .ASCII  /%N%/
003115  101  052  052                .ASCII  /A**/
003120  040  104  125                .ASCII  / DU/
003123  120  137  122                .ASCII  /P_R/
003126  123  120  072                .ASCII  /SP:/
003131  040  122  105                .ASCII  / RE/
003134  124  120  113                .ASCII  /TPK/
003137  124  040  116                .ASCII  /T N/
003142  117  124  040                .ASCII  /OT /
003145  101  126  101                .ASCII  /AVA/
003150  111  114  101                .ASCII  /ILA/
003153  102  114  105                .ASCII  /BLE/
003156  000  000                    .ASCII  <00><00>
003160  045  116  045     P.ABQ:     .ASCII  /%N%/
003163  101  125  116                .ASCII  /AUN/
003166  111  124  045                .ASCII  /IT%/
003171  104  062  045                .ASCII  /D2%/
003174  101  040  104                .ASCII  /A D/
003177  122  117  120                .ASCII  /ROP/
003202  120  105  104                .ASCII  /PED/
003205  040  055  040                .ASCII  / - /
003210  000  000                    .ASCII  <00><00>
003212  045  101  125     P.ABS:     .ASCII  /%AU/
003215  123  105  122                .ASCII  /SER/
003220  040  103  117                .ASCII  / CO/
003223  115  115  101                .ASCII  /MMA/
003226  116  104  045                .ASCII  /ND%/
003231  116  000  000                .ASCII  /N/<00><00>
003234  045  101  103     P.ABT:     .ASCII  /%AC/
003237  117  116  106                .ASCII  /ONF/
003242  111  107  125                .ASCII  /IGU/
003245  122  101  124                .ASCII  /RAT/
003250  111  117  116                .ASCII  /ION/
003253  040  105  122                .ASCII  / ER/
003256  122  117  122                .ASCII  /ROR/
003261  045  116  000                .ASCII  /%N/<00>
003264  045  101  111     P.ABU:     .ASCII  /%AI/
003267  116  111  124                .ASCII  /NIT/
003272  040  105  122                .ASCII  / ER/
003275  122  117  122                .ASCII  /ROR/
```

```
003300  045  116  000                .ASCII  /%N/<00>
003303  000                         .ASCII  <00>
003304  045  101  124       P.ABV:   .ASCII  /%AT/
003307  122  101  116                .ASCII  /RAN/
003312  123  106  105                .ASCII  /SFE/
003315  122  040  114                .ASCII  /R L/
003320  111  115  111                .ASCII  /IMI/
003323  124  040  122                .ASCII  /T R/
003326  105  101  103                .ASCII  /EAC/
003331  110  105  104                .ASCII  /HED/
003334  045  116  000                .ASCII  /%N/<00>
003337  000                         .ASCII  <00>
003340  045  101  105       P.ABW:   .ASCII  /%AE/
003343  122  122  117                .ASCII  /RRO/
003346  122  040  114                .ASCII  /R L/
003351  111  115  111                .ASCII  /IMI/
003354  124  040  122                .ASCII  /T R/
003357  105  101  103                .ASCII  /EAC/
003362  110  105  104                .ASCII  /HED/
003365  045  116  000                .ASCII  /%N/<00>
003370  045  101  125       P.ABX:   .ASCII  /%AU/
003373  116  122  105                .ASCII  /NRE/
003376  103  117  126                .ASCII  /COV/
003401  105  122  101                .ASCII  /ERA/
003404  102  114  105                .ASCII  /BLE/
003407  040  104  105                .ASCII  / DE/
003412  126  111  103                .ASCII  /VIC/
003415  105  040  105                .ASCII  /E E/
003420  122  122  117                .ASCII  /RRO/
003423  122  045  116                .ASCII  /R%N/
003426  000  000                    .ASCII  <00><00>
003430  045  101  125       P.ABY:   .ASCII  /%AU/
003433  116  122  105                .ASCII  /NRE/
003436  103  117  126                .ASCII  /COV/
003441  105  122  101                .ASCII  /ERA/
003444  102  114  105                .ASCII  /BLE/
003447  040  103  117                .ASCII  / CO/
003452  116  124  122                .ASCII  /NTR/
003455  117  114  114                .ASCII  /OLL/
003460  105  122  040                .ASCII  /ER /
003463  105  122  122                .ASCII  /ERR/
003466  117  122  045                .ASCII  /OR%/
003471  116  000  000                .ASCII  /N/<00><00>
003474  045  101  106       P.ABZ:   .ASCII  /%AF/
003477  101  111  114                .ASCII  /AIL/
003502  105  104  040                .ASCII  /ED /
003505  124  117  040                .ASCII  /TO /
003510  103  117  115                .ASCII  /COM/
003513  105  040  117                .ASCII  /E O/
003516  116  114  111                .ASCII  /NLI/
003521  116  105  045                .ASCII  /NE%/
003524  116  000                    .ASCII  /N/<00>
003526  045  101  106       P.ACA:   .ASCII  /%AF/
```

```
003531  101  111  114                .ASCII  /AIL/
003534  105  104  040                .ASCII  /ED /
003537  124  117  040                .ASCII  /TO /
003542  101  103  103                .ASCII  /ACC/
003545  105  123  123                .ASCII  /ESS/
003550  040  114  101                .ASCII  / LA/
003553  123  124  040                .ASCII  /ST /
003556  124  122  101                .ASCII  /TRA/
003561  103  113  040                .ASCII  /CK /
003564  104  125  122                .ASCII  /DUR/
003567  111  116  107                .ASCII  /ING/
003572  040  111  116                .ASCII  / IN/
003575  111  124  045                .ASCII  /IT%/
003600  116  000                    .ASCII  /N/<00>
003602  045  101  104       P.ACB:   .ASCII  /%AD/
003605  111  123  113                .ASCII  /ISK/
003610  040  127  122                .ASCII  / WR/
003613  111  124  105                .ASCII  /ITE/
003616  040  120  122                .ASCII  / PR/
003621  117  124  105                .ASCII  /OTE/
003624  103  124  105                .ASCII  /CTE/
003627  104  045  116                .ASCII  /D%N/
003632  000  000                    .ASCII  <00><00>
003634  045  101  103       P.ACC:   .ASCII  /%AC/
003637  117  115  115                .ASCII  /OMM/
003642  101  116  104                .ASCII  /AND/
003645  040  124  111                .ASCII  / TI/
003650  115  105  040                .ASCII  /ME /
003653  117  125  124                .ASCII  /OUT/
003656  045  116  000                .ASCII  /%N/<00>
003661  000                         .ASCII  <00>
003662  045  101  125       P.ACD:   .ASCII  /%AU/
003665  116  111  124                .ASCII  /NIT/
003670  040  127  105                .ASCII  / WE/
003673  116  124  040                .ASCII  /NT /
003676  124  117  040                .ASCII  /TO /
003701  101  126  101                .ASCII  /AVA/
003704  111  114  101                .ASCII  /ILA/
003707  102  114  105                .ASCII  /BLE/
003712  040  123  124                .ASCII  / ST/
003715  101  124  105                .ASCII  /ATE/
003720  045  116  000                .ASCII  /%N/<00>
003723  000                         .ASCII  <00>
003724  003212'      P.ABR:   .WORD   P.ABS
003726  003234'               .WORD   P.ABT
003730  003264'               .WORD   P.ABU
003732  003304'               .WORD   P.ABV
003734  003340'               .WORD   P.ABW
003736  003370'               .WORD   P.ABX
003740  003430'               .WORD   P.ABY
003742  003474'               .WORD   P.ABZ
003744  003526'               .WORD   P.ACA
003746  003602'               .WORD   P.ACB
```

```
003750  003634'                         .WORD   P.ACC
003752  003662'                         .WORD   P.ACD
003754     045     116     045  P.ACE:  .ASCII  /%N%/
003757     101     120     117         .ASCII  /APO/
003762     127     105     122         .ASCII  /WER/
003765     040     104     105         .ASCII  / DE/
003770     114     101     131         .ASCII  /LAY/
003773     040     055     040         .ASCII  / - /
003776     127     101     111         .ASCII  /WAI/
004001     124     111     116         .ASCII  /TIN/
004004     107     000                 .ASCII  /G/<00>
004006     045     116     045  P.ACF:  .ASCII  /%N%/
004011     101     106     125         .ASCII  /AFU/
004014     116     103     124         .ASCII  /NCT/
004017     111     117     116         .ASCII  /ION/
004022     101     114     040         .ASCII  /AL /
004025     124     105     123         .ASCII  /TES/
004030     124     040     123         .ASCII  /T S/
004033     124     101     122         .ASCII  /TAR/
004036     124     105     104         .ASCII  /TED/
004041     000                         .ASCII  <00>
004042     045     116     045  P.ACG:  .ASCII  /%N%/
004045     116     045     101         .ASCII  /N%A/
004050     105     130     105         .ASCII  /EXE/
004053     122     103     111         .ASCII  /RCI/
004056     123     105     122         .ASCII  /SER/
004061     040     123     124         .ASCII  / ST/
004064     101     122     124         .ASCII  /ART/
004067     105     104     045         .ASCII  /ED%/
004072     116     000                 .ASCII  /N/<00>
004074     045     116     045  P.ACH:  .ASCII  /%N%/
004077     116     045     101         .ASCII  /N%A/
004102     125     116     124         .ASCII  /UNT/
004105     040     104     123         .ASCII  / DS/
004110     113     045     123         .ASCII  /K%S/
004113     070     045     101         .ASCII  /8%A/
004116     043     040     117         .ASCII  /# O/
004121     106     040     040         .ASCII  /F  /
004124     040     043     040         .ASCII  / # /
004127     102     131     124         .ASCII  /BYT/
004132     105     123     040         .ASCII  /ES /
004135     040     040     043         .ASCII  /  #/
004140     040     117     106         .ASCII  / OF/
004143     040     040     040         .ASCII  /   /
004146     040     043     040         .ASCII  / # /
004151     102     131     124         .ASCII  /BYT/
004154     105     123     000         .ASCII  /ES/<00>
004157     000                         .ASCII  <00>
004160     045     101     040  P.ACI:  .ASCII  /%A /
004163     040     055     055         .ASCII  / --/
004166     110     101     122         .ASCII  /HAR/
004171     104     040     105         .ASCII  /D E/
004174     122     122     117         .ASCII  /RRO/
```

```
004177  122  123  055                .ASCII  /RS-/
004202  055  040  055                .ASCII  /- -/
004205  055  123  117                .ASCII  /-SO/
004210  106  124  040                .ASCII  /FT /
004213  105  122  122                .ASCII  /ERR/
004216  117  122  123                .ASCII  /ORS/
004221  055  055  000                .ASCII  /--/<00>
004224  045  116  045      P.ACJ:    .ASCII  /%N%/
004227  101  040  043                .ASCII  /A #/
004232  040  040  040                .ASCII  /   /
004235  043  040  040                .ASCII  /#  /
004240  124  131  120                .ASCII  /TYP/
004243  105  040  040                .ASCII  /E  /
004246  122  105  101                .ASCII  /REA/
004251  104  123  040                .ASCII  /DS /
004254  040  040  040                .ASCII  /   /
004257  040  122  105                .ASCII  / RE/
004262  101  104  040                .ASCII  /AD /
004265  040  040  127                .ASCII  /  W/
004270  122  111  124                .ASCII  /RIT/
004273  105  123  040                .ASCII  /ES /
004276  040  040  127                .ASCII  /  W/
004301  122  111  124                .ASCII  /RIT/
004304  124  105  116                .ASCII  /TEN/
004307  000                         .ASCII  <00>
004310  045  101  040      P.ACK:    .ASCII  /%A /
004313  040  123  105                .ASCII  / SE/
004316  113  040  104                .ASCII  /K D/
004321  101  124  040                .ASCII  /AT /
004324  104  122  126                .ASCII  /DRV/
004327  040  110  123                .ASCII  / HS/
004332  124  040  123                .ASCII  /T S/
004335  105  113  040                .ASCII  /EK /
004340  104  101  124                .ASCII  /DAT/
004343  040  104  122                .ASCII  / DR/
004346  126  040  110                .ASCII  /V H/
004351  123  124  000                .ASCII  /ST/<00>
004354  045  116  045      P.ACL:    .ASCII  /%N%/
004357  101  055  055                .ASCII  /A--/
004362  055  040  055                .ASCII  /- -/
004365  055  055  040                .ASCII  /-- /
004370  055  055  055                .ASCII  /---/
004373  055  040  040                .ASCII  /-  /
004376  055  055  055                .ASCII  /---/
004401  055  055  040                .ASCII  /-- /
004404  040  055  055                .ASCII  / --/
004407  055  055  055                .ASCII  /---/
004412  055  055  055                .ASCII  /---/
004415  055  040  055                .ASCII  /- -/
004420  055  055  055                .ASCII  /---/
004423  055  055  040                .ASCII  /-- /
004426  040  055  055                .ASCII  / --/
004431  055  055  055                .ASCII  /---/
```

```
004434  055  055  055                .ASCII  /---/
004437  055  000  000                .ASCII  /-/<00><00>
004442  045  101  040      P.ACM:    .ASCII  /%A /
004445  055  055  055                .ASCII  /---/
004450  040  055  055                .ASCII  / --/
004453  055  040  055                .ASCII  /- -/
004456  055  055  040                .ASCII  /-- /
004461  055  055  055                .ASCII  /---/
004464  040  055  055                .ASCII  / --/
004467  055  040  055                .ASCII  /- -/
004472  055  055  040                .ASCII  /-- /
004475  055  055  055                .ASCII  /---/
004500  040  055  055                .ASCII  / --/
004503  055  000  000                .ASCII  /-/<00><00>
004506  045  116  045      P.ACN:    .ASCII  /%N%/
004511  104  062  045                .ASCII  /D2%/
004514  104  064  045                .ASCII  /D4%/
004517  101  040  040                .ASCII  /A  /
004522  122  130  065                .ASCII  /RX5/
004525  060  000  000                .ASCII  /0/<00><00>
004530  045  116  045      P.ACO:    .ASCII  /%N%/
004533  104  062  045                .ASCII  /D2%/
004536  104  064  045                .ASCII  /D4%/
004541  101  040  040                .ASCII  /A  /
004544  122  104  065                .ASCII  /RD5/
004547  061  000  000                .ASCII  /1/<00><00>
004552  045  104  064      P.ACP:    .ASCII  /%D4/
004555  045  132  063                .ASCII  /%Z3/
004560  045  104  063                .ASCII  /%D3/
004563  045  101  054                .ASCII  /%A,/
004566  045  132  063                .ASCII  /%Z3/
004571  045  101  054                .ASCII  /%A,/
004574  045  132  063                .ASCII  /%Z3/
004577  000                         .ASCII  <00>
004600  045  104  064      P.ACQ:    .ASCII  /%D4/
004603  045  104  064                .ASCII  /%D4/
004606  045  104  064                .ASCII  /%D4/
004611  045  104  064                .ASCII  /%D4/
004614  045  104  064                .ASCII  /%D4/
004617  045  104  064                .ASCII  /%D4/
004622  045  104  064                .ASCII  /%D4/
004625  045  104  064                .ASCII  /%D4/
004630  000  000                    .ASCII  <00><00>
004632  045  116  045      P.ACR:    .ASCII  /%N%/
004635  101  040  056                .ASCII  /A ./
004640  040  040  040                .ASCII  /   /
004643  056  040  040                .ASCII  /.  /
004646  103  116  124                .ASCII  /CNT/
004651  122  040  040                .ASCII  /R  /
004654  040  040  040                .ASCII  /   /
004657  040  056  040                .ASCII  / . /
004662  040  056  056                .ASCII  / ../
004665  056  056  056                .ASCII  /.../
```

```
004670  056  056  056            .ASCII  /.../
004673  056  040  040            .ASCII  /.  /
004676  040  040  040            .ASCII  /   /
004701  040  056  040            .ASCII  / . /
004704  040  056  056            .ASCII  / ../
004707  056  056  056            .ASCII  /.../
004712  056  056  056            .ASCII  /.../
004715  056  000  000            .ASCII  /./<00><00>
004720  045  101  040  P.ACS:    .ASCII  /%A /
004723  040  040  056            .ASCII  /  ./
004726  040  040  040            .ASCII  /   /
004731  056  045  104            .ASCII  /.%D/
004734  064  045  101            .ASCII  /4%A/
004737  040  040  040            .ASCII  /   /
004742  056  040  040            .ASCII  /.  /
004745  040  056  040            .ASCII  / . /
004750  040  040  056            .ASCII  /  ./
004753  045  104  064            .ASCII  /%D4/
004756  045  101  040            .ASCII  /%A /
004761  040  040  056            .ASCII  /  ./
004764  000  000                 .ASCII  <00><00>
004766  045  116  045  P.ACT:    .ASCII  /%N%/
004771  116  045  101            .ASCII  /N%A/
004774  125  116  111            .ASCII  /UNI/
004777  124  040  040            .ASCII  /T  /
005002  104  111  123            .ASCII  /DIS/
005005  113  040  040            .ASCII  /K  /
005010  040  040  040            .ASCII  /   /
005013  040  040  040            .ASCII  /   /
005016  040  040  040            .ASCII  /   /
005021  040  043  040            .ASCII  / # /
005024  117  106  040            .ASCII  /OF /
005027  040  040  043            .ASCII  /  #/
005032  040  102  114            .ASCII  / BL/
005035  113  123  040            .ASCII  /KS /
005040  040  040  040            .ASCII  /   /
005043  040  040  040            .ASCII  /   /
005046  043  040  117            .ASCII  /# O/
005051  106  040  040            .ASCII  /F  /
005054  040  040  043            .ASCII  /  #/
005057  040  102  114            .ASCII  / BL/
005062  113  123  040            .ASCII  /KS /
005065  000                      .ASCII  <00>
005066  045  116  045  P.ACU:    .ASCII  /%N%/
005071  101  040  040            .ASCII  /A  /
005074  043  040  040            .ASCII  /#  /
005077  040  040  040            .ASCII  /   /
005102  043  040  040            .ASCII  /#  /
005105  040  040  040            .ASCII  /   /
005110  124  131  120            .ASCII  /TYP/
005113  105  040  040            .ASCII  /E  /
005116  040  122  105            .ASCII  / RE/
005121  101  104  123            .ASCII  /ADS/
```

```
005124  040     040     040             .ASCII  /   /
005127  040     040     122             .ASCII  /  R/
005132  105     101     104             .ASCII  /EAD/
005135  040     040     040             .ASCII  /   /
005140  040     040     040             .ASCII  /   /
005143  127     122     111             .ASCII  /WRI/
005146  124     105     123             .ASCII  /TES/
005151  040     040     127             .ASCII  /  W/
005154  122     111     124             .ASCII  /RIT/
005157  124     105     116             .ASCII  /TEN/
005162  040     000                     .ASCII  / /<00>
005164  045     116     045     P.ACV:  .ASCII  /%N%/
005167  101     055     055             .ASCII  /A--/
005172  055     055     040             .ASCII  /-- /
005175  040     055     055             .ASCII  / --/
005200  055     055     040             .ASCII  /-- /
005203  040     055     055             .ASCII  / --/
005206  055     055     055             .ASCII  /---/
005211  055     055     040             .ASCII  /-- /
005214  040     055     055             .ASCII  / --/
005217  055     055     055             .ASCII  /---/
005222  055     040     040             .ASCII  /-  /
005225  040     055     055             .ASCII  / --/
005230  055     055     055             .ASCII  /---/
005233  055     040     040             .ASCII  /-  /
005236  040     040     040             .ASCII  /   /
005241  055     055     055             .ASCII  /---/
005244  055     055     055             .ASCII  /---/
005247  040     040     040             .ASCII  /   /
005252  055     055     055             .ASCII  /---/
005255  055     055     055             .ASCII  /---/
005260  040     040     000             .ASCII  /  /<00>
005263  000                             .ASCII  <00>
005264  045     116     045     P.ACW:  .ASCII  /%N%/
005267  123     061     045             .ASCII  /S1%/
005272  104     062     045             .ASCII  /D2%/
005275  123     064     045             .ASCII  /S4%/
005300  104     062     045             .ASCII  /D2%/
005303  101     040     040             .ASCII  /A  /
005306  040     104     102             .ASCII  / DB/
005311  116     122     104             .ASCII  /NRD/
005314  065     061     040             .ASCII  /51 /
005317  040     045     104             .ASCII  / %D/
005322  066     045     123             .ASCII  /6%S/
005325  063     045     104             .ASCII  /3%D/
005330  066     045     123             .ASCII  /6%S/
005333  065     045     104             .ASCII  /5%D/
005336  066     045     123             .ASCII  /6%S/
005341  063     045     104             .ASCII  /3%D/
005344  066     000                     .ASCII  /6/<00>
005346  124     117     117     P.ACX:  .ASCII  /TOO/
005351  040     115     101             .ASCII  / MA/
005354  116     131     040             .ASCII  /NY /
```

```
005357  125  116  111                .ASCII  /UNI/
005362  124  123  000                .ASCII  /TS/<00>
005365  000                         .ASCII  <00>
005366  116  117  124     P.ACY:     .ASCII  /NOT/
005371  040  105  116                .ASCII  / EN/
005374  117  125  107                .ASCII  /OUG/
005377  110  040  106                .ASCII  /H F/
005402  122  105  105                .ASCII  /REE/
005405  040  115  105                .ASCII  / ME/
005410  115  117  122                .ASCII  /MOR/
005413  131  040  106                .ASCII  /Y F/
005416  117  122  040                .ASCII  /OR /
005421  101  114  114                .ASCII  /ALL/
005424  117  103  101                .ASCII  /OCA/
005427  124  111  116                .ASCII  /TIN/
005432  107  040  122                .ASCII  /G R/
005435  105  101  104                .ASCII  /EAD/
005440  057  127  122                .ASCII  <57>/WR/
005443  111  124  105                .ASCII  /ITE/
005446  040  102  125                .ASCII  / BU/
005451  106  106  105                .ASCII  /FFE/
005454  122  123  000                .ASCII  /RS/<00>
005457  000                         .ASCII  <00>
005460  122  105  107     P.ACZ:     .ASCII  /REG/
005463  111  123  124                .ASCII  /IST/
005466  105  122  040                .ASCII  /ER /
005471  105  130  111                .ASCII  /EXI/
005474  123  124  105                .ASCII  /STE/
005477  116  103  105                .ASCII  /NCE/
005502  040  124  105                .ASCII  / TE/
005505  123  124  040                .ASCII  /ST /
005510  106  101  111                .ASCII  /FAI/
005513  114  105  104                .ASCII  /LED/
005516  000  000                    .ASCII  <00><00>
005520  126  105  103     P.ADA:     .ASCII  /VEC/
005523  124  117  122                .ASCII  /TOR/
005526  040  124  105                .ASCII  / TE/
005531  123  124  040                .ASCII  /ST /
005534  106  101  111                .ASCII  /FAI/
005537  114  105  104                .ASCII  /LED/
005542  000  000                    .ASCII  <00><00>
005544  102  122  040     P.ADB:     .ASCII  /BR /
005547  114  105  126                .ASCII  /LEV/
005552  105  114  040                .ASCII  /EL /
005555  124  105  123                .ASCII  /TES/
005560  124  040  106                .ASCII  /T F/
005563  101  111  114                .ASCII  /AIL/
005566  105  104  000                .ASCII  /ED/<00>
005571  000                         .ASCII  <00>
005572  111  116  111     P.ADC:     .ASCII  /INI/
005575  124  040  123                .ASCII  /T S/
005600  105  121  125                .ASCII  /EQU/
005603  105  116  103                .ASCII  /ENC/
```

```
005606  105  040  106          .ASCII  /E F/
005611  101  111  114          .ASCII  /AIL/
005614  105  104  000          .ASCII  /ED/<00>
005617  000                    .ASCII  <00>
005620  106  101  124  P.ADD:   .ASCII  /FAT/
005623  101  114  040          .ASCII  /AL /
005626  103  117  116          .ASCII  /CON/
005631  124  122  117          .ASCII  /TRO/
005634  114  114  105          .ASCII  /LLE/
005637  122  040  105          .ASCII  /R E/
005642  122  122  117          .ASCII  /RRO/
005645  122  000  000          .ASCII  /R/<00><00>
005650  117  116  114  P.ADE:   .ASCII  /ONL/
005653  111  116  105          .ASCII  /INE/
005656  040  106  101          .ASCII  / FA/
005661  111  114  105          .ASCII  /ILE/
005664  104  000               .ASCII  /D/<00>
005666  127  122  111  P.ADF:   .ASCII  /WRI/
005671  124  105  055          .ASCII  /TE-/
005674  120  122  117          .ASCII  /PRO/
005677  124  105  103          .ASCII  /TEC/
005702  124  040  103          .ASCII  /T C/
005705  117  116  106          .ASCII  /ONF/
005710  114  111  103          .ASCII  /LIC/
005713  124  000  000          .ASCII  /T/<00><00>
005716  101  103  103  P.ADG:   .ASCII  /ACC/
005721  105  123  123          .ASCII  /ESS/
005724  040  106  101          .ASCII  / FA/
005727  111  114  105          .ASCII  /ILE/
005732  104  000               .ASCII  /D/<00>
005734  106  101  124  P.ADH:   .ASCII  /FAT/
005737  101  114  040          .ASCII  /AL /
005742  111  057  117          .ASCII  /I/<57>/O/
005745  040  105  122          .ASCII  / ER/
005750  122  117  122          .ASCII  /ROR/
005753  000                    .ASCII  <00>
005754  106  101  111  P.ADI:   .ASCII  /FAI/
005757  114  105  104          .ASCII  /LED/
005762  040  124  117          .ASCII  / TO/
005765  040  123  105          .ASCII  / SE/
005770  116  104  040          .ASCII  /ND /
005773  123  105  124          .ASCII  /SET/
005776  055  103  117          .ASCII  /-CO/
006001  116  124  122          .ASCII  /NTR/
006004  117  114  114          .ASCII  /OLL/
006007  105  122  055          .ASCII  /ER-/
006012  103  110  101          .ASCII  /CHA/
006015  122  101  103          .ASCII  /RAC/
006020  124  105  122          .ASCII  /TER/
006023  111  123  124          .ASCII  /IST/
006026  111  103  123          .ASCII  /ICS/
006031  040  103  117          .ASCII  / CO/
006034  115  115  101          .ASCII  /MMA/
```

```
006037  116  104  000          .ASCII  /ND/<00>
006042  123  105  124  P.ADJ:  .ASCII  /SET/
006045  055  103  117          .ASCII  /-CO/
006050  116  124  122          .ASCII  /NTR/
006053  117  114  114          .ASCII  /OLL/
006056  105  122  055          .ASCII  /ER-/
006061  103  110  101          .ASCII  /CHA/
006064  122  101  103          .ASCII  /RAC/
006067  124  105  122          .ASCII  /TER/
006072  111  123  124          .ASCII  /IST/
006075  111  103  123          .ASCII  /ICS/
006100  040  122  105          .ASCII  / RE/
006103  123  120  117          .ASCII  /SPO/
006106  116  123  105          .ASCII  /NSE/
006111  040  110  101          .ASCII  / HA/
006114  123  040  102          .ASCII  /S B/
006117  101  104  040          .ASCII  /AD /
006122  105  116  104          .ASCII  /END/
006125  103  117  104          .ASCII  /COD/
006130  105  040  117          .ASCII  /E O/
006133  122  040  106          .ASCII  /R F/
006136  114  101  107          .ASCII  /LAG/
006141  123  040  111          .ASCII  /S I/
006144  116  040  105          .ASCII  /N E/
006147  122  122  117          .ASCII  /RRO/
006152  122  000               .ASCII  /R/<00>
006154  106  101  111  P.ADK:  .ASCII  /FAI/
006157  114  105  104          .ASCII  /LED/
006162  040  124  117          .ASCII  / TO/
006165  040  123  105          .ASCII  / SE/
006170  116  104  040          .ASCII  /ND /
006173  117  116  055          .ASCII  /ON-/
006176  114  111  116          .ASCII  /LIN/
006201  105  040  103          .ASCII  /E C/
006204  117  115  115          .ASCII  /OMM/
006207  101  116  104          .ASCII  /AND/
006212  000  000               .ASCII  <00><00>
006214  117  116  055  P.ADL:  .ASCII  /ON-/
006217  114  111  116          .ASCII  /LIN/
006222  105  040  122          .ASCII  /E R/
006225  105  123  120          .ASCII  /ESP/
006230  117  116  123          .ASCII  /ONS/
006233  105  040  110          .ASCII  /E H/
006236  101  123  040          .ASCII  /AS /
006241  102  101  104          .ASCII  /BAD/
006244  040  105  116          .ASCII  / EN/
006247  104  103  117          .ASCII  /DCO/
006252  104  105  000          .ASCII  /DE/<00>
006255  000               .ASCII  <00>
006256  111  057  117  P.ADM:  .ASCII  /I/<57>/O/
006261  040  122  105          .ASCII  / RE/
006264  121  125  105          .ASCII  /QUE/
006267  123  124  040          .ASCII  /ST /
```

```
006272  106  101  111                .ASCII  /FAI/
006275  114  105  104                .ASCII  /LED/
006300  000  000                     .ASCII  <00><00>
006302  045  101  115          P.ADN: .ASCII  /%AM/
006305  117  122  105                .ASCII  /ORE/
006310  040  124  110                .ASCII  / TH/
006313  101  116  040                .ASCII  /AN /
006316  045  104  062                .ASCII  /%D2/
006321  045  101  040                .ASCII  /%A /
006324  125  116  111                .ASCII  /UNI/
006327  124  123  040                .ASCII  /TS /
006332  123  120  105                .ASCII  /SPE/
006335  103  111  106                .ASCII  /CIF/
006340  111  105  104                .ASCII  /IED/
006343  000                         .ASCII  <00>
006344  045  101  052          P.ADO: .ASCII  /%A*/
006347  040  116  117                .ASCII  / NO/
006352  040  122  105                .ASCII  / RE/
006355  123  120  117                .ASCII  /SPO/
006360  116  123  105                .ASCII  /NSE/
006363  040  101  124                .ASCII  / AT/
006366  040  101  104                .ASCII  / AD/
006371  104  122  105                .ASCII  /DRE/
006374  123  123  040                .ASCII  /SS /
006377  045  117  066                .ASCII  /%06/
006402  000  000                     .ASCII  <00><00>
006404  045  101  052          P.ADP: .ASCII  /%A*/
006407  040  111  116                .ASCII  / IN/
006412  103  117  122                .ASCII  /COR/
006415  122  105  103                .ASCII  /REC/
006420  124  040  102                .ASCII  /T B/
006423  122  040  114                .ASCII  /R L/
006426  105  126  105                .ASCII  /EVE/
006431  114  040  106                .ASCII  /L F/
006434  117  122  040                .ASCII  /OR /
006437  104  105  126                .ASCII  /DEV/
006442  111  103  105                .ASCII  /ICE/
006445  040  045  117                .ASCII  / %O/
006450  066  000                     .ASCII  /6/<00>
006452  045  101  052          P.ADQ: .ASCII  /%A*/
006455  040  123  124                .ASCII  / ST/
006460  105  120  040                .ASCII  /EP /
006463  045  104  061                .ASCII  /%D1/
006466  045  101  040                .ASCII  /%A /
006471  122  105  101                .ASCII  /REA/
006474  104  040  105                .ASCII  /D E/
006477  122  122  117                .ASCII  /RRO/
006502  122  000                     .ASCII  /R/<00>
006504  045  101  052          P.ADR: .ASCII  /%A*/
006507  040  102  101                .ASCII  / BA/
006512  104  040  123                .ASCII  /D S/
006515  101  040  103                .ASCII  /A C/
006520  117  104  105                .ASCII  /ODE/
```

```
006523  040  106  122                .ASCII  / FR/
006526  117  115  040                .ASCII  /OM /
006531  104  105  126                .ASCII  /DEV/
006534  111  103  105                .ASCII  /ICE/
006537  040  045  117                .ASCII  / %O/
006542  066  000                    .ASCII  /6/<00>
006544  045  101  052          P.ADS:.ASCII  /%A*/
006547  040  104  111                .ASCII  / DI/
006552  123  113  045                .ASCII  /SK%/
006555  104  062  045                .ASCII  /D2%/
006560  101  040  127                .ASCII  /A W/
006563  105  116  124                .ASCII  /ENT/
006566  040  117  106                .ASCII  / OF/
006571  106  114  111                .ASCII  /FLI/
006574  116  105  000                .ASCII  /NE/<00>
006577  000                         .ASCII  <00>
006600  045  101  052          P.ADT:.ASCII  /%A*/
006603  040  104  105                .ASCII  / DE/
006606  126  111  103                .ASCII  /VIC/
006611  105  040  045                .ASCII  /E %/
006614  117  066  045                .ASCII  /O6%/
006617  101  040  116                .ASCII  /A N/
006622  117  124  040                .ASCII  /OT /
006625  120  122  117                .ASCII  /PRO/
006630  103  105  123                .ASCII  /CES/
006633  123  111  116                .ASCII  /SIN/
006636  107  040  103                .ASCII  /G C/
006641  117  115  115                .ASCII  /OMM/
006644  101  116  104                .ASCII  /AND/
006647  040  120  101                .ASCII  / PA/
006652  103  113  105                .ASCII  /CKE/
006655  124  123  000                .ASCII  /TS/<00>
006660  040  055  040          P.ADU:.ASCII  / - /
006663  165  156  162                .ASCII  /unr/
006666  145  143  157                .ASCII  /eco/
006671  147  156  151                .ASCII  /gni/
006674  172  145  144                .ASCII  /zed/
006677  040  115  105                .ASCII  / ME/
006702  123  123  101                .ASCII  /SSA/
006705  107  105  040                .ASCII  /GE /
006710  124  131  120                .ASCII  /TYP/
006713  105  000  000                .ASCII  /E/<00><00>
006716  040  055  040          P.ADV:.ASCII  / - /
006721  165  156  162                .ASCII  /unr/
006724  145  143  157                .ASCII  /eco/
006727  147  156  151                .ASCII  /gni/
006732  172  145  144                .ASCII  /zed/
006735  040  143  157                .ASCII  / co/
006740  156  156  145                .ASCII  /nne/
006743  143  164  151                .ASCII  /cti/
006746  157  156  040                .ASCII  /on /
006751  151  144  000                .ASCII  /id/<00>
006754  040  055  040          P.ADW:.ASCII  / - /
```

```
006757  165  156  162            .ASCII  /unr/
006762  145  143  157            .ASCII  /eco/
006765  147  156  151            .ASCII  /gni/
006770  172  145  144            .ASCII  /zed/
006773  040  122  105            .ASCII  / RE/
006776  124  125  122            .ASCII  /TUR/
007001  116  040  155            .ASCII  /N m/
007004  145  163  163            .ASCII  /ess/
007007  141  147  145            .ASCII  /age/
007012  000  000                 .ASCII  <00><00>
007014  040  055  040  P.ADX:     .ASCII  / - /
007017  165  156  162            .ASCII  /unr/
007022  145  143  157            .ASCII  /eco/
007025  147  156  151            .ASCII  /gni/
007030  172  145  144            .ASCII  /zed/
007033  040  122  105            .ASCII  / RE/
007036  124  125  122            .ASCII  /TUR/
007041  116  040  120            .ASCII  /N P/
007044  101  103  113            .ASCII  /ACK/
007047  105  124  000            .ASCII  /ET/<00>
007052  040  055  040  P.ADY:     .ASCII  / - /
007055  165  156  162            .ASCII  /unr/
007060  145  143  157            .ASCII  /eco/
007063  147  156  151            .ASCII  /gni/
007066  172  145  144            .ASCII  /zed/
007071  040  103  122            .ASCII  / CR/
007074  116  000                 .ASCII  /N/<00>
007076  040  055  040  P.ADZ:     .ASCII  / - /
007101  125  116  122            .ASCII  /UNR/
007104  105  103  117            .ASCII  /ECO/
007107  107  116  111            .ASCII  /GNI/
007112  132  105  104            .ASCII  /ZED/
007115  040  117  120            .ASCII  / OP/
007120  103  117  104            .ASCII  /COD/
007123  105  000  000            .ASCII  /E/<00><00>
007126  040  055  040  P.AEA:     .ASCII  / - /
007131  115  123  103            .ASCII  /MSC/
007134  120  040  123            .ASCII  /P S/
007137  124  101  124            .ASCII  /TAT/
007142  125  123  040            .ASCII  /US /
007145  103  117  104            .ASCII  /COD/
007150  105  040  105            .ASCII  /E E/
007153  122  122  117            .ASCII  /RRO/
007156  122  000                 .ASCII  /R/<00>
007160  040  055  040  P.AEB:     .ASCII  / - /
007163  104  125  120            .ASCII  /DUP/
007166  040  123  124            .ASCII  / ST/
007171  101  124  125            .ASCII  /ATU/
007174  123  040  103            .ASCII  /S C/
007177  117  104  105            .ASCII  /ODE/
007202  040  105  122            .ASCII  / ER/
007205  122  117  122            .ASCII  /ROR/
007210  000  000                 .ASCII  <00><00>
```

```
007212  040     055     040     P.AEC:  .ASCII  / - /
007215  165     156     162             .ASCII  /unr/
007220  145     143     157             .ASCII  /eco/
007223  147     156     151             .ASCII  /gni/
007226  172     145     144             .ASCII  /zed/
007231  040     123     124             .ASCII  / ST/
007234  101     124     125             .ASCII  /ATU/
007237  123     040     103             .ASCII  /S C/
007242  117     104     105             .ASCII  /ODE/
007245  000                             .ASCII  <00>
007246  040     055     040     P.AED:  .ASCII  / - /
007251  114     102     116             .ASCII  /LBN/
007254  040     110     117             .ASCII  / HO/
007257  123     124     040             .ASCII  /ST /
007262  103     117     115             .ASCII  /COM/
007265  120     101     122             .ASCII  /PAR/
007270  105     040     105             .ASCII  /E E/
007273  122     122     117             .ASCII  /RRO/
007276  122     000                     .ASCII  /R/<00>
007300  040     055     040     P.AEE:  .ASCII  / - /
007303  104     102     116             .ASCII  /DBN/
007306  040     110     117             .ASCII  / HO/
007311  123     124     040             .ASCII  /ST /
007314  103     117     115             .ASCII  /COM/
007317  120     101     122             .ASCII  /PAR/
007322  105     040     105             .ASCII  /E E/
007325  122     122     117             .ASCII  /RRO/
007330  122     000                     .ASCII  /R/<00>
007332  040     055     040     P.AEF:  .ASCII  / - /
007335  125     116     101             .ASCII  /UNA/
007340  102     114     105             .ASCII  /BLE/
007343  040     124     117             .ASCII  / TO/
007346  040     114     117             .ASCII  / LO/
007351  101     104     040             .ASCII  /AD /
007354  104     125     120             .ASCII  /DUP/
007357  040     115     105             .ASCII  / ME/
007362  104     111     101             .ASCII  /DIA/
007365  040     000     000             .ASCII  / /<00><00>
007370  040     055     040     P.AEG:  .ASCII  / - /
007373  105     122     122             .ASCII  /ERR/
007376  117     122     040             .ASCII  /OR /
007401  111     116     040             .ASCII  /IN /
007404  104     125     120             .ASCII  /DUP/
007407  055     120     113             .ASCII  /-PK/
007412  124     040     127             .ASCII  /T W/
007415  110     105     116             .ASCII  /HEN/
007420  040     125     123             .ASCII  / US/
007423  111     116     107             .ASCII  /ING/
007426  040     103     124             .ASCII  / CT/
007431  114     122     040             .ASCII  /LR /
007434  114     103     040             .ASCII  /LC /
007437  120     122     107             .ASCII  /PRG/
007442  000     000                     .ASCII  <00><00>
```

```
007444  045  101  111        P.AEI:  .ASCII  /%AI/
007447  116  126  101                .ASCII  /NVA/
007452  114  111  104                .ASCII  /LID/
007455  040  103  117                .ASCII  / CO/
007460  115  115  101                .ASCII  /MMA/
007463  116  104  000                .ASCII  /ND/<00>
007466  045  101  103        P.AEJ:  .ASCII  /%AC/
007471  117  115  115                .ASCII  /OMM/
007474  101  116  104                .ASCII  /AND/
007477  040  101  102                .ASCII  / AB/
007502  117  122  124                .ASCII  /ORT/
007505  105  104  000                .ASCII  /ED/<00>
007510  045  101  125        P.AEK:  .ASCII  /%AU/
007513  116  111  124                .ASCII  /NIT/
007516  040  117  106                .ASCII  / OF/
007521  106  114  111                .ASCII  /FLI/
007524  116  105  000                .ASCII  /NE/<00>
007527  000                        .ASCII  <00>
007530  045  101  124        P.AEL:  .ASCII  /%AT/
007533  122  101  116                .ASCII  /RAN/
007536  123  111  124                .ASCII  /SIT/
007541  111  117  116                .ASCII  /ION/
007544  040  124  117                .ASCII  / TO/
007547  040  101  126                .ASCII  / AV/
007552  101  111  114                .ASCII  /AIL/
007555  101  102  114                .ASCII  /ABL/
007560  105  040  123                .ASCII  /E S/
007563  124  101  124                .ASCII  /TAT/
007566  105  000                    .ASCII  /E/<00>
007570  045  101  115        P.AEM:  .ASCII  /%AM/
007573  105  104  111                .ASCII  /EDI/
007576  101  040  106                .ASCII  /A F/
007601  117  122  115                .ASCII  /ORM/
007604  101  124  040                .ASCII  /AT /
007607  105  122  122                .ASCII  /ERR/
007612  117  122  000                .ASCII  /OR/<00>
007615  000                        .ASCII  <00>
007616  045  101  127        P.AEN:  .ASCII  /%AW/
007621  122  111  124                .ASCII  /RIT/
007624  105  055  120                .ASCII  /E-P/
007627  122  117  124                .ASCII  /ROT/
007632  105  103  124                .ASCII  /ECT/
007635  105  104  000                .ASCII  /ED/<00>
007640  045  101  104        P.AEO:  .ASCII  /%AD/
007643  105  126  111                .ASCII  /EVI/
007646  103  105  040                .ASCII  /CE /
007651  103  117  115                .ASCII  /COM/
007654  120  101  122                .ASCII  /PAR/
007657  105  040  105                .ASCII  /E E/
007662  122  122  117                .ASCII  /RRO/
007665  122  000  000                .ASCII  /R/<00><00>
007670  045  101  104        P.AEP:  .ASCII  /%AD/
007673  101  124  101                .ASCII  /ATA/
```

```
007676  040  105  122            .ASCII  / ER/
007701  122  117  122            .ASCII  /ROR/
007704  000  000                 .ASCII  <00><00>
007706  045  101  110  P.AEQ:     .ASCII  /%AH/
007711  117  123  124            .ASCII  /OST/
007714  040  102  125            .ASCII  / BU/
007717  106  106  105            .ASCII  /FFE/
007722  122  040  101            .ASCII  /R A/
007725  103  103  105            .ASCII  /CCE/
007730  123  123  040            .ASCII  /SS /
007733  105  122  122            .ASCII  /ERR/
007736  117  122  000            .ASCII  /OR/<00>
007741  000                     .ASCII  <00>
007742  045  101  103  P.AER:     .ASCII  /%AC/
007745  117  116  124            .ASCII  /ONT/
007750  122  117  114            .ASCII  /ROL/
007753  114  105  122            .ASCII  /LER/
007756  040  105  122            .ASCII  / ER/
007761  122  117  122            .ASCII  /ROR/
007764  000  000                 .ASCII  <00><00>
007766  045  101  104  P.AES:     .ASCII  /%AD/
007771  122  111  126            .ASCII  /RIV/
007774  105  040  105            .ASCII  /E E/
007777  122  122  117            .ASCII  /RRO/
010002  122  000                 .ASCII  /R/<00>
010004  045  101  115  P.AET:     .ASCII  /%AM/
010007  105  123  123            .ASCII  /ESS/
010012  101  107  105            .ASCII  /AGE/
010015  040  106  122            .ASCII  / FR/
010020  117  115  040            .ASCII  /OM /
010023  111  116  124            .ASCII  /INT/
010026  105  122  116            .ASCII  /ERN/
010031  101  114  040            .ASCII  /AL /
010034  104  111  101            .ASCII  /DIA/
010037  107  116  117            .ASCII  /GNO/
010042  123  124  111            .ASCII  /STI/
010045  103  123  000            .ASCII  /CS/<00>
010050  045  101  110  P.AEU:     .ASCII  /%AH/
010053  117  123  124            .ASCII  /OST/
010056  040  103  117            .ASCII  / CO/
010061  115  120  101            .ASCII  /MPA/
010064  122  105  040            .ASCII  /RE /
010067  105  122  122            .ASCII  /ERR/
010072  117  122  000            .ASCII  /OR/<00>
010075  000                     .ASCII  <00>
010076  045  101  103  P.AEV:     .ASCII  /%AC/
010101  117  115  115            .ASCII  /OMM/
010104  101  116  104            .ASCII  /AND/
010107  040  124  111            .ASCII  / TI/
010112  115  105  117            .ASCII  /MEO/
010115  125  124  000            .ASCII  /UT/<00>
010120  007444'          P.AEH:     .WORD   P.AEI
010122  007466'                     .WORD   P.AEJ
```

```
010124  007510'                                 .WORD   P.AEK
010126  007530'                                 .WORD   P.AEL
010130  007570'                                 .WORD   P.AEM
010132  007616'                                 .WORD   P.AEN
010134  007640'                                 .WORD   P.AEO
010136  007670'                                 .WORD   P.AEP
010140  007706'                                 .WORD   P.AEQ
010142  007742'                                 .WORD   P.AER
010144  007766'                                 .WORD   P.AES
010146  010004'                                 .WORD   P.AET
010150  010050'                                 .WORD   P.AEU
010152  010076'                                 .WORD   P.AEV
010154      045     116     045         P.AEW:  .ASCII  /%N%/
010157      101     105     122                 .ASCII  /AER/
010162      122     117     122                 .ASCII  /ROR/
010165      040     114     117                 .ASCII  / LO/
010170      107     040     115                 .ASCII  /G M/
010173      105     123     123                 .ASCII  /ESS/
010176      101     107     105                 .ASCII  /AGE/
010201      040     122     105                 .ASCII  / RE/
010204      103     105     111                 .ASCII  /CEI/
010207      126     105     104                 .ASCII  /VED/
010212      072     045     116                 .ASCII  /:%N/
010215      000                                 .ASCII  <00>
010216      045     101     052         P.AEY:  .ASCII  /%A*/
010221      040     103     117                 .ASCII  / CO/
010224      116     124     122                 .ASCII  /NTR/
010227      117     114     114                 .ASCII  /OLL/
010232      105     122     040                 .ASCII  /ER /
010235      105     122     122                 .ASCII  /ERR/
010240      117     122     045                 .ASCII  /OR%/
010243      116     000     000                 .ASCII  /N/<00><00>
010246      045     101     052         P.AEZ:  .ASCII  /%A*/
010251      040     110     117                 .ASCII  / HO/
010254      123     124     040                 .ASCII  /ST /
010257      115     105     115                 .ASCII  /MEM/
010262      117     122     131                 .ASCII  /ORY/
010265      040     101     103                 .ASCII  / AC/
010270      103     105     123                 .ASCII  /CES/
010273      123     040     105                 .ASCII  /S E/
010276      122     122     117                 .ASCII  /RRO/
010301      122     045     116                 .ASCII  /R%N/
010304      000     000                         .ASCII  <00><00>
010306      045     101     052         P.AFA:  .ASCII  /%A*/
010311      040     104     111                 .ASCII  / DI/
010314      123     113     045                 .ASCII  /SK%/
010317      104     062     045                 .ASCII  /D2%/
010322      101     040     055                 .ASCII  /A -/
010325      040     104     111                 .ASCII  / DI/
010330      123     113     040                 .ASCII  /SK /
010333      124     122     101                 .ASCII  /TRA/
010336      116     123     106                 .ASCII  /NSF/
010341      105     122     040                 .ASCII  /ER /
```

```
010344  105     122     122             .ASCII  /ERR/
010347  117     122     045             .ASCII  /OR%/
010352  116     000                     .ASCII  /N/<00>
010354  045     101     052     P.AFB:  .ASCII  /%A*/
010357  040     104     111             .ASCII  / DI/
010362  123     113     045             .ASCII  /SK%/
010365  104     062     045             .ASCII  /D2%/
010370  101     040     055             .ASCII  /A -/
010373  040     042     123             .ASCII  / "S/
010376  124     101     116             .ASCII  /TAN/
010401  104     101     122             .ASCII  /DAR/
010404  104     040     104             .ASCII  /D D/
010407  111     123     113             .ASCII  /ISK/
010412  040     111     116             .ASCII  / IN/
010415  124     105     122             .ASCII  /TER/
010420  103     117     116             .ASCII  /CON/
010423  116     105     103             .ASCII  /NEC/
010426  124     042     040             .ASCII  /T" /
010431  105     122     122             .ASCII  /ERR/
010434  117     122     045             .ASCII  /OR%/
010437  116     000     000             .ASCII  /N/<00><00>
010442  045     101     052     P.AFC:  .ASCII  /%A*/
010445  040     104     111             .ASCII  / DI/
010450  123     113     045             .ASCII  /SK%/
010453  104     062     045             .ASCII  /D2%/
010456  101     040     055             .ASCII  /A -/
010461  040     042     123             .ASCII  / "S/
010464  115     101     114             .ASCII  /MAL/
010467  114     040     104             .ASCII  /L D/
010472  111     123     113             .ASCII  /ISK/
010475  042     040     105             .ASCII  /" E/
010500  122     122     117             .ASCII  /RRO/
010503  122     045     116             .ASCII  /R%N/
010506  000     000                     .ASCII  <00><00>
010510  010216' 000000  000000  P.AEX:  .WORD   P.AEY
010512  010246' 000000  000000          .WORD   P.AEZ
010514  010306' 000000  000000          .WORD   P.AFA
010516  010354' 000000  000000          .WORD   P.AFB
010520  010442' 000000  000000          .WORD   P.AFC
010522  045     116     045     P.AFD:  .ASCII  /%N%/
010525  101     111     057             .ASCII  /AI/<57>
010530  117     040     102             .ASCII  /O B/
010533  125     106     106             .ASCII  /UFF/
010536  105     122     040             .ASCII  /ER /
010541  101     104     104             .ASCII  /ADD/
010544  122     105     123             .ASCII  /RES/
010547  123     040     106             .ASCII  /S F/
010552  117     122     040             .ASCII  /OR /
010555  122     105     101             .ASCII  /REA/
010560  104     040     050             .ASCII  /D (/
010563  063     062     040             .ASCII  /32 /
010566  102     111     124             .ASCII  /BIT/
010571  123     051     072             .ASCII  /S):/
```

```
010574  040  045  117                .ASCII  / %O/
010577  066  045  101                .ASCII  /6%A/
010602  040  045  117                .ASCII  / %O/
010605  066  045  116                .ASCII  /6%N/
010610  000  000                     .ASCII  <00><00>
010612  045  116  045       P.AFE:   .ASCII  /%N%/
010615  101  111  057                .ASCII  /AI/<57>
010620  117  040  102                .ASCII  /O B/
010623  125  106  106                .ASCII  /UFF/
010626  105  122  040                .ASCII  /ER /
010631  101  104  104                .ASCII  /ADD/
010634  122  105  123                .ASCII  /RES/
010637  123  040  106                .ASCII  /S F/
010642  117  122  040                .ASCII  /OR /
010645  127  122  111                .ASCII  /WRI/
010650  124  105  040                .ASCII  /TE /
010653  050  063  062                .ASCII  /(32/
010656  040  102  111                .ASCII  / BI/
010661  124  123  051                .ASCII  /TS)/
010664  072  040  045                .ASCII  /: %/
010667  117  066  045                .ASCII  /06%/
010672  101  040  045                .ASCII  /A %/
010675  117  066  000                .ASCII  /06/<00>
010700  045  116  045       P.AFF:   .ASCII  /%N%/
010703  101  114  102                .ASCII  /ALB/
010706  116  072  040                .ASCII  /N: /
010711  050  122  105                .ASCII  /(RE/
010714  101  104  051                .ASCII  /AD)/
010717  040  045  104                .ASCII  / %D/
010722  065  045  101                .ASCII  /5%A/
010725  056  040  050                .ASCII  /. (/
010730  117  103  124                .ASCII  /OCT/
010733  040  045  117                .ASCII  / %O/
010736  066  045  101                .ASCII  /6%A/
010741  051  000  000                .ASCII  /)/<00><00>
010744  045  116  045       P.AFG:   .ASCII  /%N%/
010747  101  114  102                .ASCII  /ALB/
010752  116  072  040                .ASCII  /N: /
010755  050  127  122                .ASCII  /(WR/
010760  111  124  105                .ASCII  /ITE/
010763  051  040  045                .ASCII  /) %/
010766  104  065  045                .ASCII  /D5%/
010771  101  056  040                .ASCII  /A. /
010774  050  117  103                .ASCII  /(OC/
010777  124  040  045                .ASCII  /T %/
011002  117  066  045                .ASCII  /06%/
011005  101  051  000                .ASCII  /A)/<00>
011010  045  116  045       P.AFH:   .ASCII  /%N%/
011013  101  122  105                .ASCII  /ARE/
011016  120  114  101                .ASCII  /PLA/
011021  103  105  115                .ASCII  /CEM/
011024  105  116  124                .ASCII  /ENT/
011027  040  102  114                .ASCII  / BL/
```

```
011032  117  103  113                .ASCII  /OCK/
011035  040  116  117                .ASCII  / NO/
011040  056  040  045                .ASCII  /. %/
011043  104  065  045                .ASCII  /D5%/
011046  101  056  040                .ASCII  /A. /
011051  050  117  103                .ASCII  /(OC/
011054  124  040  045                .ASCII  /T %/
011057  117  066  045                .ASCII  /O6%/
011062  101  051  000                .ASCII  /A)/<00>
011065  000                        .ASCII  <00>
011066  045  116  045  P.AFI:       .ASCII  /%N%/
011071  101  102  131                .ASCII  /ABY/
011074  124  105  040                .ASCII  /TE /
011077  103  117  125                .ASCII  /COU/
011102  116  124  040                .ASCII  /NT /
011105  111  116  040                .ASCII  /IN /
011110  122  105  101                .ASCII  /REA/
011113  104  040  103                .ASCII  /D C/
011116  117  115  115                .ASCII  /OMM/
011121  101  116  104                .ASCII  /AND/
011124  072  040  045                .ASCII  /: %/
011127  104  065  045                .ASCII  /D5%/
011132  101  056  000                .ASCII  /A./<00>
011135  000                        .ASCII  <00>
011136  045  116  045  P.AFJ:       .ASCII  /%N%/
011141  101  102  131                .ASCII  /ABY/
011144  124  105  040                .ASCII  /TE /
011147  103  117  125                .ASCII  /COU/
011152  116  124  040                .ASCII  /NT /
011155  111  116  040                .ASCII  /IN /
011160  127  122  111                .ASCII  /WRI/
011163  124  105  040                .ASCII  /TE /
011166  103  117  115                .ASCII  /COM/
011171  115  101  116                .ASCII  /MAN/
011174  104  072  040                .ASCII  /D: /
011177  045  104  065                .ASCII  /%D5/
011202  045  101  056                .ASCII  /%A./
011205  000                        .ASCII  <00>
011206  045  116  045  P.AFK:       .ASCII  /%N%/
011211  101  040  052                .ASCII  /A *
011214  040  104  111                .ASCII  / DI/
011217  123  113  040                .ASCII  /SK /
011222  072  040  045                .ASCII  /: %/
011225  104  062  000                .ASCII  /D2/<00>
011230  045  116  045  P.AFL:       .ASCII  /%N%/
011233  101  123  101                .ASCII  /ASA/
011236  072  040  045                .ASCII  /: %/
011241  117  066  000                .ASCII  /O6/<00>
011244  045  116  045  P.AFM:       .ASCII  /%N%/
011247  101  123  124                .ASCII  /AST/
011252  101  124  125                .ASCII  /ATU/
011255  123  040  103                .ASCII  /S C/
011260  117  104  105                .ASCII  /ODE/
```

```
011263  072     040     000                     .ASCII  /: /<00>
011266  045     116     045     P.AFN:          .ASCII  /%N%/
011271  101     123     124                     .ASCII  /AST/
011274  101     124     125                     .ASCII  /ATU/
011277  123     040     123                     .ASCII  /S S/
011302  125     102     055                     .ASCII  /UB-/
011305  103     117     104                     .ASCII  /COD/
011310  105     072     040                     .ASCII  /E: /
011313  000                                     .ASCII  <00>
011314  045     116     045     P.AFO:          .ASCII  /%N%/
011317  101     103     117                     .ASCII  /ACO/
011322  115     115     101                     .ASCII  /MMA/
011325  116     104     072                     .ASCII  /ND:/
011330  040     000                             .ASCII  / /<00>
011332  045     101     055     P.AFP:          .ASCII  /%A-/
011335  104     125     120                     .ASCII  /DUP/
011340  055     000                             .ASCII  /-/<00>
011342  045     101     055     P.AFQ:          .ASCII  /%A-/
011345  115     123     103                     .ASCII  /MSC/
011350  120     055     000                     .ASCII  /P-/<00>
011353  000                                     .ASCII  <00>
011354  045     101     055     P.AFR:          .ASCII  /%A-/
011357  103     117     115                     .ASCII  /COM/
011362  120     101     122                     .ASCII  /PAR/
011365  105     000     000                     .ASCII  /E/<00><00>
011370  045     116     045     P.AFS:          .ASCII  /%N%/
011373  101     102     101                     .ASCII  /ABA/
011376  104     040     102                     .ASCII  /D B/
011401  114     117     103                     .ASCII  /LOC/
011404  113     040     122                     .ASCII  /K R/
011407  105     120     117                     .ASCII  /EPO/
011412  122     124     105                     .ASCII  /RTE/
011415  104     072     040                     .ASCII  /D: /
011420  045     117     065                     .ASCII  /%O5/
011423  045     101     056                     .ASCII  /%A./
011426  000     000                             .ASCII  <00><00>
011430  045     116     045     P.AFT:          .ASCII  /%N%/
011433  101     114     102                     .ASCII  /ALB/
011436  116     072     040                     .ASCII  /N: /
011441  045     104     065                     .ASCII  /%D5/
011444  045     101     056                     .ASCII  /%A./
011447  040     050     117                     .ASCII  / (O/
011452  103     124     040                     .ASCII  /CT /
011455  045     117     066                     .ASCII  /%O6/
011460  045     101     051                     .ASCII  /%A)/
011463  000                                     .ASCII  <00>
011464  045     116     045     P.AFU:          .ASCII  /%N%/
011467  101     104     102                     .ASCII  /ADB/
011472  116     072     040                     .ASCII  /N: /
011475  045     104     065                     .ASCII  /%D5/
011500  045     101     056                     .ASCII  /%A./
011503  040     050     117                     .ASCII  / (O/
011506  103     124     040                     .ASCII  /CT /
```

```
011511  045  117  066                .ASCII  /%O6/
011514  045  101  051                .ASCII  /%A)/
011517  000                         .ASCII  <00>
011520  045  116  045  P.AFV:       .ASCII  /%N%/
011523  101  102  131                .ASCII  /ABY/
011526  124  105  040                .ASCII  /TE /
011531  103  117  125                .ASCII  /COU/
011534  116  124  040                .ASCII  /NT /
011537  111  116  040                .ASCII  /IN /
011542  103  117  115                .ASCII  /COM/
011545  115  101  116                .ASCII  /MAN/
011550  104  072  040                .ASCII  /D: /
011553  045  104  065                .ASCII  /%D5/
011556  000  000                     .ASCII  <00><00>
011560  045  116  045  P.AFW:       .ASCII  /%N%/
011563  101  101  103                .ASCII  /AAC/
011566  124  125  101                .ASCII  /TUA/
011571  114  040  043                .ASCII  /L #/
011574  040  117  106                .ASCII  / OF/
011577  040  102  131                .ASCII  / BY/
011602  124  105  123                .ASCII  /TES/
011605  040  124  122                .ASCII  / TR/
011610  101  116  123                .ASCII  /ANS/
011613  106  105  122                .ASCII  /FER/
011616  122  105  104                .ASCII  /RED/
011621  072  040  045                .ASCII  /: %/
011624  104  065  000                .ASCII  /D5/<00>
011627  000                         .ASCII  <00>
011630  045  116  045  P.AFX:       .ASCII  /%N%/
011633  101  111  057                .ASCII  /AI/<57>
011636  117  040  102                .ASCII  /O B/
011641  125  106  106                .ASCII  /UFF/
011644  105  122  040                .ASCII  /ER /
011647  101  104  104                .ASCII  /ADD/
011652  122  105  123                .ASCII  /RES/
011655  123  040  050                .ASCII  /S (/
011660  063  062  040                .ASCII  /32 /
011663  102  111  124                .ASCII  /BIT/
011666  123  051  072                .ASCII  /S):/
011671  040  045  117                .ASCII  / %O/
011674  066  045  101                .ASCII  /6%A/
011677  040  045  117                .ASCII  / %O/
011702  066  000                     .ASCII  /6/<00>
011704  045  116  045  P.AFY:       .ASCII  /%N%/
011707  101  103  117                .ASCII  /ACO/
011712  116  124  105                .ASCII  /NTE/
011715  116  124  123                .ASCII  /NTS/
011720  040  117  106                .ASCII  / OF/
011723  040  122  105                .ASCII  / RE/
011726  124  125  122                .ASCII  /TUR/
011731  116  040  120                .ASCII  /N P/
011734  101  103  113                .ASCII  /ACK/
011737  105  124  072                .ASCII  /ET:/
```

```
011742  045  116  000                .ASCII  /%N/<00>
011745  000                         .ASCII  <00>
011746  045  116  045       P.AFZ:  .ASCII  /%N%/
011751  101  115  105               .ASCII  /AME/
011754  123  123  101               .ASCII  /SSA/
011757  107  105  040               .ASCII  /GE /
011762  124  131  120               .ASCII  /TYP/
011765  105  072  040               .ASCII  /E: /
011770  000  000                    .ASCII  <00><00>
011772  045  116  045       P.AGA:  .ASCII  /%N%/
011775  101  115  105               .ASCII  /AME/
012000  123  123  101               .ASCII  /SSA/
012003  107  105  040               .ASCII  /GE /
012006  116  125  115               .ASCII  /NUM/
012011  102  105  122               .ASCII  /BER/
012014  123  072  040               .ASCII  /S: /
012017  000                         .ASCII  <00>
012020  045  116  045       P.AGB:  .ASCII  /%N%/
012023  101  115  105               .ASCII  /AME/
012026  123  123  101               .ASCII  /SSA/
012031  107  105  040               .ASCII  /GE /
012034  105  122  122               .ASCII  /ERR/
012037  117  122  040               .ASCII  /OR /
012042  103  117  104               .ASCII  /COD/
012045  105  123  072               .ASCII  /ES:/
012050  040  000                    .ASCII  / /<00>
012052  045  116  045       P.AGC:  .ASCII  /%N%/
012055  101  102  131               .ASCII  /ABY/
012060  124  105  040               .ASCII  /TE /
012063  116  125  115               .ASCII  /NUM/
012066  102  105  122               .ASCII  /BER/
012071  072  040  045               .ASCII  /: %/
012074  104  063  000               .ASCII  /D3/<00>
012077  000                         .ASCII  <00>
012100  045  116  045       P.AGD:  .ASCII  /%N%/
012103  101  122  101               .ASCII  /ARA/
012106  116  104  117               .ASCII  /NDO/
012111  115  040  127               .ASCII  /M W/
012114  122  111  124               .ASCII  /RIT/
012117  124  105  116               .ASCII  /TEN/
012122  040  127  117               .ASCII  / WO/
012125  122  104  040               .ASCII  /RD /
012130  072  045  102               .ASCII  /:%B/
012133  061  066  000               .ASCII  /16/<00>
012136  045  116  045       P.AGE:  .ASCII  /%N%/
012141  101  122  101               .ASCII  /ARA/
012144  116  104  117               .ASCII  /NDO/
012147  115  040  122               .ASCII  /M R/
012152  105  101  104               .ASCII  /EAD/
012155  040  127  117               .ASCII  / WO/
012160  122  104  040               .ASCII  /RD /
012163  142  151  156               .ASCII  /bin/
012166  072  045  102               .ASCII  /:%B/
```

```
012171  061  066  045            .ASCII  /16%/
012174  101  040  157            .ASCII  /A o/
012177  143  164  072            .ASCII  /ct:/
012202  045  117  066            .ASCII  /%O6/
012205  000                      .ASCII  <00>
012206  045  116  045  P.AGF:     .ASCII  /%N%/
012211  101  103  122            .ASCII  /ACR/
012214  116  040  072            .ASCII  /N :/
012217  040  045  117            .ASCII  / %O/
012222  066  000                 .ASCII  /6/<00>
012224  045  101  040  P.AGG:     .ASCII  /%A /
012227  055  040  125            .ASCII  /- U/
012232  116  113  116            .ASCII  /NKN/
012235  117  127  116            .ASCII  /OWN/
012240  040  072  040            .ASCII  / : /
012243  045  104  062            .ASCII  /%D2/
012246  000  000                 .ASCII  <00><00>
012250  045  101  040  P.AGH:     .ASCII  /%A /
012253  055  040  125            .ASCII  /- U/
012256  116  113  116            .ASCII  /NKN/
012261  117  127  116            .ASCII  /OWN/
012264  040  103  117            .ASCII  / CO/
012267  116  116  105            .ASCII  /NNE/
012272  103  124  111            .ASCII  /CTI/
012275  117  116  040            .ASCII  /ON /
012300  111  104  072            .ASCII  /ID:/
012303  040  045  104            .ASCII  / %D/
012306  063  045  101            .ASCII  /3%A/
012311  040  055  000            .ASCII  / -/<00>
012314  045  116  045  P.AGI:     .ASCII  /%N%/
012317  101  103  117            .ASCII  /ACO/
012322  116  124  122            .ASCII  /NTR/
012325  117  114  114            .ASCII  /OLL/
012330  105  122  040            .ASCII  /ER /
012333  106  114  101            .ASCII  /FLA/
012336  107  123  072            .ASCII  /GS:/
012341  000                      .ASCII  <00>
012342  045  116  045  P.AGJ:     .ASCII  /%N%/
012345  101  125  116            .ASCII  /AUN/
012350  111  124  040            .ASCII  /IT /
012353  106  114  101            .ASCII  /FLA/
012356  107  123  072            .ASCII  /GS:/
012361  000                      .ASCII  <00>
012362  045  116  045  P.AGK:     .ASCII  /%N%/
012365  101  105  116            .ASCII  /AEN/
012370  104  040  115            .ASCII  /D M/
012373  105  123  123            .ASCII  /ESS/
012376  101  107  105            .ASCII  /AGE/
012401  040  106  114            .ASCII  / FL/
012404  101  107  123            .ASCII  /AGS/
012407  072  000  000            .ASCII  /:/<00><00>
012412  045  116  045  P.AGL:     .ASCII  /%N%/
012415  101  040  104            .ASCII  /A D/
```

```
012420  122  111  126            .ASCII  /RIV/
012423  105  122  040            .ASCII  /ER /
012426  103  117  116            .ASCII  /CON/
012431  124  122  117            .ASCII  /TRO/
012434  114  114  105            .ASCII  /LLE/
012437  122  040  124            .ASCII  /R T/
012442  101  102  114            .ASCII  /ABL/
012445  105  040  075            .ASCII  /E =/
012450  040  101  104            .ASCII  / AD/
012453  104  122  072            .ASCII  /DR:/
012456  040  045  104            .ASCII  / %D/
012461  066  045  116            .ASCII  /6%N/
012464  000  000                 .ASCII  <00><00>
012466  045  116  045  P.AGM:     .ASCII  /%N%/
012471  101  040  103            .ASCII  /A C/
012474  115  104  040            .ASCII  /MD /
012477  111  116  124            .ASCII  /INT/
012502  054  040  122            .ASCII  /, R/
012505  123  120  040            .ASCII  /SP /
012510  111  116  124            .ASCII  /INT/
012513  054  040  103            .ASCII  /, C/
012516  117  115  115            .ASCII  /OMM/
012521  101  116  104            .ASCII  /AND/
012524  040  122  111            .ASCII  / RI/
012527  116  107  040            .ASCII  /NG /
012532  075  040  101            .ASCII  /= A/
012535  104  104  122            .ASCII  /DDR/
012540  072  040  045            .ASCII  /: %/
012543  104  066  045            .ASCII  /D6%/
012546  116  000                 .ASCII  /N/<00>
012550  045  116  045  P.AGN:     .ASCII  /%N%/
012553  101  040  101            .ASCII  /A A/
012556  114  114  040            .ASCII  /LL /
012561  120  101  103            .ASCII  /PAC/
012564  113  105  124            .ASCII  /KET/
012567  123  040  111            .ASCII  /S I/
012572  116  040  115            .ASCII  /N M/
012575  105  123  123            .ASCII  /ESS/
012600  101  107  105            .ASCII  /AGE/
012603  040  101  122            .ASCII  / AR/
012606  105  101  000            .ASCII  /EA/<00>
012611  000                      .ASCII  <00>
012612  045  116  045  P.AGO:     .ASCII  /%N%/
012615  101  040  101            .ASCII  /A A/
012620  114  114  040            .ASCII  /LL /
012623  122  105  124            .ASCII  /RET/
012626  125  122  116            .ASCII  /URN/
012631  040  120  101            .ASCII  / PA/
012634  103  113  105            .ASCII  /CKE/
012637  124  123  040            .ASCII  /TS /
012642  111  116  040            .ASCII  /IN /
012645  101  122  105            .ASCII  /ARE/
012650  101  000                 .ASCII  /A/<00>
```

```
012652  045  116  045  P.AGP:  .ASCII  /%N%/
012655  101  040  101          .ASCII  /A A/
012660  104  104  122          .ASCII  /DDR/
012663  072  040  045          .ASCII  /: %/
012666  104  066  045          .ASCII  /D6%/
012671  101  040  040          .ASCII  /A  /
012674  040  040  040          .ASCII  /   /
012677  120  101  103          .ASCII  /PAC/
012702  113  105  124          .ASCII  /KET/
012705  040  075  040          .ASCII  / = /
012710  045  116  000          .ASCII  /%N/<00>
012713  000                    .ASCII  <00>
012714  045  116  045  P.AGQ:  .ASCII  /%N%/
012717  101  040  101          .ASCII  /A A/
012722  104  104  122          .ASCII  /DDR/
012725  040  117  106          .ASCII  / OF/
012730  040  122  105          .ASCII  / RE/
012733  123  120  117          .ASCII  /SPO/
012736  116  123  105          .ASCII  /NSE/
012741  040  122  111          .ASCII  / RI/
012744  116  107  040          .ASCII  /NG /
012747  124  117  040          .ASCII  /TO /
012752  102  105  040          .ASCII  /BE /
012755  120  117  114          .ASCII  /POL/
012760  114  105  104          .ASCII  /LED/
012763  040  045  104          .ASCII  / %D/
012766  066  000                .ASCII  /6/<00>
012770  045  116  045  P.AGR:  .ASCII  /%N%/
012773  101  040  101          .ASCII  /A A/
012776  104  104  122          .ASCII  /DDR/
013001  040  117  106          .ASCII  / OF/
013004  040  115  105          .ASCII  / ME/
013007  123  123  101          .ASCII  /SSA/
013012  107  105  040          .ASCII  /GE /
013015  120  101  103          .ASCII  /PAC/
013020  113  105  124          .ASCII  /KET/
013023  040  122  105          .ASCII  / RE/
013026  123  120  117          .ASCII  /SPO/
013031  116  123  105          .ASCII  /NSE/
013034  040  122  111          .ASCII  / RI/
013037  116  107  040          .ASCII  /NG /
013042  123  114  117          .ASCII  /SLO/
013045  124  040  120          .ASCII  /T P/
013050  117  111  116          .ASCII  /OIN/
013053  124  123  040          .ASCII  /TS /
013056  124  117  040          .ASCII  /TO /
013061  045  104  066          .ASCII  /%D6/
013064  000  000                .ASCII  <00><00>
013066  045  116  045  P.AGS:  .ASCII  /%N%/
013071  101  040  101          .ASCII  /A A/
013074  104  104  122          .ASCII  /DDR/
013077  040  117  106          .ASCII  / OF/
013102  040  115  105          .ASCII  / ME/
```

```
013105  123  123  101                .ASCII  /SSA/
013110  107  105  040                .ASCII  /GE /
013113  120  101  103                .ASCII  /PAC/
013116  113  105  124                .ASCII  /KET/
013121  040  103  117                .ASCII  / CO/
013124  115  115  101                .ASCII  /MMA/
013127  116  104  040                .ASCII  /ND /
013132  122  111  116                .ASCII  /RIN/
013135  107  040  123                .ASCII  /G S/
013140  114  117  124                .ASCII  /LOT/
013143  040  120  117                .ASCII  / PO/
013146  111  116  124                .ASCII  /INT/
013151  123  040  124                .ASCII  /S T/
013154  117  040  045                .ASCII  /O %/
013157  104  066  000                .ASCII  /D6/<00>
013162  045  116  045      P.AGT:    .ASCII  /%N%/
013165  101  104  125                .ASCII  /ADU/
013170  120  114  111                .ASCII  /PLI/
013173  103  101  124                .ASCII  /CAT/
013176  105  040  125                .ASCII  /E U/
013201  116  111  124                .ASCII  /NIT/
013204  072  045  104                .ASCII  /:%D/
013207  062  045  101                .ASCII  /2%A/
013212  040  101  124                .ASCII  / AT/
013215  040  111  120                .ASCII  / IP/
013220  072  040  045                .ASCII  /: %/
013223  117  066  000                .ASCII  /O6/<00>
013226  045  116  045      P.AGU:    .ASCII  /%N%/
013231  101  115  117                .ASCII  /AMO/
013234  122  105  040                .ASCII  /RE /
013237  124  110  101                .ASCII  /THA/
013242  116  040  045                .ASCII  /N %/
013245  104  061  045                .ASCII  /D1%/
013250  101  040  104                .ASCII  /A D/
013253  111  106  106                .ASCII  /IFF/
013256  105  122  105                .ASCII  /ERE/
013261  116  124  040                .ASCII  /NT /
013264  111  120  040                .ASCII  /IP /
013267  101  104  104                .ASCII  /ADD/
013272  122  105  123                .ASCII  /RES/
013275  123  105  123                .ASCII  /SES/
013300  000  000                     .ASCII  <00><00>
013302  045  101  055      P.AGV:    .ASCII  /%A-/
013305  040  123  105                .ASCII  / SE/
013310  121  125  105                .ASCII  /QUE/
013313  116  124  111                .ASCII  /NTI/
013316  101  114  000                .ASCII  /AL/<00>
013321  000                         .ASCII  <00>
013322  045  101  055      P.AGW:    .ASCII  /%A-/
013325  040  103  122                .ASCII  / CR/
013330  105  104  111                .ASCII  /EDI/
013333  124  040  116                .ASCII  /T N/
013336  117  124  111                .ASCII  /OTI/
```

```
013341  106     111     103                     .ASCII  /FIC/
013344  101     124     111                     .ASCII  /ATI/
013347  117     116     000                     .ASCII  /ON/<00>
013352  045     101     055     P.AGX:          .ASCII  /%A-/
013355  040     115     101                     .ASCII  / MA/
013360  111     116     124                     .ASCII  /INT/
013363  105     116     101                     .ASCII  /ENA/
013366  116     103     105                     .ASCII  /NCE/
013371  000                                     .ASCII  <00>
013372  045     101     055     P.AGY:          .ASCII  /%A-/
013375  040     104     101                     .ASCII  / DA/
013400  124     101     107                     .ASCII  /TAG/
013403  122     101     115                     .ASCII  /RAM/
013406  000     000                             .ASCII  <00><00>
013410  045     101     122     P.AGZ:          .ASCII  /%AR/
013413  105     101     104                     .ASCII  /EAD/
013416  000     000                             .ASCII  <00><00>
013420  045     101     127     P.AHA:          .ASCII  /%AW/
013423  122     111     124                     .ASCII  /RIT/
013426  105     000                             .ASCII  /E/<00>
013430  045     101     101     P.AHB:          .ASCII  /%AA/
013433  103     103     105                     .ASCII  /CCE/
013436  123     123     000                     .ASCII  /SS/<00>
013441  000                                     .ASCII  <00>
013442  045     101     117     P.AHC:          .ASCII  /%AO/
013445  116     040     114                     .ASCII  /N L/
013450  111     116     105                     .ASCII  /INE/
013453  000                                     .ASCII  <00>
013454  045     101     123     P.AHD:          .ASCII  /%AS/
013457  105     124     040                     .ASCII  /ET /
013462  103     117     116                     .ASCII  /CON/
013465  124     122     117                     .ASCII  /TRO/
013470  114     040     103                     .ASCII  /L C/
013473  110     101     122                     .ASCII  /HAR/
013476  056     000                             .ASCII  /./<00>
013500  045     101     107     P.AHE:          .ASCII  /%AG/
013503  105     124     040                     .ASCII  /ET /
013506  104     125     123                     .ASCII  /DUS/
013511  124     040     123                     .ASCII  /T S/
013514  124     101     124                     .ASCII  /TAT/
013517  125     123     000                     .ASCII  /US/<00>
013522  045     101     105     P.AHF:          .ASCII  /%AE/
013525  130     105     103                     .ASCII  /XEC/
013530  125     124     105                     .ASCII  /UTE/
013533  040     123     125                     .ASCII  / SU/
013536  120     120     114                     .ASCII  /PPL/
013541  111     105     104                     .ASCII  /IED/
013544  040     120     122                     .ASCII  / PR/
013547  107     000     000                     .ASCII  /G/<00><00>
013552  045     101     105     P.AHG:          .ASCII  /%AE/
013555  130     105     103                     .ASCII  /XEC/
013560  125     124     105                     .ASCII  /UTE/
013563  040     114     117                     .ASCII  / LO/
```

```
013566  103  101  114          .ASCII  /CAL/
013571  040  120  122          .ASCII  / PR/
013574  107  000               .ASCII  /G/<00>
013576  045  101  123  P.AHH:   .ASCII  /%AS/
013601  105  116  104          .ASCII  /END/
013604  040  104  101          .ASCII  / DA/
013607  124  101  000          .ASCII  /TA/<00>
013612  045  101  122  P.AHI:   .ASCII  /%AR/
013615  105  103  105          .ASCII  /ECE/
013620  111  126  105          .ASCII  /IVE/
013623  040  104  101          .ASCII  / DA/
013626  124  101  000          .ASCII  /TA/<00>
013631  000                    .ASCII  <00>
013632  045  101  101  P.AHJ:   .ASCII  /%AA/
013635  102  117  122          .ASCII  /BOR/
013640  124  000               .ASCII  /T/<00>
013642  045  101  123  P.AHK:   .ASCII  /%AS/
013645  120  111  116          .ASCII  /PIN/
013650  055  104  117          .ASCII  /-DO/
013653  127  116  040          .ASCII  /WN /
013656  111  107  116          .ASCII  /IGN/
013661  117  122  105          .ASCII  /ORE/
013664  104  000               .ASCII  /D/<00>
013666  045  101  123  P.AHL:   .ASCII  /%AS/
013671  124  111  114          .ASCII  /TIL/
013674  114  040  103          .ASCII  /L C/
013677  117  116  116          .ASCII  /ONN/
013702  105  103  124          .ASCII  /ECT/
013705  105  104  000          .ASCII  /ED/<00>
013710  045  101  104  P.AHM:   .ASCII  /%AD/
013713  125  120  114          .ASCII  /UPL/
013716  111  103  101          .ASCII  /ICA/
013721  124  105  040          .ASCII  /TE /
013724  125  116  111          .ASCII  /UNI/
013727  124  040  116          .ASCII  /T N/
013732  125  115  102          .ASCII  /UMB/
013735  105  122  000          .ASCII  /ER/<00>
013740  045  101  101  P.AHN:   .ASCII  /%AA/
013743  114  122  105          .ASCII  /LRE/
013746  101  104  131          .ASCII  /ADY/
013751  040  117  116          .ASCII  / ON/
013754  114  111  116          .ASCII  /LIN/
013757  105  000  000          .ASCII  /E/<00><00>
013762  045  101  123  P.AHO:   .ASCII  /%AS/
013765  124  111  114          .ASCII  /TIL/
013770  114  040  117          .ASCII  /L O/
013773  116  114  111          .ASCII  /NLI/
013776  116  105  000          .ASCII  /NE/<00>
014001  000                    .ASCII  <00>
014002  045  101  125  P.AHP:   .ASCII  /%AU/
014005  116  111  124          .ASCII  /NIT/
014010  040  125  116          .ASCII  / UN/
014013  113  116  117          .ASCII  /KNO/
```

```
014016  127  116  040            .ASCII  /WN /
014021  117  122  040            .ASCII  /OR /
014024  117  116  114            .ASCII  /ONL/
014027  111  116  105            .ASCII  /INE/
014032  040  124  117            .ASCII  / TO/
014035  040  101  116            .ASCII  / AN/
014040  117  124  110            .ASCII  /OTH/
014043  105  122  040            .ASCII  /ER /
014046  103  117  116            .ASCII  /CON/
014051  124  122  117            .ASCII  /TRO/
014054  114  114  105            .ASCII  /LLE/
014057  122  000  000            .ASCII  /R/<00><00>
014062  045  101  116  P.AHQ:     .ASCII  /%AN/
014065  117  040  126            .ASCII  /O V/
014070  117  114  125            .ASCII  /OLU/
014073  115  105  040            .ASCII  /ME /
014076  115  117  125            .ASCII  /MOU/
014101  116  124  105            .ASCII  /NTE/
014104  104  040  117            .ASCII  /D O/
014107  122  040  104            .ASCII  /R D/
014112  122  111  126            .ASCII  /RIV/
014115  105  040  104            .ASCII  /E D/
014120  111  123  101            .ASCII  /ISA/
014123  102  114  105            .ASCII  /BLE/
014126  104  040  102            .ASCII  /D B/
014131  131  040  123            .ASCII  /Y S/
014134  127  111  124            .ASCII  /WIT/
014137  103  110  000            .ASCII  /CH/<00>
014142  045  101  125  P.AHR:     .ASCII  /%AU/
014145  116  111  124            .ASCII  /NIT/
014150  040  111  116            .ASCII  / IN/
014153  117  120  105            .ASCII  /OPE/
014156  122  101  124            .ASCII  /RAT/
014161  111  126  105            .ASCII  /IVE/
014164  040  050  122            .ASCII  / (R/
014167  104  065  061            .ASCII  /D51/
014172  040  127  122            .ASCII  / WR/
014175  111  124  105            .ASCII  /ITE/
014200  040  106  101            .ASCII  / FA/
014203  125  114  124            .ASCII  /ULT/
014206  000  000                 .ASCII  <00><00>
014210  045  101  125  P.AHS:     .ASCII  /%AU/
014213  116  111  124            .ASCII  /NIT/
014216  040  104  111            .ASCII  / DI/
014221  123  101  102            .ASCII  /SAB/
014224  114  105  104            .ASCII  /LED/
014227  040  102  131            .ASCII  / BY/
014232  040  106  111            .ASCII  / FI/
014235  105  114  104            .ASCII  /ELD/
014240  040  123  105            .ASCII  / SE/
014243  122  126  111            .ASCII  /RVI/
014246  103  105  040            .ASCII  /CE /
014251  117  122  040            .ASCII  /OR /
```

```
014254  111  116  124                .ASCII  /INT/
014257  105  122  116                .ASCII  /ERN/
014262  101  114  040                .ASCII  /AL /
014265  104  111  101                .ASCII  /DIA/
014270  107  116  117                .ASCII  /GNO/
014273  123  124  111                .ASCII  /STI/
014276  103  123  000                .ASCII  /CS/<00>
014301  000                        .ASCII  <00>
014302  045  101  042   P.AHT:      .ASCII  /%A"/
014305  106  117  122                .ASCII  /FOR/
014310  103  105  104                .ASCII  /CED/
014313  040  105  122                .ASCII  / ER/
014316  122  117  122                .ASCII  /ROR/
014321  042  040  104                .ASCII  /" D/
014324  105  124  105                .ASCII  /ETE/
014327  103  124  105                .ASCII  /CTE/
014332  104  040  127                .ASCII  /D W/
014335  110  111  114                .ASCII  /HIL/
014340  105  040  101                .ASCII  /E A/
014343  103  103  105                .ASCII  /CCE/
014346  123  123  111                .ASCII  /SSI/
014351  116  107  040                .ASCII  /NG /
014354  106  103  124                .ASCII  /FCT/
014357  040  117  122                .ASCII  / OR/
014362  040  122  103                .ASCII  / RC/
014365  124  000  000                .ASCII  /T/<00><00>
014370  045  101  123   P.AHU:      .ASCII  /%AS/
014373  105  103  124                .ASCII  /ECT/
014376  117  122  040                .ASCII  /OR /
014401  127  122  111                .ASCII  /WRI/
014404  124  124  105                .ASCII  /TTE/
014407  116  040  127                .ASCII  /N W/
014412  111  124  110                .ASCII  /ITH/
014415  040  042  106                .ASCII  / "F/
014420  117  122  103                .ASCII  /ORC/
014423  105  104  040                .ASCII  /ED /
014426  105  122  122                .ASCII  /ERR/
014431  117  122  042                .ASCII  /OR"/
014434  040  115  117                .ASCII  / MO/
014437  104  111  106                .ASCII  /DIF/
014442  111  105  122                .ASCII  /IER/
014445  000                        .ASCII  <00>
014446  045  101  106   P.AHV:      .ASCII  /%AF/
014451  103  124  040                .ASCII  /CT /
014454  117  122  040                .ASCII  /OR /
014457  122  103  124                .ASCII  /RCT/
014462  040  125  116                .ASCII  / UN/
014465  122  105  101                .ASCII  /REA/
014470  104  101  102                .ASCII  /DAB/
014473  114  105  040                .ASCII  /LE /
014476  055  040  111                .ASCII  /- I/
014501  116  126  101                .ASCII  /NVA/
014504  114  111  104                .ASCII  /LID/
```

```
014507  040  123  105                .ASCII  / SE/
014512  103  124  117                .ASCII  /CTO/
014515  122  040  110                .ASCII  /R H/
014520  105  101  104                .ASCII  /EAD/
014523  105  122  000                .ASCII  /ER/<00>
014526  045  101  110      P.AHW:    .ASCII  /%AH/
014531  105  101  104                .ASCII  /EAD/
014534  105  122  040                .ASCII  /ER /
014537  103  117  115                .ASCII  /COM/
014542  120  101  122                .ASCII  /PAR/
014545  105  040  105                .ASCII  /E E/
014550  122  122  117                .ASCII  /RRO/
014553  122  040  050                .ASCII  /R (/
014556  126  101  114                .ASCII  /VAL/
014561  111  104  040                .ASCII  /ID /
014564  110  105  101                .ASCII  /HEA/
014567  104  105  122                .ASCII  /DER/
014572  040  116  117                .ASCII  / NO/
014575  124  040  106                .ASCII  /T F/
014600  117  125  116                .ASCII  /OUN/
014603  104  051  000                .ASCII  /D)/<00>
014606  045  101  106      P.AHX:    .ASCII  /%AF/
014611  103  124  040                .ASCII  /CT /
014614  117  122  040                .ASCII  /OR /
014617  122  103  124                .ASCII  /RCT/
014622  040  125  116                .ASCII  / UN/
014625  122  105  101                .ASCII  /REA/
014630  104  101  102                .ASCII  /DAB/
014633  114  105  040                .ASCII  /LE /
014636  055  040  104                .ASCII  /- D/
014641  101  124  101                .ASCII  /ATA/
014644  040  123  131                .ASCII  / SY/
014647  116  103  040                .ASCII  /NC /
014652  124  111  115                .ASCII  /TIM/
014655  105  117  125                .ASCII  /EOU/
014660  124  000                     .ASCII  /T/<00>
014662  045  101  104      P.AHY:    .ASCII  /%AD/
014665  101  124  101                .ASCII  /ATA/
014670  040  123  131                .ASCII  / SY/
014673  116  103  040                .ASCII  /NC /
014676  116  117  124                .ASCII  /NOT/
014701  040  106  117                .ASCII  / FO/
014704  125  116  104                .ASCII  /UND/
014707  040  050  104                .ASCII  / (D/
014712  101  124  101                .ASCII  /ATA/
014715  040  123  131                .ASCII  / SY/
014720  116  103  040                .ASCII  /NC /
014723  124  111  115                .ASCII  /TIM/
014726  105  117  125                .ASCII  /EOU/
014731  124  051  000                .ASCII  /T)/<00>
014734  045  101  106      P.AHZ:    .ASCII  /%AF/
014737  103  124  040                .ASCII  /CT /
014742  117  122  040                .ASCII  /OR /
```

```
014745  122     103     124             .ASCII  /RCT/
014750  040     125     116             .ASCII  / UN/
014753  122     105     101             .ASCII  /REA/
014756  104     101     102             .ASCII  /DAB/
014761  114     105     040             .ASCII  /LE /
014764  055     040     125             .ASCII  /- U/
014767  116     103     117             .ASCII  /NCO/
014772  122     122     105             .ASCII  /RRE/
014775  103     124     101             .ASCII  /CTA/
015000  102     114     105             .ASCII  /BLE/
015003  040     105     103             .ASCII  / EC/
015006  103     040     105             .ASCII  /C E/
015011  122     122     117             .ASCII  /RRO/
015014  122     000                     .ASCII  /R/<00>
015016  045     101     125     P.AIA:  .ASCII  /%AU/
015021  116     103     117             .ASCII  /NCO/
015024  122     122     105             .ASCII  /RRE/
015027  103     124     101             .ASCII  /CTA/
015032  102     114     105             .ASCII  /BLE/
015035  040     105     103             .ASCII  / EC/
015040  103     040     105             .ASCII  /C E/
015043  122     122     117             .ASCII  /RRO/
015046  122     000                     .ASCII  /R/<00>
015050  045     101     122     P.AIB:  .ASCII  /%AR/
015053  103     124     040             .ASCII  /CT /
015056  103     117     122             .ASCII  /COR/
015061  122     125     120             .ASCII  /RUP/
015064  124     105     104             .ASCII  /TED/
015067  000                             .ASCII  <00>
015070  045     101     116     P.AIC:  .ASCII  /%AN/
015073  117     040     122             .ASCII  /O R/
015076  105     120     114             .ASCII  /EPL/
015101  101     103     105             .ASCII  /ACE/
015104  115     105     116             .ASCII  /MEN/
015107  124     040     102             .ASCII  /T B/
015112  114     117     103             .ASCII  /LOC/
015115  113     040     101             .ASCII  /K A/
015120  126     101     111             .ASCII  /VAI/
015123  114     101     102             .ASCII  /LAB/
015126  114     105     040             .ASCII  /LE /
015131  050     122     103             .ASCII  /(RC/
015134  124     040     106             .ASCII  /T F/
015137  125     114     114             .ASCII  /ULL/
015142  051     000                     .ASCII  /)/<00>
015144  045     101     104     P.AID:  .ASCII  /%AD/
015147  111     123     113             .ASCII  /ISK/
015152  040     116     117             .ASCII  / NO/
015155  124     040     106             .ASCII  /T F/
015160  117     122     115             .ASCII  /ORM/
015163  101     124     124             .ASCII  /ATT/
015166  105     104     040             .ASCII  /ED /
015171  127     111     124             .ASCII  /WIT/
015174  110     040     065             .ASCII  /H 5/
```

```
015177  061  062  040                .ASCII  /12 /
015202  102  131  124                .ASCII  /BYT/
015205  105  040  123                .ASCII  /E S/
015210  105  103  124                .ASCII  /ECT/
015213  117  122  123                .ASCII  /ORS/
015216  000  000                     .ASCII  <00><00>
015220  045  101  104  P.AIE:        .ASCII  /%AD/
015223  111  123  113                .ASCII  /ISK/
015226  040  116  117                .ASCII  / NO/
015231  124  040  106                .ASCII  /T F/
015234  117  122  115                .ASCII  /ORM/
015237  101  124  124                .ASCII  /ATT/
015242  105  104  040                .ASCII  /ED /
015245  117  122  040                .ASCII  /OR /
015250  106  103  124                .ASCII  /FCT/
015253  040  103  117                .ASCII  / CO/
015256  122  122  125                .ASCII  /RRU/
015261  120  124  105                .ASCII  /PTE/
015264  104  000                     .ASCII  /D/<00>
015266  045  101  117  P.AIF:        .ASCII  /%AO/
015271  116  105  040                .ASCII  /NE /
015274  123  131  115                .ASCII  /SYM/
015277  102  117  114                .ASCII  /BOL/
015302  040  105  103                .ASCII  / EC/
015305  103  040  105                .ASCII  /C E/
015310  122  122  117                .ASCII  /RRO/
015313  122  000  000                .ASCII  /R/<00><00>
015316  045  101  124  P.AIG:        .ASCII  /%AT/
015321  127  117  040                .ASCII  /WO /
015324  123  131  115                .ASCII  /SYM/
015327  102  117  114                .ASCII  /BOL/
015332  040  105  103                .ASCII  / EC/
015335  103  040  105                .ASCII  /C E/
015340  122  122  117                .ASCII  /RRO/
015343  122  000  000                .ASCII  /R/<00><00>
015346  045  101  124  P.AIH:        .ASCII  /%AT/
015351  110  122  105                .ASCII  /HRE/
015354  105  040  123                .ASCII  /E S/
015357  131  115  102                .ASCII  /YMB/
015362  117  114  040                .ASCII  /OL /
015365  105  103  103                .ASCII  /ECC/
015370  040  105  122                .ASCII  / ER/
015373  122  117  122                .ASCII  /ROR/
015376  000  000                     .ASCII  <00><00>
015400  045  101  106  P.AII:        .ASCII  /%AF/
015403  117  125  122                .ASCII  /OUR/
015406  040  123  131                .ASCII  / SY/
015411  115  102  117                .ASCII  /MBO/
015414  114  040  105                .ASCII  /L E/
015417  103  103  040                .ASCII  /CC /
015422  105  122  122                .ASCII  /ERR/
015425  117  122  000                .ASCII  /OR/<00>
015430  045  101  106  P.AIJ:        .ASCII  /%AF/
```

```
015433  111  126  105                .ASCII  /IVE/
015436  040  123  131                .ASCII  / SY/
015441  115  102  117                .ASCII  /MBO/
015444  114  040  105                .ASCII  /L E/
015447  103  103  040                .ASCII  /CC /
015452  105  122  122                .ASCII  /ERR/
015455  117  122  000                .ASCII  /OR/<00>
015460  045  101  123  P.AIK:        .ASCII  /%AS/
015463  111  130  040                .ASCII  /IX /
015466  123  131  115                .ASCII  /SYM/
015471  102  117  114                .ASCII  /BOL/
015474  040  105  103                .ASCII  / EC/
015477  103  040  105                .ASCII  /C E/
015502  122  122  117                .ASCII  /RRO/
015505  122  000  000                .ASCII  /R/<00><00>
015510  045  101  123  P.AIL:        .ASCII  /%AS/
015513  105  126  105                .ASCII  /EVE/
015516  116  040  123                .ASCII  /N S/
015521  131  115  102                .ASCII  /YMB/
015524  117  114  040                .ASCII  /OL /
015527  105  103  103                .ASCII  /ECC/
015532  040  105  122                .ASCII  / ER/
015535  122  117  122                .ASCII  /ROR/
015540  000  000                     .ASCII  <00><00>
015542  045  101  105  P.AIM:        .ASCII  /%AE/
015545  111  107  110                .ASCII  /IGH/
015550  124  040  123                .ASCII  /T S/
015553  131  115  102                .ASCII  /YMB/
015556  117  114  040                .ASCII  /OL /
015561  105  103  103                .ASCII  /ECC/
015564  040  105  122                .ASCII  / ER/
015567  122  117  122                .ASCII  /ROR/
015572  000  000                     .ASCII  <00><00>
015574  045  101  103  P.AIN:        .ASCII  /%AC/
015577  117  122  122                .ASCII  /ORR/
015602  105  103  124                .ASCII  /ECT/
015605  101  102  114                .ASCII  /ABL/
015610  105  040  105                .ASCII  /E E/
015613  122  122  117                .ASCII  /RRO/
015616  122  040  111                .ASCII  /R I/
015621  116  040  105                .ASCII  /N E/
015624  103  103  040                .ASCII  /CC /
015627  106  111  105                .ASCII  /FIE/
015632  114  104  000                .ASCII  /LD/<00>
015635  000                         .ASCII  <00>
015636  045  101  125  P.AIO:        .ASCII  /%AU/
015641  116  111  124                .ASCII  /NIT/
015644  040  123  117                .ASCII  / SO/
015647  106  124  127                .ASCII  /FTW/
015652  101  122  105                .ASCII  /ARE/
015655  040  127  122                .ASCII  / WR/
015660  111  124  105                .ASCII  /ITE/
015663  040  120  122                .ASCII  / PR/
```

```
015666  117  124  105                .ASCII  /OTE/
015671  103  124  105                .ASCII  /CTE/
015674  104  000                     .ASCII  /D/<00>
015676  045  101  125        P.AIP:  .ASCII  /%AU/
015701  116  111  124                .ASCII  /NIT/
015704  040  110  101                .ASCII  / HA/
015707  122  104  127                .ASCII  /RDW/
015712  101  122  105                .ASCII  /ARE/
015715  040  127  122                .ASCII  / WR/
015720  111  124  105                .ASCII  /ITE/
015723  040  120  122                .ASCII  / PR/
015726  117  124  105                .ASCII  /OTE/
015731  103  124  105                .ASCII  /CTE/
015734  104  000                     .ASCII  /D/<00>
015736  045  101  117        P.AIQ:  .ASCII  /%AO/
015741  104  104  040                .ASCII  /DD /
015744  124  122  101                .ASCII  /TRA/
015747  116  123  106                .ASCII  /NSF/
015752  105  122  040                .ASCII  /ER /
015755  101  104  104                .ASCII  /ADD/
015760  122  105  123                .ASCII  /RES/
015763  123  000  000                .ASCII  /S/<00><00>
015766  045  101  117        P.AIR:  .ASCII  /%AO/
015771  104  104  040                .ASCII  /DD /
015774  102  131  124                .ASCII  /BYT/
015777  105  040  103                .ASCII  /E C/
016002  117  125  116                .ASCII  /OUN/
016005  124  000  000                .ASCII  /T/<00><00>
016010  045  101  116        P.AIS:  .ASCII  /%AN/
016013  117  116  055                .ASCII  /ON-/
016016  105  130  111                .ASCII  /EXI/
016021  123  124  105                .ASCII  /STE/
016024  116  124  040                .ASCII  /NT /
016027  110  117  123                .ASCII  /HOS/
016032  124  040  115                .ASCII  /T M/
016035  105  115  117                .ASCII  /EMO/
016040  122  131  000                .ASCII  /RY/<00>
016043  000                         .ASCII  <00>
016044  045  101  110        P.AIT:  .ASCII  /%AH/
016047  117  123  124                .ASCII  /OST/
016052  040  115  105                .ASCII  / ME/
016055  115  117  122                .ASCII  /MOR/
016060  131  040  120                .ASCII  /Y P/
016063  101  122  111                .ASCII  /ARI/
016066  124  131  040                .ASCII  /TY /
016071  105  122  122                .ASCII  /ERR/
016074  117  122  000                .ASCII  /OR/<00>
016077  000                         .ASCII  <00>
016100  045  101  103        P.AIU:  .ASCII  /%AC/
016103  117  115  115                .ASCII  /OMM/
016106  101  116  104                .ASCII  /AND/
016111  040  124  111                .ASCII  / TI/
016114  115  117  125                .ASCII  /MOU/
```

```
016117  124  040  117          .ASCII  /T O/
016122  122  040  122          .ASCII  /R R/
016125  105  124  122          .ASCII  /ETR/
016130  131  040  114          .ASCII  /Y L/
016133  111  115  111          .ASCII  /IMI/
016136  124  040  105          .ASCII  /T E/
016141  130  103  105          .ASCII  /XCE/
016144  105  104  105          .ASCII  /EDE/
016147  104  000  000          .ASCII  /D/<00><00>
016152  045  101  123  P.AIV:  .ASCII  /%AS/
016155  105  122  111          .ASCII  /ERI/
016160  101  114  111          .ASCII  /ALI/
016163  132  105  122          .ASCII  /ZER/
016166  057  104  105          .ASCII  <57>/DE/
016171  123  105  122          .ASCII  /SER/
016174  111  101  114          .ASCII  /IAL/
016177  111  132  105          .ASCII  /IZE/
016202  122  040  117          .ASCII  /R O/
016205  126  105  122          .ASCII  /VER/
016210  122  125  116          .ASCII  /RUN/
016213  040  117  122          .ASCII  / OR/
016216  040  125  116          .ASCII  / UN/
016221  104  105  122          .ASCII  /DER/
016224  122  125  116          .ASCII  /RUN/
016227  000                    .ASCII  <00>
016230  045  101  105  P.AIW:  .ASCII  /%AE/
016233  104  103  040          .ASCII  /DC /
016236  105  122  122          .ASCII  /ERR/
016241  117  122  000          .ASCII  /OR/<00>
016244  045  101  111  P.AIX:  .ASCII  /%AI/
016247  116  103  117          .ASCII  /NCO/
016252  116  123  111          .ASCII  /NSI/
016255  123  124  105          .ASCII  /STE/
016260  116  124  040          .ASCII  /NT /
016263  111  116  124          .ASCII  /INT/
016266  105  122  116          .ASCII  /ERN/
016271  101  114  040          .ASCII  /AL /
016274  104  101  124          .ASCII  /DAT/
016277  101  040  123          .ASCII  /A S/
016302  124  122  125          .ASCII  /TRU/
016305  103  124  125          .ASCII  /CTU/
016310  122  105  000          .ASCII  /RE/<00>
016313  000                    .ASCII  <00>
016314  045  101  104  P.AIY:  .ASCII  /%AD/
016317  122  111  126          .ASCII  /RIV/
016322  105  040  103          .ASCII  /E C/
016325  117  115  115          .ASCII  /OMM/
016330  101  116  104          .ASCII  /AND/
016333  040  124  111          .ASCII  / TI/
016336  115  105  117          .ASCII  /MEO/
016341  125  124  040          .ASCII  /UT /
016344  050  116  117          .ASCII  /(NO/
016347  040  122  105          .ASCII  / RE/
```

```
016352  123  120  117                .ASCII  /SPO/
016355  116  123  105                .ASCII  /NSE/
016360  040  117  122                .ASCII  / OR/
016363  040  123  105                .ASCII  / SE/
016366  105  113  040                .ASCII  /EK /
016371  111  116  103                .ASCII  /INC/
016374  117  115  120                .ASCII  /OMP/
016377  114  105  124                .ASCII  /LET/
016402  105  051  000                .ASCII  /E)/<00>
016405  000                        .ASCII  <00>
016406  045  101  103     P.AIZ:     .ASCII  /%AC/
016411  117  116  124                .ASCII  /ONT/
016414  122  117  114                .ASCII  /ROL/
016417  114  105  122                .ASCII  /LER/
016422  040  104  105                .ASCII  / DE/
016425  124  105  103                .ASCII  /TEC/
016430  124  105  104                .ASCII  /TED/
016433  040  124  122                .ASCII  / TR/
016436  101  116  123                .ASCII  /ANS/
016441  115  111  123                .ASCII  /MIS/
016444  123  111  117                .ASCII  /SIO/
016447  116  040  117                .ASCII  /N O/
016452  122  040  120                .ASCII  /R P/
016455  122  117  124                .ASCII  /ROT/
016460  117  103  117                .ASCII  /OCO/
016463  114  040  105                .ASCII  /L E/
016466  122  122  117                .ASCII  /RRO/
016471  122  000  000                .ASCII  /R/<00><00>
016474  045  101  120     P.AJA:     .ASCII  /%AP/
016477  117  123  111                .ASCII  /OSI/
016502  124  111  117                .ASCII  /TIO/
016505  116  040  105                .ASCII  /N E/
016510  122  122  117                .ASCII  /RRO/
016513  122  040  050                .ASCII  /R (/
016516  115  111  123                .ASCII  /MIS/
016521  055  123  105                .ASCII  /-SE/
016524  105  113  051                .ASCII  /EK)/
016527  000                        .ASCII  <00>
016530  045  101  114     P.AJB:     .ASCII  /%AL/
016533  117  123  124                .ASCII  /OST/
016536  040  122  105                .ASCII  / RE/
016541  101  104  057                .ASCII  /AD/<57>
016544  127  122  111                .ASCII  /WRI/
016547  124  105  040                .ASCII  /TE /
016552  122  105  101                .ASCII  /REA/
016555  104  131  040                .ASCII  /DY /
016560  104  125  122                .ASCII  /DUR/
016563  111  116  107                .ASCII  /ING/
016566  057  102  105                .ASCII  <57>/BE/
016571  124  127  105                .ASCII  /TWE/
016574  105  116  040                .ASCII  /EN /
016577  124  122  101                .ASCII  /TRA/
016602  116  123  106                .ASCII  /NSF/
```

```
016605  105  122  123                 .ASCII  /ERS/
016610  000  000                      .ASCII  <00><00>
016612  045  101  104       P.AJC:    .ASCII  /%AD/
016615  122  111  126                 .ASCII  /RIV/
016620  105  040  103                 .ASCII  /E C/
016623  114  117  103                 .ASCII  /LOC/
016626  113  040  104                 .ASCII  /K D/
016631  122  117  120                 .ASCII  /ROP/
016634  117  125  124                 .ASCII  /OUT/
016637  000                          .ASCII  <00>
016640  045  101  114       P.AJD:    .ASCII  /%AL/
016643  117  123  124                 .ASCII  /OST/
016646  040  122  105                 .ASCII  / RE/
016651  103  105  111                 .ASCII  /CEI/
016654  126  105  122                 .ASCII  /VER/
016657  040  122  105                 .ASCII  / RE/
016662  101  104  131                 .ASCII  /ADY/
016665  040  102  105                 .ASCII  / BE/
016670  124  127  105                 .ASCII  /TWE/
016673  105  116  040                 .ASCII  /EN /
016676  123  105  103                 .ASCII  /SEC/
016701  124  117  122                 .ASCII  /TOR/
016704  123  000                      .ASCII  /S/<00>
016706  045  101  104       P.AJE:    .ASCII  /%AD/
016711  122  111  126                 .ASCII  /RIV/
016714  105  040  104                 .ASCII  /E D/
016717  105  124  105                 .ASCII  /ETE/
016722  103  124  105                 .ASCII  /CTE/
016725  104  040  105                 .ASCII  /D E/
016730  122  122  117                 .ASCII  /RRO/
016733  122  000  000                 .ASCII  /R/<00><00>
016736  045  101  103       P.AJF:    .ASCII  /%AC/
016741  117  116  124                 .ASCII  /ONT/
016744  122  117  114                 .ASCII  /ROL/
016747  114  105  122                 .ASCII  /LER/
016752  040  104  105                 .ASCII  / DE/
016755  124  105  103                 .ASCII  /TEC/
016760  124  105  104                 .ASCII  /TED/
016763  040  120  125                 .ASCII  / PU/
016766  114  123  105                 .ASCII  /LSE/
016771  040  117  122                 .ASCII  / OR/
016774  040  123  124                 .ASCII  / ST/
016777  101  124  105                 .ASCII  /ATE/
017002  040  120  101                 .ASCII  / PA/
017005  122  111  124                 .ASCII  /RIT/
017010  131  040  105                 .ASCII  /Y E/
017013  122  122  117                 .ASCII  /RRO/
017016  122  000                      .ASCII  /R/<00>
017020  045  116  045       P.AJG:    .ASCII  /%N%/
017023  101  040  174                 .ASCII  /A /<174>
017026  040  102  141                 .ASCII  / Ba/
017031  144  040  102                 .ASCII  /d B/
017034  154  157  143                 .ASCII  /loc/
```

ZRQAM1 RD/RX EXERCISER 21-Jul-1983 15:25:58 VAX-11 Bliss-16 V3-555
V01.2 PROTECTION TABLE 21-Jul-1983 15:19:20 DISK$USER2:[DIETZ.RELEASE]ZRQABO.BL1;10 (35)

```
017037  153  040  125                    .ASCII  /k U/
017042  156  162  145                    .ASCII  /nre/
017045  160  157  162                    .ASCII  /por/
017050  164  145  144                    .ASCII  /ted/
017053  000                             .ASCII  <00>
017054  045  116  045          P.AJH:    .ASCII  /%N%/
017057  101  040  174                    .ASCII  /A /<174>
017062  040  105  162                    .ASCII  / Er/
017065  162  157  162                    .ASCII  /ror/
017070  040  114  157                    .ASCII  / Lo/
017073  147  040  107                    .ASCII  /g G/
017076  145  156  145                    .ASCII  /ene/
017101  162  141  164                    .ASCII  /rat/
017104  145  144  000                    .ASCII  /ed/<00>
017107  000                             .ASCII  <00>
017110  045  116  045          P.AJI:    .ASCII  /%N%/
017113  101  040  174                    .ASCII  /A /<174>
017116  040  123  145                    .ASCII  / Se/
017121  162  151  157                    .ASCII  /rio/
017124  165  163  040                    .ASCII  /us /
017127  105  170  143                    .ASCII  /Exc/
017132  145  160  164                    .ASCII  /ept/
017135  151  157  156                    .ASCII  /ion/
017140  000  000                        .ASCII  <00><00>
017142  045  116  045          P.AJJ:    .ASCII  /%N%/
017145  101  174  040                    .ASCII  /A/<174>/ /
017150  105  156  141                    .ASCII  /Ena/
017153  142  154  145                    .ASCII  /ble/
017156  040  101  164                    .ASCII  / At/
017161  164  145  156                    .ASCII  /ten/
017164  164  151  157                    .ASCII  /tio/
017167  156  040  115                    .ASCII  /n M/
017172  145  163  163                    .ASCII  /ess/
017175  141  147  145                    .ASCII  /age/
017200  163  000                        .ASCII  /s/<00>
017202  045  116  045          P.AJK:    .ASCII  /%N%/
017205  101  174  040                    .ASCII  /A/<174>/ /
017210  105  156  141                    .ASCII  /Ena/
017213  142  154  145                    .ASCII  /ble/
017216  040  115  151                    .ASCII  / Mi/
017221  163  143  145                    .ASCII  /sce/
017224  154  154  141                    .ASCII  /lla/
017227  156  145  157                    .ASCII  /neo/
017232  165  163  040                    .ASCII  /us /
017235  105  162  162                    .ASCII  /Err/
017240  157  162  040                    .ASCII  /or /
017243  114  157  147                    .ASCII  /Log/
017246  040  115  145                    .ASCII  / Me/
017251  163  163  141                    .ASCII  /ssa/
017254  147  145  163                    .ASCII  /ges/
017257  000                             .ASCII  <00>
017260  045  116  045          P.AJL:    .ASCII  /%N%/
017263  101  174  040                    .ASCII  /A/<174>/ /
```

```
017266  105  156  141          .ASCII  /Ena/
017271  142  154  145          .ASCII  /ble/
017274  040  117  164          .ASCII  / Ot/
017277  150  145  162          .ASCII  /her/
017302  040  110  157          .ASCII  / Ho/
017305  163  164  163          .ASCII  /sts/
017310  040  105  162          .ASCII  / Er/
017313  162  157  162          .ASCII  /ror/
017316  040  114  157          .ASCII  / Lo/
017321  147  040  115          .ASCII  /g M/
017324  145  163  163          .ASCII  /ess/
017327  141  147  145          .ASCII  /age/
017332  163  000               .ASCII  /s/<00>
017334  045  116  045  P.AJM:   .ASCII  /%N%/
017337  101  174  040          .ASCII  /A/<174>/ /
017342  105  156  141          .ASCII  /Ena/
017345  142  154  145          .ASCII  /ble/
017350  040  124  150          .ASCII  / Th/
017353  151  163  040          .ASCII  /is /
017356  110  157  163          .ASCII  /Hos/
017361  164  163  040          .ASCII  /ts /
017364  105  162  162          .ASCII  /Err/
017367  157  162  040          .ASCII  /or /
017372  114  157  147          .ASCII  /Log/
017375  040  115  145          .ASCII  / Me/
017400  163  163  141          .ASCII  /ssa/
017403  147  145  163          .ASCII  /ges/
017406  000  000               .ASCII  <00><00>
017410  045  116  045  P.AJN:   .ASCII  /%N%/
017413  101  174  040          .ASCII  /A/<174>/ /
017416  103  157  156          .ASCII  /Con/
017421  164  162  157          .ASCII  /tro/
017424  154  154  145          .ASCII  /lle/
017427  162  040  111          .ASCII  /r I/
017432  156  151  164          .ASCII  /nit/
017435  151  141  164          .ASCII  /iat/
017440  145  144  040          .ASCII  /ed /
017443  102  141  144          .ASCII  /Bad/
017446  040  102  154          .ASCII  / Bl/
017451  157  143  153          .ASCII  /ock/
017454  040  122  160          .ASCII  / Rp/
017457  154  143  155          .ASCII  /lcm/
017462  156  164  000          .ASCII  /nt/<00>
017465  000                    .ASCII  <00>
017466  045  116  045  P.AJO:   .ASCII  /%N%/
017471  101  174  040          .ASCII  /A/<174>/ /
017474  123  150  141          .ASCII  /Sha/
017477  144  157  167          .ASCII  /dow/
017502  151  156  147          .ASCII  /ing/
017505  000                    .ASCII  <00>
017506  045  116  045  P.AJP:   .ASCII  /%N%/
017511  101  174  040          .ASCII  /A/<174>/ /
017514  065  067  066          .ASCII  /576/
```

```
017517  040  102  171                .ASCII  / By/
017522  164  145  040                .ASCII  /te /
017525  123  145  143                .ASCII  /Sec/
017530  164  157  162                .ASCII  /tor/
017533  163  000  000                .ASCII  /s/<00><00>
017536  045  116  045      P.AJQ:    .ASCII  /%N%/
017541  101  174  040                .ASCII  /A/<174>/ /
017544  103  157  155                .ASCII  /Com/
017547  160  141  162                .ASCII  /par/
017552  145  040  122                .ASCII  /e R/
017555  145  141  144                .ASCII  /ead/
017560  163  000                     .ASCII  /s/<00>
017562  045  116  045      P.AJR:    .ASCII  /%N%/
017565  101  174  040                .ASCII  /A/<174>/ /
017570  103  157  155                .ASCII  /Com/
017573  160  141  162                .ASCII  /par/
017576  145  040  127                .ASCII  /e W/
017601  162  151  164                .ASCII  /rit/
017604  145  163  000                .ASCII  /es/<00>
017607  000                         .ASCII  <00>
017610  045  116  045      P.AJS:    .ASCII  /%N%/
017613  101  174  040                .ASCII  /A/<174>/ /
017616  103  157  156                .ASCII  /Con/
017621  164  162  157                .ASCII  /tro/
017624  154  154  145                .ASCII  /lle/
017627  162  040  111                .ASCII  /r I/
017632  156  151  164                .ASCII  /nit/
017635  151  141  164                .ASCII  /iat/
017640  145  144  040                .ASCII  /ed /
017643  102  141  144                .ASCII  /Bad/
017646  040  102  154                .ASCII  / Bl/
017651  157  143  153                .ASCII  /ock/
017654  040  122  160                .ASCII  / Rp/
017657  154  143  155                .ASCII  /lcm/
017662  156  164  000                .ASCII  /nt/<00>
017665  000                         .ASCII  <00>
017666  045  116  045      P.AJT:    .ASCII  /%N%/
017671  101  174  040                .ASCII  /A/<174>/ /
017674  111  156  141                .ASCII  /Ina/
017677  143  164  151                .ASCII  /cti/
017702  166  145  040                .ASCII  /ve /
017705  123  150  141                .ASCII  /Sha/
017710  144  157  167                .ASCII  /dow/
017713  040  123  145                .ASCII  / Se/
017716  164  040  125                .ASCII  /t U/
017721  156  151  164                .ASCII  /nit/
017724  000  000                     .ASCII  <00><00>
017726  045  116  045      P.AJU:    .ASCII  /%N%/
017731  101  174  040                .ASCII  /A/<174>/ /
017734  122  145  155                .ASCII  /Rem/
017737  157  166  141                .ASCII  /ova/
017742  142  154  145                .ASCII  /ble/
017745  040  115  145                .ASCII  / Me/
```

```
017750  144     151     141             .ASCII  /dia/
017753  000                             .ASCII  <00>
017754  045     116     045     P.AJV:  .ASCII  /%N%/
017757  101     174     040             .ASCII  /A/<174>/ /
017762  123     165     160             .ASCII  /Sup/
017765  160     162     145             .ASCII  /pre/
017770  163     163     040             .ASCII  /ss /
017773  103     141     143             .ASCII  /Cac/
017776  150     151     156             .ASCII  /hin/
020001  147     040     050             .ASCII  /g (/
020004  150     151     147             .ASCII  /hig/
020007  150     040     163             .ASCII  /h s/
020012  160     145     145             .ASCII  /pee/
020015  144     051     000             .ASCII  /d)/<00>
020020  045     116     045     P.AJW:  .ASCII  /%N%/
020023  101     174     040             .ASCII  /A/<174>/ /
020026  123     165     160             .ASCII  /Sup/
020031  160     162     145             .ASCII  /pre/
020034  163     163     040             .ASCII  /ss /
020037  103     141     143             .ASCII  /Cac/
020042  150     151     156             .ASCII  /hin/
020045  147     040     050             .ASCII  /g (/
020050  154     157     167             .ASCII  /low/
020053  040     163     160             .ASCII  / sp/
020056  145     145     144             .ASCII  /eed/
020061  051     000     000             .ASCII  /)/<00><00>
020064  045     116     045     P.AJX:  .ASCII  /%N%/
020067  101     174     040             .ASCII  /A/<174>/ /
020072  127     162     151             .ASCII  /Wri/
020075  164     145     055             .ASCII  /te-/
020100  142     141     143             .ASCII  /bac/
020103  153     040     050             .ASCII  /k (/
020106  156     157     156             .ASCII  /non/
020111  055     166     157             .ASCII  /-vo/
020114  154     141     164             .ASCII  /lat/
020117  151     154     145             .ASCII  /ile/
020122  051     000                     .ASCII  /)/<00>
020124  045     116     045     P.AJY:  .ASCII  /%N%/
020127  101     174     040             .ASCII  /A/<174>/ /
020132  127     162     151             .ASCII  /Wri/
020135  164     145     040             .ASCII  /te /
020140  120     162     157             .ASCII  /Pro/
020143  164     145     143             .ASCII  /tec/
020146  164     040     050             .ASCII  /t (/
020151  150     141     162             .ASCII  /har/
020154  144     167     141             .ASCII  /dwa/
020157  162     145     051             .ASCII  /re)/
020162  000     000                     .ASCII  <00><00>
020164  045     116     045     P.AJZ:  .ASCII  /%N%/
020167  101     174     040             .ASCII  /A/<174>/ /
020172  127     162     151             .ASCII  /Wri/
020175  164     145     040             .ASCII  /te /
020200  120     162     157             .ASCII  /Pro/
```

```
020203  164  145  143                    .ASCII  /tec/
020206  164  040  050                    .ASCII  /t (/
020211  163  157  146                    .ASCII  /sof/
020214  164  167  141                    .ASCII  /twa/
020217  162  145  040                    .ASCII  /re /
020222  157  162  040                    .ASCII  /or /
020225  166  157  154                    .ASCII  /vol/
020230  165  155  145                    .ASCII  /ume/
020233  051  000  000                    .ASCII  /)/<00><00>
020236  045  116  045          P.AKA:    .ASCII  /%N%/
020241  101  174  040                    .ASCII  /A/<174>/ /
020244  065  067  066                    .ASCII  /576/
020247  040  102  171                    .ASCII  / By/
020252  164  145  040                    .ASCII  /te /
020255  123  145  143                    .ASCII  /Sec/
020260  164  157  162                    .ASCII  /tor/
020263  163  000  000                    .ASCII  /s/<00><00>
020266  045  101  040          P.AKB:    .ASCII  /%A /
020271  055  040  123                    .ASCII  /- S/
020274  125  103  103                    .ASCII  /UCC/
020277  105  123  123                    .ASCII  /ESS/
020302  000  000                         .ASCII  <00><00>
020304  045  101  040          P.AKC:    .ASCII  /%A /
020307  055  040  111                    .ASCII  /- I/
020312  116  126  101                    .ASCII  /NVA/
020315  114  111  104                    .ASCII  /LID/
020320  040  103  117                    .ASCII  / CO/
020323  115  115  101                    .ASCII  /MMA/
020326  116  104  050                    .ASCII  /ND(/
020331  123  105  122                    .ASCII  /SER/
020334  126  105  122                    .ASCII  /VER/
020337  040  156  157                    .ASCII  / no/
020342  156  111  104                    .ASCII  /nID/
020345  114  105  040                    .ASCII  /LE /
020350  157  162  040                    .ASCII  /or /
020353  156  157  040                    .ASCII  /no /
020356  155  145  144                    .ASCII  /med/
020361  151  141  040                    .ASCII  /ia /
020364  151  146  040                    .ASCII  /if /
020367  105  130  055                    .ASCII  /EX-/
020372  114  103  055                    .ASCII  /LC-/
020375  120  122  107                    .ASCII  /PRG/
020400  040  143  155                    .ASCII  / cm/
020403  144  051  000                    .ASCII  /d)/<00>
020406  045  101  040          P.AKD:    .ASCII  /%A /
020411  055  040  116                    .ASCII  /- N/
020414  117  040  122                    .ASCII  /O R/
020417  105  107  111                    .ASCII  /EGI/
020422  117  116  040                    .ASCII  /ON /
020425  101  126  101                    .ASCII  /AVA/
020430  111  114  101                    .ASCII  /ILA/
020433  102  114  105                    .ASCII  /BLE/
020436  000  000                         .ASCII  <00><00>
```

```
020440  045  101  040   P.AKE:  .ASCII  /%A /
020443  055  040  116           .ASCII  /- N/
020446  117  040  122           .ASCII  /O R/
020451  105  107  111           .ASCII  /EGI/
020454  117  116  040           .ASCII  /ON /
020457  123  125  111           .ASCII  /SUI/
020462  124  101  102           .ASCII  /TAB/
020465  114  105  000           .ASCII  /LE/<00>
020470  045  101  040   P.AKF:  .ASCII  /%A /
020473  055  040  120           .ASCII  /- P/
020476  122  117  107           .ASCII  /ROG/
020501  122  101  115           .ASCII  /RAM/
020504  040  116  117           .ASCII  / NO/
020507  124  040  113           .ASCII  /T K/
020512  116  117  127           .ASCII  /NOW/
020515  116  040  050           .ASCII  /N (/
020520  116  117  040           .ASCII  /NO /
020523  123  125  103           .ASCII  /SUC/
020526  110  040  120           .ASCII  /H P/
020531  122  117  107           .ASCII  /ROG/
020534  122  101  115           .ASCII  /RAM/
020537  040  117  116           .ASCII  / ON/
020542  040  115  105           .ASCII  / ME/
020545  104  111  101           .ASCII  /DIA/
020550  051  000               .ASCII  /)/<00>
020552  045  101  040   P.AKG:  .ASCII  /%A /
020555  055  040  114           .ASCII  /- L/
020560  117  101  104           .ASCII  /OAD/
020563  040  106  101           .ASCII  / FA/
020566  111  114  125           .ASCII  /ILU/
020571  122  105  040           .ASCII  /RE /
020574  050  111  116           .ASCII  /(IN/
020577  120  125  124           .ASCII  /PUT/
020602  040  105  122           .ASCII  / ER/
020605  122  117  122           .ASCII  /ROR/
020610  040  127  110           .ASCII  / WH/
020613  111  114  105           .ASCII  /ILE/
020616  040  114  117           .ASCII  / LO/
020621  101  104  111           .ASCII  /ADI/
020624  116  107  040           .ASCII  /NG /
020627  120  122  117           .ASCII  /PRO/
020632  107  122  101           .ASCII  /GRA/
020635  115  051  000           .ASCII  /M)/<00>
020640  045  101  040   P.AKH:  .ASCII  /%A /
020643  055  040  123           .ASCII  /- S/
020646  124  101  116           .ASCII  /TAN/
020651  104  101  114           .ASCII  /DAL/
020654  117  116  105           .ASCII  /ONE/
020657  040  050  123           .ASCII  / (S/
020662  124  101  116           .ASCII  /TAN/
020665  104  101  114           .ASCII  /DAL/
020670  117  116  105           .ASCII  /ONE/
020673  040  115  117           .ASCII  / MO/
```

```
020676  104     111     106             .ASCII  /DIF/
020701  111     105     122             .ASCII  /IER/
020704  040     116     117             .ASCII  / NO/
020707  124     040     123             .ASCII  /T S/
020712  120     105     103             .ASCII  /PEC/
020715  111     106     111             .ASCII  /IFI/
020720  105     104     040             .ASCII  /ED /
020723  106     117     122             .ASCII  /FOR/
020726  040     123     124             .ASCII  / ST/
020731  101     116     104             .ASCII  /AND/
020734  040     101     114             .ASCII  / AL/
020737  117     116     105             .ASCII  /ONE/
020742  040     120     122             .ASCII  / PR/
020745  107     056     051             .ASCII  /G.)/
020750  000     000                     .ASCII  <00><00>
020752  045     116     045     P.AKI:  .ASCII  /%N%/
020755  101     174     040             .ASCII  /A/<174>/ /
020760  117     156     145             .ASCII  /One/
020763  040     123     145             .ASCII  / Se/
020766  162     166     145             .ASCII  /rve/
020771  162     040     141             .ASCII  /r a/
020774  164     040     141             .ASCII  /t a/
020777  040     124     151             .ASCII  / Ti/
021002  155     145     000             .ASCII  /me/<00>
021005  000                             .ASCII  <00>
021006  045     116     045     P.AKJ:  .ASCII  /%N%/
021011  101     174     040             .ASCII  /A/<174>/ /
021014  103     157     156             .ASCII  /Con/
021017  164     141     151             .ASCII  /tai/
021022  156     163     040             .ASCII  /ns /
021025  114     157     143             .ASCII  /Loc/
021030  141     154     040             .ASCII  /al /
021033  115     145     144             .ASCII  /Med/
021036  151     141     000             .ASCII  /ia/<00>
021041  000                             .ASCII  <00>
021042  045     116     045     P.AKK:  .ASCII  /%N%/
021045  101     174     040             .ASCII  /A/<174>/ /
021050  105     170     145             .ASCII  /Exe/
021053  143     165     164             .ASCII  /cut/
021056  145     040     114             .ASCII  /e L/
021061  157     143     141             .ASCII  /oca/
021064  154     040     120             .ASCII  /l P/
021067  162     147     040             .ASCII  /rg /
021072  143     155     144             .ASCII  /cmd/
021075  040     151     163             .ASCII  / is/
021100  040     125     116             .ASCII  / UN/
021103  123     125     120             .ASCII  /SUP/
021106  120     117     122             .ASCII  /POR/
021111  124     105     104             .ASCII  /TED/
021114  000     000                     .ASCII  <00><00>
021116  045     116     045     P.AKL:  .ASCII  /%N%/
021121  101     174     040             .ASCII  /A/<174>/ /
021124  103     165     162             .ASCII  /Cur/
```

ZRQAM1 RD/RX EXERCISER 21-Jul-1983 15:25:58 VAX-11 Bliss-16 V3-555
V01.2 PROTECTION TABLE 21-Jul-1983 15:19:20 DISK$USER2:[DIETZ.RELEASE]ZRQAB0.BL1;10 (35)

```
021127  162  145  156                .ASCII  /ren/
021132  164  154  171                .ASCII  /tly/
021135  040  151  156                .ASCII  / in/
021140  040  101  143                .ASCII  / Ac/
021143  164  151  166                .ASCII  /tiv/
021146  145  040  123                .ASCII  /e S/
021151  164  141  164                .ASCII  /tat/
021154  145  000                     .ASCII  /e/<00>
021156  045  116  045       P.AKM:   .ASCII  /%N%/
021161  101  174  040                .ASCII  /A/<174>/ /
021164  123  164  141                .ASCII  /Sta/
021167  156  144  141                .ASCII  /nda/
021172  154  157  156                .ASCII  /lon/
021175  145  040  120                .ASCII  /e P/
021200  162  147  000                .ASCII  /rg/<00>
021203  000                         .ASCII  <00>
021204  045  116  045       P.AKN:   .ASCII  /%N%/
021207  101  174  040                .ASCII  /A/<174>/ /
021212  116  145  145                .ASCII  /Nee/
021215  144  163  040                .ASCII  /ds /
021220  157  166  145                .ASCII  /ove/
021223  162  154  141                .ASCII  /rla/
021226  171  000                     .ASCII  /y/<00>
021230  045  116  045       P.AKO:   .ASCII  /%N%/
021233  101  174  040                .ASCII  /A/<174>/ /
021236  116  145  145                .ASCII  /Nee/
021241  144  163  040                .ASCII  /ds /
021244  127  162  151                .ASCII  /Wri/
021247  164  145  141                .ASCII  /tea/
021252  142  154  145                .ASCII  /ble/
021255  057  122  145                .ASCII  <57>/Re/
021260  141  144  141                .ASCII  /ada/
021263  142  154  145                .ASCII  /ble/
021266  040  117  166                .ASCII  / Ov/
021271  145  162  154                .ASCII  /erl/
021274  141  171  000                .ASCII  /ay/<00>
021277  000                         .ASCII  <00>
021300  045  116  045       P.AKP:   .ASCII  /%N%/
021303  101  174  040                .ASCII  /A/<174>/ /
021306  125  163  145                .ASCII  /Use/
021311  163  040  123                .ASCII  /s S/
021314  164  144  040                .ASCII  /td /
021317  104  165  160                .ASCII  /Dup/
021322  040  104  151                .ASCII  / Di/
021325  141  154  157                .ASCII  /alo/
021330  147  073  040                .ASCII  /g; /
021333  122  105  103                .ASCII  /REC/
021336  057  123  105                .ASCII  <57>/SE/
021341  116  104  057                .ASCII  /ND/<57>
021344  122  105  103                .ASCII  /REC/
021347  000                         .ASCII  <00>
021350  045  116  045       P.AKQ:   .ASCII  /%N%/
021353  101  011  052                .ASCII  /A/<11>/*/
```

ZRQAM1 RD/RX EXERCISER
V01.2 PROTECTION TABLE

21-Jul-1983 15:25:58 VAX-11 Bliss-16 V3-555
21-Jul-1983 15:19:20 DISK$USER2:[DIETZ.RELEASE]ZRQAB0.BL1;10 (35)

```
021356  052  040  121                .ASCII  /* Q/
021361  125  105  123                .ASCII  /UES/
021364  124  111  117                .ASCII  /TIO/
021367  116  000  000                .ASCII  /N/<00><00>
021372  045  116  045     P.AKR:     .ASCII  /%N%/
021375  101  011  052                .ASCII  /A/<11>/*/
021400  052  040  104                .ASCII  /* D/
021403  105  106  101                .ASCII  /EFA/
021406  125  114  124                .ASCII  /ULT/
021411  040  121  125                .ASCII  / QU/
021414  105  123  124                .ASCII  /EST/
021417  111  117  116                .ASCII  /ION/
021422  000  000                     .ASCII  <00><00>
021424  045  116  045     P.AKS:     .ASCII  /%N%/
021427  101  011  052                .ASCII  /A/<11>/*/
021432  052  040  111                .ASCII  /* I/
021435  116  106  117                .ASCII  /NFO/
021440  122  115  101                .ASCII  /RMA/
021443  124  111  117                .ASCII  /TIO/
021446  116  000                     .ASCII  /N/<00>
021450  045  116  045     P.AKT:     .ASCII  /%N%/
021453  101  011  052                .ASCII  /A/<11>/*/
021456  052  040  124                .ASCII  /* T/
021461  105  122  115                .ASCII  /ERM/
021464  111  116  101                .ASCII  /INA/
021467  124  111  117                .ASCII  /TIO/
021472  116  000                     .ASCII  /N/<00>
021474  045  116  045     P.AKU:     .ASCII  /%N%/
021477  101  011  052                .ASCII  /A/<11>/*/
021502  052  040  106                .ASCII  /* F/
021505  101  124  101                .ASCII  /ATA/
021510  114  040  105                .ASCII  /L E/
021513  122  122  117                .ASCII  /RRO/
021516  122  000                     .ASCII  /R/<00>
021520  045  116  045     P.AKV:     .ASCII  /%N%/
021523  101  011  052                .ASCII  /A/<11>/*/
021526  052  040  123                .ASCII  /* S/
021531  120  105  103                .ASCII  /PEC/
021534  111  101  114                .ASCII  /IAL/
021537  000                          .ASCII  <00>
021540  045  116  045     P.AKW:     .ASCII  /%N%/
021543  101  040  055                .ASCII  /A -/
021546  040  111  114                .ASCII  / IL/
021551  114  105  107                .ASCII  /LEG/
021554  101  114  040                .ASCII  /AL /
021557  125  116  111                .ASCII  /UNI/
021562  124  040  116                .ASCII  /T N/
021565  125  115  102                .ASCII  /UMB/
021570  105  122  000                .ASCII  /ER/<00>
021573  000                          .ASCII  <00>
021574  045  116  045     P.AKX:     .ASCII  /%N%/
021577  101  040  055                .ASCII  /A -/
021602  040  111  114                .ASCII  / IL/
```

ZRQAM1 RD/RX EXERCISER 21-Jul-1983 15:25:58 VAX-11 Bliss-16 V3-555
V01.2 PROTECTION TABLE 21-Jul-1983 15:19:20 DISK$USER2:[DIETZ.RELEASE]ZRQAB0.BL1;10 (35)

```
021605  114  105  107                .ASCII  /LEG/
021610  101  114  040                .ASCII  /AL /
021613  120  110  131                .ASCII  /PHY/
021616  123  111  103                .ASCII  /SIC/
021621  101  114  040                .ASCII  /AL /
021624  117  122  040                .ASCII  /OR /
021627  122  105  114                .ASCII  /REL/
021632  101  124  111                .ASCII  /ATI/
021635  126  105  040                .ASCII  /VE /
021640  102  114  117                .ASCII  /BLO/
021643  103  113  040                .ASCII  /CK /
021646  116  125  115                .ASCII  /NUM/
021651  102  105  122                .ASCII  /BER/
021654  000  000                     .ASCII  <00><00>
021656  045  116  045       P.AKY:   .ASCII  /%N%/
021661  101  040  055                .ASCII  /A -/
021664  040  104  105                .ASCII  / DE/
021667  126  111  103                .ASCII  /VIC/
021672  105  040  105                .ASCII  /E E/
021675  122  122  117                .ASCII  /RRO/
021700  122  000                     .ASCII  /R/<00>
021702  045  116  045       P.AKZ:   .ASCII  /%N%/
021705  101  040  055                .ASCII  /A -/
021710  040  132  105                .ASCII  / ZE/
021713  122  117  040                .ASCII  /RO /
021716  114  105  116                .ASCII  /LEN/
021721  107  110  124                .ASCII  /GHT/
021724  040  115  105                .ASCII  / ME/
021727  123  123  101                .ASCII  /SSA/
021732  107  105  000                .ASCII  /GE/<00>
021735  000                         .ASCII  <00>
021736  045  116  045       P.ALA:   .ASCII  /%N%/
021741  101  040  055                .ASCII  /A -/
021744  055  040  101                .ASCII  /- A/
021747  123  103  111                .ASCII  /SCI/
021752  111  040  111                .ASCII  /I I/
021755  116  106  117                .ASCII  /NFO/
021760  122  115  101                .ASCII  /RMA/
021763  124  111  117                .ASCII  /TIO/
021766  116  040  000                .ASCII  /N /<00>
021771  000                         .ASCII  <00>
021772  045  116  045       P.ALB:   .ASCII  /%N%/
021775  101  040  055                .ASCII  /A -/
022000  055  040  116                .ASCII  /- N/
022003  117  116  055                .ASCII  /ON-/
022006  101  103  123                .ASCII  /ACS/
022011  111  111  040                .ASCII  /II /
022014  111  116  106                .ASCII  /INF/
022017  117  122  115                .ASCII  /ORM/
022022  101  124  111                .ASCII  /ATI/
022025  117  116  000                .ASCII  /ON/<00>
022030  045  116  045       P.ALC:   .ASCII  /%N%/
022033  101  040  055                .ASCII  /A -/
```

ZRQAM1 RD/RX EXERCISER
V01.2 PROTECTION TABLE

21-Jul-1983 15:25:58 VAX-11 Bliss-16 V3-555
21-Jul-1983 15:19:20 DISK$USER2:[DIETZ.RELEASE]ZRQAB0.BL1;10 (35)

```
022036  055  040  124            .ASCII  /- T/
022041  105  122  115            .ASCII  /ERM/
022044  111  116  101            .ASCII  /INA/
022047  124  111  117            .ASCII  /TIO/
022052  116  040  115            .ASCII  /N M/
022055  105  123  123            .ASCII  /ESS/
022060  101  107  105            .ASCII  /AGE/
022063  000                      .ASCII  <00>
022064  045  116  045  P.ALD:    .ASCII  /%N%/
022067  101  040  055            .ASCII  /A -/
022072  055  040  123            .ASCII  /- S/
022075  125  103  103            .ASCII  /UCC/
022100  105  123  123            .ASCII  /ESS/
022103  057  106  101            .ASCII  <57>/FA/
022106  111  114  125            .ASCII  /ILU/
022111  122  105  040            .ASCII  /RE /
022114  103  117  104            .ASCII  /COD/
022117  105  000  000            .ASCII  /E/<00><00>
022122  045  116  045  P.ALE:    .ASCII  /%N%/
022125  101  040  055            .ASCII  /A -/
022130  055  040  123            .ASCII  /- S/
022133  105  116  104            .ASCII  /END/
022136  040  102  111            .ASCII  / BI/
022141  116  101  122            .ASCII  /NAR/
022144  131  040  104            .ASCII  /Y D/
022147  101  124  101            .ASCII  /ATA/
022152  000  000                 .ASCII  <00><00>
022154  045  116  045  P.ALF:    .ASCII  /%N%/
022157  101  040  055            .ASCII  /A -/
022162  055  040  123            .ASCII  /- S/
022165  105  116  104            .ASCII  /END/
022170  040  125  116            .ASCII  / UN/
022173  111  124  040            .ASCII  /IT /
022176  116  125  115            .ASCII  /NUM/
022201  102  105  122            .ASCII  /BER/
022204  054  040  122            .ASCII  /, R/
022207  105  114  101            .ASCII  /ELA/
022212  124  111  126            .ASCII  /TIV/
022215  105  040  104            .ASCII  /E D/
022220  102  116  000            .ASCII  /BN/<00>
022223  000                      .ASCII  <00>
022224  045  116  045  P.ALG:    .ASCII  /%N%/
022227  101  040  055            .ASCII  /A -/
022232  055  040  123            .ASCII  /- S/
022235  105  116  104            .ASCII  /END/
022240  040  125  116            .ASCII  / UN/
022243  111  124  040            .ASCII  /IT /
022246  116  125  115            .ASCII  /NUM/
022251  102  105  122            .ASCII  /BER/
022254  054  040  122            .ASCII  /, R/
022257  105  114  101            .ASCII  /ELA/
022262  124  111  126            .ASCII  /TIV/
022265  105  040  104            .ASCII  /E D/
```

```
022270  102     116     054             .ASCII  /BN,/
022273  040     127     122             .ASCII  / WR/
022276  111     124     105             .ASCII  /ITE/
022301  040     120     101             .ASCII  / PA/
022304  124     124     105             .ASCII  /TTE/
022307  122     116     000             .ASCII  /RN/<00>
022312  045     116     045     P.ALH:  .ASCII  /%N%/
022315  101     040     055             .ASCII  /A -/
022320  055     040     123             .ASCII  /- S/
022323  105     116     104             .ASCII  /END/
022326  040     125     116             .ASCII  / UN/
022331  111     124     040             .ASCII  /IT /
022334  116     125     115             .ASCII  /NUM/
022337  102     105     122             .ASCII  /BER/
022342  054     040     120             .ASCII  /, P/
022345  110     131     123             .ASCII  /HYS/
022350  111     103     101             .ASCII  /ICA/
022353  114     040     102             .ASCII  /L B/
022356  114     117     103             .ASCII  /LOC/
022361  113     040     116             .ASCII  /K N/
022364  125     115     102             .ASCII  /UMB/
022367  105     122     000             .ASCII  /ER/<00>
022372  045     116     045     P.ALI:  .ASCII  /%N%/
022375  101     040     055             .ASCII  /A -/
022400  055     040     123             .ASCII  /- S/
022403  105     116     104             .ASCII  /END/
022406  040     125     116             .ASCII  / UN/
022411  111     124     040             .ASCII  /IT /
022414  116     125     115             .ASCII  /NUM/
022417  102     105     122             .ASCII  /BER/
022422  054     040     114             .ASCII  /, L/
022425  117     107     111             .ASCII  /OGI/
022430  103     101     114             .ASCII  /CAL/
022433  040     040     102             .ASCII  /  B/
022436  114     117     103             .ASCII  /LOC/
022441  113     040     116             .ASCII  /K N/
022444  125     115     102             .ASCII  /UMB/
022447  105     122     040             .ASCII  /ER /
022452  000     000                     .ASCII  <00><00>
022454  045     101     103     P.ALK:  .ASCII  /%AC/
022457  117     116     124             .ASCII  /ONT/
022462  122     117     114             .ASCII  /ROL/
022465  114     105     122             .ASCII  /LER/
022470  040     124     111             .ASCII  / TI/
022473  115     105     117             .ASCII  /MEO/
022476  125     124     000             .ASCII  /UT/<00>
022501  000                             .ASCII  <00>
022502  045     101     105     P.ALL:  .ASCII  /%AE/
022505  116     126     105             .ASCII  /NVE/
022510  114     117     120             .ASCII  /LOP/
022513  105     057     120             .ASCII  /E/<57>/P/
022516  101     103     113             .ASCII  /AC%/
022521  105     124     040             .ASCII  /ET /
```

```
022524  122  105  101                    .ASCII  /REA/
022527  104  040  105                    .ASCII  /D E/
022532  122  122  117                    .ASCII  /RRO/
022535  122  040  050                    .ASCII  /R (/
022540  120  101  122                    .ASCII  /PAR/
022543  111  124  131                    .ASCII  /ITY/
022546  040  117  122                    .ASCII  / OR/
022551  040  124  111                    .ASCII  / TI/
022554  115  105  117                    .ASCII  /MEO/
022557  125  124  051                    .ASCII  /UT)/
022562  000  000                         .ASCII  <00><00>
022564  045  101  105          P.ALM:    .ASCII  /%AE/
022567  116  126  105                    .ASCII  /NVE/
022572  114  117  120                    .ASCII  /LOP/
022575  105  057  120                    .ASCII  /E/<57>/P/
022600  101  103  113                    .ASCII  /ACK/
022603  105  124  040                    .ASCII  /ET /
022606  127  122  111                    .ASCII  /WRI/
022611  124  105  040                    .ASCII  /TE /
022614  105  122  122                    .ASCII  /ERR/
022617  117  122  040                    .ASCII  /OR /
022622  050  120  101                    .ASCII  /(PA/
022625  122  111  124                    .ASCII  /RIT/
022630  131  040  117                    .ASCII  /Y O/
022633  122  040  124                    .ASCII  /R T/
022636  111  115  105                    .ASCII  /IME/
022641  117  125  124                    .ASCII  /OUT/
022644  051  000                         .ASCII  /)/<00>
022646  045  101  103          P.ALN:    .ASCII  /%AC/
022651  117  116  124                    .ASCII  /ONT/
022654  122  117  114                    .ASCII  /ROL/
022657  114  105  122                    .ASCII  /LER/
022662  040  122  117                    .ASCII  / RO/
022665  115  040  101                    .ASCII  /M A/
022670  116  104  040                    .ASCII  /ND /
022673  122  101  115                    .ASCII  /RAM/
022676  040  120  101                    .ASCII  / PA/
022701  122  111  124                    .ASCII  /RIT/
022704  131  040  105                    .ASCII  /Y E/
022707  122  122  117                    .ASCII  /RRO/
022712  122  000                         .ASCII  /R/<00>
022714  045  101  103          P.ALO:    .ASCII  /%AC/
022717  117  116  124                    .ASCII  /ONT/
022722  122  117  114                    .ASCII  /ROL/
022725  114  105  122                    .ASCII  /LER/
022730  040  122  101                    .ASCII  / RA/
022733  115  040  120                    .ASCII  /M P/
022736  101  122  111                    .ASCII  /ARI/
022741  124  131  040                    .ASCII  /TY /
022744  105  122  122                    .ASCII  /ERR/
022747  117  122  000                    .ASCII  /OR/<00>
022752  045  101  103          P.ALP:    .ASCII  /%AC/
022755  117  116  124                    .ASCII  /ONT/
```

```
022760  122  117  114            .ASCII  /ROL/
022763  114  105  122            .ASCII  /LER/
022766  040  122  117            .ASCII  / RO/
022771  115  040  120            .ASCII  /M P/
022774  101  122  111            .ASCII  /ARI/
022777  124  131  040            .ASCII  /TY /
023002  105  122  122            .ASCII  /ERR/
023005  117  122  000            .ASCII  /OR/<00>
023010  045  101  122  P.ALQ:    .ASCII  /%AR/
023013  111  116  107            .ASCII  /ING/
023016  040  122  105            .ASCII  / RE/
023021  101  104  040            .ASCII  /AD /
023024  105  122  122            .ASCII  /ERR/
023027  117  122  040            .ASCII  /OR /
023032  050  120  101            .ASCII  /(PA/
023035  122  111  124            .ASCII  /RIT/
023040  131  040  117            .ASCII  /Y O/
023043  122  040  124            .ASCII  /R T/
023046  111  115  105            .ASCII  /IME/
023051  117  125  124            .ASCII  /OUT/
023054  051  000                 .ASCII  /)/<00>
023056  045  101  122  P.ALR:    .ASCII  /%AR/
023061  111  116  107            .ASCII  /ING/
023064  040  127  122            .ASCII  / WR/
023067  111  124  105            .ASCII  /ITE/
023072  040  105  122            .ASCII  / ER/
023075  122  117  122            .ASCII  /ROR/
023100  040  050  120            .ASCII  / (P/
023103  101  122  111            .ASCII  /ARI/
023106  124  131  040            .ASCII  /TY /
023111  117  122  040            .ASCII  /OR /
023114  124  111  115            .ASCII  /TIM/
023117  105  117  125            .ASCII  /EOU/
023122  124  051  000            .ASCII  /T)/<00>
023125  000                      .ASCII  <00>
023126  111  116  124  P.ALS:    .ASCII  /INT/
023131  105  122  122            .ASCII  /ERR/
023134  125  120  124            .ASCII  /UPT/
023137  040  115  101            .ASCII  / MA/
023142  123  124  105            .ASCII  /STE/
023145  122  040  106            .ASCII  /R F/
023150  101  111  114            .ASCII  /AIL/
023153  125  122  105            .ASCII  /URE/
023156  000  000                 .ASCII  <00><00>
023160  045  101  110  P.ALT:    .ASCII  /%AH/
023163  117  123  124            .ASCII  /OST/
023166  040  101  103            .ASCII  / AC/
023171  103  105  123            .ASCII  /CES/
023174  123  040  124            .ASCII  /S T/
023177  111  115  105            .ASCII  /IME/
023202  117  125  124            .ASCII  /OUT/
023205  040  050  110            .ASCII  / (H/
023210  111  107  110            .ASCII  /IGH/
```

ZRQAM1 RD/RX EXERCISER
V01.2 PROTECTION TABLE

21-Jul-1983 15:25:58 VAX-11 Bliss-16 V3-555
21-Jul-1983 15:19:20 DISK$USER2:[DIETZ.RELEASE]ZRQAB0.BL1;10 (35)

```
023213  105  122  040                .ASCII  /ER /
023216  114  105  126                .ASCII  /LEV/
023221  105  114  040                .ASCII  /EL /
023224  120  122  117                .ASCII  /PRO/
023227  124  117  103                .ASCII  /TOC/
023232  117  114  040                .ASCII  /OL /
023235  104  105  120                .ASCII  /DEP/
023240  105  116  104                .ASCII  /END/
023243  105  116  124                .ASCII  /ENT/
023246  051  000                    .ASCII  /)/<00>
023250  045  101  103     P.ALU:     .ASCII  /%AC/
023253  122  105  104                .ASCII  /RED/
023256  111  124  040                .ASCII  /IT /
023261  114  111  115                .ASCII  /LIM/
023264  111  124  040                .ASCII  /IT /
023267  105  130  103                .ASCII  /EXC/
023272  105  105  104                .ASCII  /EED/
023275  105  104  000                .ASCII  /ED/<00>
023300  045  101  121     P.ALV:     .ASCII  /%AQ/
023303  055  102  125                .ASCII  /-BU/
023306  123  040  115                .ASCII  /S M/
023311  101  123  124                .ASCII  /AST/
023314  105  122  040                .ASCII  /ER /
023317  105  122  122                .ASCII  /ERR/
023322  117  122  000                .ASCII  /OR/<00>
023325  000                        .ASCII  <00>
023326  045  101  103     P.ALW:     .ASCII  /%AC/
023331  117  116  124                .ASCII  /ONT/
023334  122  117  114                .ASCII  /ROL/
023337  114  105  122                .ASCII  /LER/
023342  040  106  101                .ASCII  / FA/
023345  124  101  114                .ASCII  /TAL/
023350  040  105  122                .ASCII  / ER/
023353  122  117  122                .ASCII  /ROR/
023356  000  000                    .ASCII  <00><00>
023360  045  101  111     P.ALX:     .ASCII  /%AI/
023363  116  123  124                .ASCII  /NST/
023366  122  125  103                .ASCII  /RUC/
023371  124  111  117                .ASCII  /TIO/
023374  116  040  114                .ASCII  /N L/
023377  117  117  120                .ASCII  /OOP/
023402  040  124  111                .ASCII  / TI/
023405  115  105  117                .ASCII  /MEO/
023410  125  124  000                .ASCII  /UT/<00>
023413  000                        .ASCII  <00>
023414  045  101  111     P.ALY:     .ASCII  /%AI/
023417  114  114  105                .ASCII  /LLE/
023422  107  101  114                .ASCII  /GAL/
023425  040  126  111                .ASCII  / VI/
023430  122  124  125                .ASCII  /RTU/
023433  101  114  040                .ASCII  /AL /
023436  103  111  122                .ASCII  /CIR/
023441  103  125  111                .ASCII  /CUI/
```

```
023444  124  040  111                .ASCII  /T I/
023447  104  000  000                .ASCII  /D/<00><00>
023452  045  101  111        P.ALZ:  .ASCII  /%AI/
023455  116  124  105                .ASCII  /NTE/
023460  122  122  125                .ASCII  /RRU/
023463  120  124  040                .ASCII  /PT /
023466  126  105  103                .ASCII  /VEC/
023471  124  117  122                .ASCII  /TOR/
023474  040  111  114                .ASCII  / IL/
023477  114  105  107                .ASCII  /LEG/
023502  101  114  000                .ASCII  /AL/<00>
023505  000                        .ASCII  <00>
023506  045  101  115        P.AMA:  .ASCII  /%AM/
023511  101  111  116                .ASCII  /AIN/
023514  124  105  116                .ASCII  /TEN/
023517  101  116  103                .ASCII  /ANC/
023522  105  040  122                .ASCII  /E R/
023525  105  101  104                .ASCII  /EAD/
023530  057  127  122                .ASCII  <57>/WR/
023533  111  124  105                .ASCII  /ITE/
023536  040  111  116                .ASCII  / IN/
023541  126  101  114                .ASCII  /VAL/
023544  111  104  040                .ASCII  /ID /
023547  122  105  107                .ASCII  /REG/
023552  111  117  116                .ASCII  /ION/
023555  040  111  104                .ASCII  / ID/
023560  105  116  124                .ASCII  /ENT/
023563  111  106  111                .ASCII  /IFI/
023566  105  122  000                .ASCII  /ER/<00>
023571  000                        .ASCII  <00>
023572  045  101  115        P.AMB:  .ASCII  /%AM/
023575  101  111  116                .ASCII  /AIN/
023600  124  105  116                .ASCII  /TEN/
023603  101  116  103                .ASCII  /ANC/
023606  105  040  127                .ASCII  /E W/
023611  122  111  124                .ASCII  /RIT/
023614  105  040  114                .ASCII  /E L/
023617  117  101  104                .ASCII  /OAD/
023622  040  124  117                .ASCII  / TO/
023625  040  116  117                .ASCII  / NO/
023630  116  055  114                .ASCII  /N-L/
023633  117  101  104                .ASCII  /OAD/
023636  101  102  114                .ASCII  /ABL/
023641  105  040  103                .ASCII  /E C/
023644  117  116  124                .ASCII  /ONT/
023647  122  117  114                .ASCII  /ROL/
023652  114  105  122                .ASCII  /LER/
023655  000                        .ASCII  <00>
023656  045  101  103        P.AMC:  .ASCII  /%AC/
023661  117  116  124                .ASCII  /ONT/
023664  122  117  114                .ASCII  /ROL/
023667  114  105  122                .ASCII  /LER/
023672  040  122  101                .ASCII  / RA/
```

```
023675  115  040  105                .ASCII  /M E/
023700  122  122  117                .ASCII  /RRO/
023703  122  040  050                .ASCII  /R (/
023706  116  117  116                .ASCII  /NON/
023711  055  120  101                .ASCII  /-PA/
023714  122  111  124                .ASCII  /RIT/
023717  131  051  000                .ASCII  /Y)/<00>
023722  045  101  111  P.AMD:        .ASCII  /%AI/
023725  116  111  124                .ASCII  /NIT/
023730  040  123  105                .ASCII  / SE/
023733  121  125  105                .ASCII  /QUE/
023736  116  103  105                .ASCII  /NCE/
023741  040  105  122                .ASCII  / ER/
023744  122  117  122                .ASCII  /ROR/
023747  000                         .ASCII  <00>
023750  045  101  110  P.AME:        .ASCII  /%AH/
023753  111  107  110                .ASCII  /IGH/
023756  105  122  040                .ASCII  /ER /
023761  114  105  126                .ASCII  /LEV/
023764  105  114  040                .ASCII  /EL /
023767  120  122  117                .ASCII  /PRO/
023772  124  117  103                .ASCII  /TOC/
023775  117  114  040                .ASCII  /OL /
024000  111  116  103                .ASCII  /INC/
024003  117  115  120                .ASCII  /OMP/
024006  101  124  111                .ASCII  /ATI/
024011  102  111  114                .ASCII  /BIL/
024014  111  124  131                .ASCII  /ITY/
024017  040  105  122                .ASCII  / ER/
024022  122  117  122                .ASCII  /ROR/
024025  000                         .ASCII  <00>
024026  045  101  120  P.AMF:        .ASCII  /%AP/
024031  125  122  107                .ASCII  /URG/
024034  105  057  120                .ASCII  /E/<57>/P/
024037  117  114  114                .ASCII  /OLL/
024042  040  110  101                .ASCII  / HA/
024045  122  104  127                .ASCII  /RDW/
024050  101  122  105                .ASCII  /ARE/
024053  040  106  101                .ASCII  / FA/
024056  111  114  125                .ASCII  /ILU/
024061  122  105  000                .ASCII  /RE/<00>
024064  045  101  115  P.AMG:        .ASCII  /%AM/
024067  101  120  120                .ASCII  /APP/
024072  111  116  107                .ASCII  /ING/
024075  040  122  105                .ASCII  / RE/
024100  107  111  123                .ASCII  /GIS/
024103  124  105  122                .ASCII  /TER/
024106  040  122  105                .ASCII  / RE/
024111  101  104  040                .ASCII  /AD /
024114  106  101  111                .ASCII  /FAI/
024117  114  125  122                .ASCII  /LUR/
024122  105  040  050                .ASCII  /E (/
024125  120  101  122                .ASCII  /PAR/
```

```
024130      111     124     131                 .ASCII  /ITY/
024133      040     117     122                 .ASCII  / OR/
024136      040     124     111                 .ASCII  / TI/
024141      115     105     117                 .ASCII  /MEO/
024144      125     124     051                 .ASCII  /UT)/
024147      000                                 .ASCII  <00>
024150   022454'                        P.ALJ:  .WORD   P.ALK
024152   022502'                                .WORD   P.ALL
024154   022564'                                .WORD   P.ALM
024156   022646'                                .WORD   P.ALN
024160   022714'                                .WORD   P.ALO
024162   022752'                                .WORD   P.ALP
024164   023010'                                .WORD   P.ALQ
024166   023056'                                .WORD   P.ALR
024170   023126'                                .WORD   P.ALS
024172   023160'                                .WORD   P.ALT
024174   023250'                                .WORD   P.ALU
024176   023300'                                .WORD   P.ALV
024200   023326'                                .WORD   P.ALW
024202   023360'                                .WORD   P.ALX
024204   023414'                                .WORD   P.ALY
024206   023452'                                .WORD   P.ALZ
024210   023506'                                .WORD   P.AMA
024212   023572'                                .WORD   P.AMB
024214   023656'                                .WORD   P.AMC
024216   023722'                                .WORD   P.AMD
024220   023750'                                .WORD   P.AME
024222   024026'                                .WORD   P.AMF
024224   024064'                                .WORD   P.AMG
024226      045     101     124         P.AMI:  .ASCII  /%AT/
024231      061     061     040                 .ASCII  /11 /
024234      103     120     125                 .ASCII  /CPU/
024237      040     106     101                 .ASCII  / FA/
024242      111     114     125                 .ASCII  /ILU/
024245      122     105     000                 .ASCII  /RE/<00>
024250      045     101     116         P.AMJ:  .ASCII  /%AN/
024253      117     116     055                 .ASCII  /ON-/
024256      120     101     122                 .ASCII  /PAR/
024261      111     124     131                 .ASCII  /ITY/
024264      040     122     101                 .ASCII  / RA/
024267      115     040     105                 .ASCII  /M E/
024272      122     122     117                 .ASCII  /RRO/
024275      122     000     000                 .ASCII  /R/<00><00>
024300      045     101     123         P.AMK:  .ASCII  /%AS/
024303      124     101     124                 .ASCII  /TAT/
024306      105     040     115                 .ASCII  /E M/
024311      101     103     110                 .ASCII  /ACH/
024314      111     116     105                 .ASCII  /INE/
024317      040     106     101                 .ASCII  / FA/
024322      111     114     125                 .ASCII  /ILU/
024325      122     105     040                 .ASCII  /RE /
024330      055     040     124                 .ASCII  /- T/
024333      061     061     040                 .ASCII  /11 /
```

```
024336  101  104  104          .ASCII  /ADD/
024341  122  105  123          .ASCII  /RES/
024344  123  040  122          .ASCII  /S R/
024347  105  107  111          .ASCII  /EGI/
024352  123  124  105          .ASCII  /STE/
024355  122  000  000          .ASCII  /R/<00><00>
024360  045  101  123  P.AML:  .ASCII  /%AS/
024363  124  101  124          .ASCII  /TAT/
024366  105  040  115          .ASCII  /E M/
024371  101  103  110          .ASCII  /ACH/
024374  111  116  105          .ASCII  /INE/
024377  040  106  101          .ASCII  / FA/
024402  111  114  125          .ASCII  /ILU/
024405  122  105  040          .ASCII  /RE /
024410  055  040  121          .ASCII  /- Q/
024413  055  102  125          .ASCII  /-BU/
024416  123  040  101          .ASCII  /S A/
024421  104  104  122          .ASCII  /DDR/
024424  105  123  123          .ASCII  /ESS/
024427  040  122  105          .ASCII  / RE/
024432  107  111  123          .ASCII  /GIS/
024435  124  105  122          .ASCII  /TER/
024440  000  000               .ASCII  <00><00>
024442  045  101  123  P.AMM:  .ASCII  /%AS/
024445  124  101  124          .ASCII  /TAT/
024450  105  040  115          .ASCII  /E M/
024453  101  103  110          .ASCII  /ACH/
024456  111  116  105          .ASCII  /INE/
024461  040  106  101          .ASCII  / FA/
024464  111  114  125          .ASCII  /ILU/
024467  122  105  040          .ASCII  /RE /
024472  055  040  103          .ASCII  /- C/
024475  122  103  040          .ASCII  /RC /
024500  122  105  107          .ASCII  /REG/
024503  111  123  124          .ASCII  /IST/
024506  105  122  000          .ASCII  /ER/<00>
024511  000                    .ASCII  <00>
024512  045  101  123  P.AMN:  .ASCII  /%AS/
024515  124  101  124          .ASCII  /TAT/
024520  105  040  115          .ASCII  /E M/
024523  101  103  110          .ASCII  /ACH/
024526  111  116  105          .ASCII  /INE/
024531  040  106  101          .ASCII  / FA/
024534  111  114  125          .ASCII  /ILU/
024537  122  105  040          .ASCII  /RE /
024542  055  040  123          .ASCII  /- S/
024545  105  122  111          .ASCII  /ERI/
024550  101  114  111          .ASCII  /ALI/
024553  132  105  122          .ASCII  /ZER/
024556  057  104  105          .ASCII  <57>/DE/
024561  123  105  122          .ASCII  /SER/
024564  111  101  114          .ASCII  /IAL/
024567  111  132  105          .ASCII  /IZE/
```

```
024572  122  040  122                    .ASCII  /R R/
024575  105  107  111                    .ASCII  /EGI/
024600  123  124  105                    .ASCII  /STE/
024603  122  000  000                    .ASCII  /R/<00><00>
024606  045  101  123          P.AMO:    .ASCII  /%AS/
024611  124  101  124                    .ASCII  /TAT/
024614  105  040  115                    .ASCII  /E M/
024617  101  103  110                    .ASCII  /ACH/
024622  111  116  105                    .ASCII  /INE/
024625  040  106  101                    .ASCII  / FA/
024630  111  114  125                    .ASCII  /ILU/
024633  122  105  040                    .ASCII  /RE /
024636  055  040  127                    .ASCII  /- W/
024641  122  117  116                    .ASCII  /RON/
024644  107  040  110                    .ASCII  /G H/
024647  101  122  104                    .ASCII  /ARD/
024652  127  101  122                    .ASCII  /WAR/
024655  105  040  126                    .ASCII  /E V/
024660  105  122  123                    .ASCII  /ERS/
024663  111  117  116                    .ASCII  /ION/
024666  000  000                         .ASCII  <00><00>
024670  024226'                 P.AMH:   .WORD   P.AMI
024672  024250'                          .WORD   P.AMJ
024674  024300'                          .WORD   P.AMK
024676  024360'                          .WORD   P.AML
024700  024442'                          .WORD   P.AMM
024702  024512'                          .WORD   P.AMN
024704  024606'                          .WORD   P.AMO
024706  045  101  040          P.AMP:    .ASCII  /%A /
024711  045  117  066                    .ASCII  /%06/
024714  000  000                         .ASCII  <00><00>
024716  045  101  157          P.AMQ:    .ASCII  /%Ao/
024721  143  164  040                    .ASCII  /ct /
024724  045  117  064                    .ASCII  /%04/
024727  000                             .ASCII  <00>
024730  045  123  064          P.AMR:    .ASCII  /%S4/
024733  000                             .ASCII  <00>
024734  045  116  000          P.AMS:    .ASCII  /%N/<00>
024737  000                             .ASCII  <00>
024740  045  101  040          P.AMT:    .ASCII  /%A /
024743  055  040  000                    .ASCII  /- /<00>
024746  045  101  052          P.AMU:    .ASCII  /%A*/
024751  040  000  000                    .ASCII  / /<00><00>
024754  000000C -               L$HWLEN::
                                        .WORD   <<L$NDHW-L$HWLEN>/2>
024756  172150                  HWPT.IP.ADDR::
                                        .WORD   -5630
024760  000154                  HWPT.VECTOR::
                                        .WORD   154
024762  000004                  HWPT.BR.LEVEL::
                                        .WORD   4
024764  100034                  HWPT.DISK::
                                        .WORD   -77744
```

```
024766  000000                  HWPT.S.TRK::
                                        .WORD   0
024770  052137                  HWPT.E.TRK::
                                        .WORD   52137
024772                          L$NDHW::.BLKW   1
024774  000000C                 L$SWLEN::
                                        .WORD   <<L$NDSW-L$SWLEN>/2>
024776  000040                  SWP.ERROR::
                                        .WORD   40
025000  000024                  SWP.XFER::
                                        .WORD   24
025002  000202                  SWP.FLAGS::
                                        .WORD   202
025004  000000                  SWP.DPAT::
                                        .WORD   0
025006  000020                  SWP.UCNT::
                                        .WORD   20
025010  000143                  SWP.RAT::
                                        .WORD   143
025012  000022                  DUPROUND::
                                        .WORD   22
025014                          SWP.UDPAT::
                                        .BLKW   20
025054                          L$NDSW::.BLKW   1
025056  000000                  L$PROT::.WORD   0
025060  177777                          .WORD   -1
025062  000006                          .WORD   6


000000                                  .PSECT  $FFF$,  RO
000000                          CST::   .BLKW   27
000056                          CST.ADDR::
                                        .BLKW   1
000060                          DCT::   .BLKW   11
000102                          DCT.ADDR::
                                        .BLKW   1
000104                          RDRX.ADDR::
                                        .BLKW   1
000106                          IRDRX.ADDR::
                                        .BLKW   1
000110                          DUPPKT::.BLKW   401
001112                          TALLY:: .BLKW   160
001452                          T.ADDR::.BLKW   1
001454                          C.ERR.TBL::
                                        .BLKW   1
001456                          MSCP.PKT::
                                        .BLKW   630
003136                          IPKT.ADDR::
                                        .BLKW   1
003140                          PKT.USE::
                                        .BLKW   6
003154                          RETPKT::.BLKW   140
```

```
003454                          RP.USE::.BLKW   2
003460                          RP.INDX::
                                        .BLKW   1
003462                          RP.ADDR::
                                        .BLKW   1
003464                          ELOG.PKT::
                                        .BLKW   614
005114                          BUFF.ADDR::
                                        .BLKW   10
005134                          BUFF.OWN::
                                        .BLKW   4
005144                          IODQ::  .BLKW   2
005150                          IODQ.IN::
                                        .BLKW   1
005152                          IODQ.OUT::
                                        .BLKW   1
005154                          ENTRY.REASON::
                                        .BLKB   1
005155                          EOP.FLAG::
                                        .BLKB   1
005156                          DUP.FLAGS::
                                        .BLKW   1
005160                          CCTLR:: .BLKW   1
005162                          CDISK:: .BLKW   1
005164                          CUOFF:: .BLKW   1
005166                          CTLR.CNT::
                                        .BLKW   1
005170                          DUR::   .BLKW   2
005174                          QIO::   .BLKB   1
                                        .EVEN
005176                          FREE.MEM.ADDR::
                                        .BLKW   1
005200                          BYTS.PER.QIO::
                                        .BLKW   1
005202                          ST.CODE::
                                        .BLKW   1
005204                          SB.CODE::
                                        .BLKW   1
005206                          STEP::  .BLKW   1
005210                          OF.RC:: .BLKW   1
005212                          SA.REG::.BLKW   1
005214                          CMD.TIME::
                                        .BLKW   1
005216                          NEX::   .BLKW   1
005220                          CRN.LOW::
                                        .BLKW   1
005222                          CRN.HIGH::
                                        .BLKW   1
005224                          P.INDEX::
                                        .BLKW   1
005226                          S.DUPPKT::
                                        .BLKW   1
005230                          S.PATTERN::
```

ZRQAM1 RD/RX EXERCISER 21-Jul-1983 15:25:58 VAX-11 Bliss-16 V3-555
V01.2 PROTECTION TABLE 21-Jul-1983 15:19:20 DISK$USER2:[DIETZ.RELEASE]ZRQAB0.BL1;10 (35)

```
                                .BLKW   1
005232                  CREDIT.BAL::
                                .BLKW   1
005234                  NEXT.PKT.USE::
                                .BLKB   1

                                .GLOBL  L$SOFT, T$PTHV, L$RPT, L$INIT
                                .GLOBL  L$CLEAN, L$LAST, L$HARD, L$DVTYP
                                .GLOBL  L$DESC, L$DU, L$AU, L$AUTO, T1


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==              1000
000400                  BIT8==              400
000200                  BIT7==              200
000100                  BIT6==              100
000040                  BIT5==              40
000020                  BIT4==              20
000010                  BIT3==              10
000004                  BIT2==              4
000002                  BIT1==              2
000001                  BIT0==              1
000040                  EF.START==          40
000037                  EF.RESTART==        37
000036                  EF.CONTINUE==       36
000035                  EF.NEW==            35
000034                  EF.PWR==            34
000340                  PRI07==             340
000300                  PRI06==             300
000240                  PRI05==             240
000200                  PRI04==             200
000140                  PRI03==             140
000100                  PRI02==             100
000040                  PRI01==             40
000000                  PRI00==             0
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
000126'                 L$ERRTBL==      ERRTYP
024776'                 L$SW==          L$SWLEN+2
024756'                 L$HW==          L$HWLEN+2
000011'                 L$DEPO==        L$REV+1
000136'                 HWQ1==          P.AAA
000152'                 HWQ2==          P.AAB
000162'                 HWQ3==          P.AAC
000174'                 HWQ4==          P.AAD
000220'                 HWQ5==          P.AAE
000310'                 HWQ6==          P.AAF
000326'                 HWQ7==          P.AAG
000406'                 HWQ8==          P.AAH
000460'                 HWQ9==          P.AAI
000560'                 HWQ10==         P.AAJ
000646'                 HWQ11==         P.AAK
000700'                 SWQ1==          P.AAL
000722'                 SWQ2==          P.AAM
001004'                 SWQ4==          P.AAN
001026'                 SWQ7==          P.AAO
001100'                 SWQ9==          P.AAP
001154'                 SWQ10==         P.AAQ
001220'                 SWQ11==         P.AAR
001252'                 SWQ12==         P.AAS
001350'                 SWQ13==         P.AAT
001426'                 SWQ14==         P.AAU
001444'                 SWQ15==         P.AAV
001514'                 SWQ17==         P.AAW
001602'                 SWQ19==         P.AAX
001672'                 SWQ22==         P.AAY
001760'                 SWM1==          P.AAZ
002050'                 NULL==          P.ABA
002052'                 DBM5==          P.ABB
002100'                 DBM12==         P.ABC
002156'                 DBM18==         P.ABD
002230'                 DBM19==         P.ABE
002314'                 DBM20==         P.ABF
002372'                 DBM21==         P.ABG
002454'                 DBM23==         P.ABH
002512'                 DBM25==         P.ABI
002556'                 DBM26==         P.ABJ
002610'                 DBM29==         P.ABK
```

```
002656'  DBM107==  P.ABL
002714'  DBM108==  P.ABM
002772'  DBM109==  P.ABN
003052'  DBM110==  P.ABO
003112'  DBM112==  P.ABP
003160'  DU.MSG==  P.ABQ
003724'  DU.RSN==  P.ABR
003754'  MSG.01==  P.ACE
004006'  MSG.02==  P.ACF
004042'  MSG.03==  P.ACG
004074'  RPT1==    P.ACH
004160'  RPT2==    P.ACI
004224'  RPT3==    P.ACJ
004310'  RPT4==    P.ACK
004354'  RPT5==    P.ACL
004442'  RPT6==    P.ACM
004506'  RPT7==    P.ACN
004530'  RPT8==    P.ACO
004552'  RPT9==    P.ACP
004600'  RPT10==   P.ACQ
004632'  RPT11==   P.ACR
004720'  RPT12==   P.ACS
004766'  RPT13==   P.ACT
005066'  RPT14==   P.ACU
005164'  RPT15==   P.ACV
005264'  RPT16==   P.ACW
005346'  EGS.01==  P.ACX
005366'  EGS.02==  P.ACY
005460'  EGD.10==  P.ACZ
005520'  EGD.11==  P.ADA
005544'  EGD.12==  P.ADB
005572'  EGD.13==  P.ADC
005620'  EGD.14==  P.ADD
005650'  EGD.15==  P.ADE
005666'  EGD.16==  P.ADF
005716'  EGD.17==  P.ADG
005734'  EGD.18==  P.ADH
005754'  EGD.20==  P.ADI
006042'  EGD.21==  P.ADJ
006154'  EGD.22==  P.ADK
006214'  EGD.23==  P.ADL
006256'  EGH.30==  P.ADM
006302'  EBS.01==  P.ADN
006344'  EBD.10==  P.ADO
006404'  EBD.12==  P.ADP
006452'  EBD.13==  P.ADQ
006504'  EBD.14==  P.ADR
006544'  EBD.18==  P.ADS
006600'  EBD.19==  P.ADT
006660'  EH.0==    P.ADU
006716'  EH.1==    P.ADV
006754'  EH.2==    P.ADW
007014'  EH.3==    P.ADX
```

```
007052'  EH.4==       P.ADY
007076'  EH.5==       P.ADZ
007126'  EH.6==       P.AEA
007160'  EH.7==       P.AEB
007212'  EH.8==       P.AEC
007246'  EH.9==       P.AED
007300'  EH.10==      P.AEE
007332'  EH.12==      P.AEF
007370'  EH.13==      P.AEG
010120'  ERR.COD==    P.AEH
010154'  ELG.00==     P.AEW
010510'  ELG.FMT==    P.AEX
010522'  EX.BDR==     P.AFD
010612'  EX.BDW==     P.AFE
010700'  EX.LBR==     P.AFF
010744'  EX.LBW==     P.AFG
011010'  EX.RBN==     P.AFH
011066'  EX.CBR==     P.AFI
011136'  EX.CBW==     P.AFJ
011206'  XX13==       P.AFK
011230'  XX14==       P.AFL
011244'  XX15==       P.AFM
011266'  XX16==       P.AFN
011314'  XX17==       P.AFO
011332'  XX18==       P.AFP
011342'  XX19==       P.AFQ
011354'  XX20==       P.AFR
011370'  XX21==       P.AFS
011430'  XX22==       P.AFT
011464'  XX23==       P.AFU
011520'  XX24==       P.AFV
011560'  XX25==       P.AFW
011630'  XX26==       P.AFX
011704'  XX27==       P.AFY
011746'  XX29==       P.AFZ
011772'  XX30==       P.AGA
012020'  XX31==       P.AGB
012052'  XX32==       P.AGC
012100'  XX33==       P.AGD
012136'  XX34==       P.AGE
012206'  XX35==       P.AGF
012224'  XX37==       P.AGG
012250'  XX38==       P.AGH
012314'  XX39==       P.AGI
012342'  XX40==       P.AGJ
012362'  XX41==       P.AGK
012412'  EB.DCT==     P.AGL
012466'  EB.COMM==    P.AGM
012550'  EB.PKT==     P.AGN
012612'  EB.RAL==     P.AGO
012652'  EB.ADDR==    P.AGP
012714'  EBNEX1==     P.AGQ
012770'  EB.NEX2==    P.AGR
```

```
013066'  EBNEX3==  P.AGS
013162'  CER.01==  P.AGT
013226'  CER.02==  P.AGU
013302'  EX.SEQ==  P.AGV
013322'  EX.CRD==  P.AGW
013352'  EX.MTN==  P.AGX
013372'  EX.DGM==  P.AGY
013410'  EX.RD==   P.AGZ
013420'  EX.WRT==  P.AHA
013430'  EX.ACC==  P.AHB
013442'  EX.ONL==  P.AHC
013454'  EX.SCC==  P.AHD
013500'  EX.GDS==  P.AHE
013522'  EX.ESP==  P.AHF
013552'  EX.ELP==  P.AHG
013576'  EX.SDD==  P.AHH
013612'  EX.RCD==  P.AHI
013632'  EX.ABP==  P.AHJ
013642'  SC.SDI==  P.AHK
013666'  SC.CON==  P.AHL
013710'  SC.DUP==  P.AHM
013740'  SC.ONL==  P.AHN
013762'  SC.SON==  P.AHO
014002'  SC.UNK==  P.AHP
014062'  SC.VOL==  P.AHQ
014142'  SC.IOP==  P.AHR
014210'  SC.DIS==  P.AHS
014302'  SC.FER==  P.AHT
014370'  SC.FE2==  P.AHU
014446'  SC.ISH==  P.AHV
014526'  SC.IS2==  P.AHW
014606'  SC.DST==  P.AHX
014662'  SC.DS2==  P.AHY
014734'  SC.ECC==  P.AHZ
015016'  SC.ECD==  P.AIA
015050'  SC.RCT==  P.AIB
015070'  SC.FUL==  P.AIC
015144'  SC.576==  P.AID
015220'  SC.FCT==  P.AIE
015266'  SC.EC1==  P.AIF
015316'  SC.EC2==  P.AIG
015346'  SC.EC3==  P.AIH
015400'  SC.EC4==  P.AII
015430'  SC.EC5==  P.AIJ
015460'  SC.EC6==  P.AIK
015510'  SC.EC7==  P.AIL
015542'  SC.EC8==  P.AIM
015574'  SC.EC9==  P.AIN
015636'  SC.SWP==  P.AIO
015676'  SC.HWP==  P.AIP
015736'  SC.ODA==  P.AIQ
015766'  SC.ODB==  P.AIR
016010'  SC.NXM==  P.AIS
```

```
016044'  SC.PAR==  P.AIT
016100'  SC.CTO==  P.AIU
016152'  SC.SDS==  P.AIV
016230'  SC.EDC==  P.AIW
016244'  SC.IDS==  P.AIX
016314'  SC.SRT==  P.AIY
016406'  SC.SRI==  P.AIZ
016474'  SC.POE==  P.AJA
016530'  SC.RDY==  P.AJB
016612'  SC.CLK==  P.AJC
016640'  SC.RSP==  P.AJD
016706'  SC.SUR==  P.AJE
016736'  SC.PSP==  P.AJF
017020'  F.1==     P.AJG
017054'  F.2==     P.AJH
017110'  F.3==     P.AJI
017142'  F.4==     P.AJJ
017202'  F.5==     P.AJK
017260'  F.6==     P.AJL
017334'  F.7==     P.AJM
017410'  F.8==     P.AJN
017466'  F.9==     P.AJO
017506'  F.10==    P.AJP
017536'  F.11==    P.AJQ
017562'  F.12==    P.AJR
017610'  F.13==    P.AJS
017666'  F.14==    P.AJT
017726'  F.15==    P.AJU
017754'  F.16==    P.AJV
020020'  F.17==    P.AJW
020064'  F.18==    P.AJX
020124'  F.19==    P.AJY
020164'  F.20==    P.AJZ
020236'  F.21==    P.AKA
020266'  EBH.30==  P.AKB
020304'  EBH.44==  P.AKC
020406'  EBH.45==  P.AKD
020440'  EBH.46==  P.AKE
020470'  EBH.47==  P.AKF
020552'  EBH.48==  P.AKG
020640'  EBH.49==  P.AKH
020752'  DF.0==    P.AKI
021006'  DF.1==    P.AKJ
021042'  DF.2==    P.AKK
021116'  DF.3==    P.AKL
021156'  DF.4==    P.AKM
021204'  DF.5==    P.AKN
021230'  DF.6==    P.AKO
021300'  DF.7==    P.AKP
021350'  T.QUE==   P.AKQ
021372'  T.DEF==   P.AKR
021424'  T.INF==   P.AKS
021450'  T.TER==   P.AKT
```

```
021474'                 T.FAT==         P.AKU
021520'                 T.SPL==         P.AKV
021540'                 E.UNT==         P.AKW
021574'                 E.BLK==         P.AKX
021656'                 E.DEV==         P.AKY
021702'                 E.ZER==         P.AKZ
021736'                 M.ASC==         P.ALA
021772'                 M.BIN==         P.ALB
022030'                 M.TER==         P.ALC
022064'                 M.COD==         P.ALD
022122'                 M.DAT==         P.ALE
022154'                 M.UR==          P.ALF
022224'                 M.URP==         P.ALG
022312'                 M.UP==          P.ALH
022372'                 M.UL==          P.ALI
024150'                 CNTR.ERR==      P.ALJ
024670'                 RDRX.ERR==      P.AMH
024706'                 EX.WRD==        P.AMP
024716'                 EX.OP==         P.AMQ
024730'                 SPACE4==        P.AMR
024734'                 CRLF==          P.AMS
024740'                 DASH==          P.AMT
024746'                 ASTERISK==      P.AMU
024756'                 DFPTBL==        L$HWLEN+2
024776'                 SFPTBL==        L$SWLEN+2

                        .EVEN

;                   PSECT SUMMARY
;
;   Psect Name                  Words     Attributes
;    $CODE$                      5402       RO ,  I  ,  LCL,  REL,  CON
;    $FFF$                       1358       RO ,  I  ,  LCL,  REL,  CON
;
;
;                   LIBRARY STATISTICS
;
;                                 -------- Symbols --------     Blocks
;    File                         Total    Loaded   Percent      Read
;
; DISK$USER2:[DIETZ.RELEASE]ZRQAB0.L16;27
;                                   455       227        49        60
;                   COMMAND QUALIFIERS
```

```
;        3415  module ZRQAM2 (
;        3416
;        3417  %title 'RD/RX EXERCISER'
;        3418                  ident = 'V01.2',
;        3419                  addressing_mode (absolute),
;        3420                  environment (noeis)
;        3421                  ) =
;        3422
;        3423  begin
;        3424
;        3425  %sbttl 'DECLARATIONS'
;        3426
;        3427  library 'ZRQAB0.L16';                          ! RDRX EXERCISER GLOBAL LIBRARY
;        3428
;        3429  require 'BLSMAC.REQ';                          ! DIAGNOSTIC SUPERVISOR LIBRARY
;        4920
;        4921  forward routine
;        4922      NEX_TRAP : L$ISR novalue,
;        4923      EMS_01 : novalue,
;        4924      EMS_DUP : novalue,
;        4925      EMS_BLK : novalue,
;        4926      EMSCMD : NOVALUE,
;        4927      SET_CPAR : novalue,
;        4928      SET_UPAR : novalue,
;        4929      EMS_DBN : novalue;
;        4930  external
;        4931      CST : blockvector [MAX_CTLR, CST_LEN, word] field (CST_FIELDS),
;        4932                          ! RUN-TIME CONTROLLER STATUS TABLES
;        4933      CST_ADDR : ref block [CST_LEN, word] field (CST_FIELDS),
;        4934                          ! CONTROLLER STATUS TABLE ADDRESS OF "CURRENT" CONTROLLER
;        4935      DCT : blockvector [MAX_CTLR, DCT_LEN, word] field (DCT_FIELDS),
;        4936                          ! DRIVER CONTROLLER TABLES
;        4937      DCT_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS),
;        4938                          ! ADDRESS OF "CURRENT" DRIVER CONTROLLER TABLE
;        4939      RDRX_ADDR : ref rdrx field (RC_REG),
;        4940                          ! DEVICE ADDRESS OF "CURRENT" CONTROLLER
;        4941      IRDRX_ADDR : ref rdrx field (RC_REG),
;        4942                          ! DEVICE ADDRESS OF INTERRUPTING CONTROLLER
;        4943      DUPPKT : BLOCK [257, WORD] field (DP_FIELDS),
;        4944                          ! BUFFER CONTAINING DUP INFORMATION FROM RECEIVE AND SEND COMMANDS
;        4945      TALLY : vector [MAX_UNITS * TALLY_LEN, word] field (T_FIELDS),
;        4946                          ! STATISTICS TABLES
;        4947      T_ADDR : ref block [TALLY_LEN, word] field (T_FIELDS),
;        4948                          ! ADDRESS OF STATISTICS TABLE (TALLY) FOR CURRENT UNIT
;        4949      C_ERR_TBL : blockvector [MAX_CTLR, C_ERR_LEN, word] field (C_ERR_FIELDS),
;        4950                          ! STATISTICS TABLE FOR CONTROLLER ERRORS
;        4951      MSCP_PKT : blockvector [PKT_CNT, PKT_LEN, word] field (PKT_FIELDS),
;        4952                          ! MSCP PACKET POOL
;        4953      IPKT_ADDR : ref block [PKT_LEN, word] field (PKT_FIELDS),
;        4954                          ! ADDRESS OF AN MSCP PACKET (INTERRUPT PROCESSING)
;        4955      PKT_USE : vector [PKT_CNT, byte, signed],
;        4956                          ! MSCP PACKET POOL ALLOCATION TABLE
;        4957      RETPKT : blockvector [RP_CNT, RP_LEN, word] field (RP_FIELDS),
```

```
;        4958                           ! RETURN PACKET POOL
;        4959        RP_USE : vector [RP_CNT, byte, signed],
;        4960                           ! RETURN PACKET POOL ALLOCATION TABLE
;        4961        RP_INDX : word,    ! CURRENT RETURN PACKET INDEX
;        4962        RP_ADDR : ref block [RP_LEN, word] field (RP_FIELDS),
;        4963                           ! CURRENT RETURN PACKET ADDRESS
;        4964        ELOG_PKT : blockvector [EP_CNT, EP_LEN, word] field (EP_FIELDS),
;        4965                           ! ERROR-LOG PACKET SAVE AREA
;        4966        BUFF_ADDR : vector [MAX_BUF_CNT],                   ! TABLE OF I/O BUFFER DESCRIPTORS
;        4967        BUFF_OWN : vector [MAX_BUF_CNT, byte, signed],      ! I/O BUFFER OWNERSHIP (CONTROLLER NUMBER)
;        4968        IODQ : vector [IODQ_LEN, byte],
;        4969                                        ! I/O DONE QUEUE - CIRCULAR QUEUE OF RETPKT INDECES
;        4970        IODQ_IN : word,                 ! I/O DONE QUEUE IN POINTER
;        4971        IODQ_OUT : word,                ! I/O DONE QUEUE OUT POINTER
;        4972        ENTRY_REASON : byte,            ! CURRENT OPERATOR COMMAND
;        4973        EOP_FLAG : byte,                ! END-OF-PASS FLAG
;        4974        DUP_FLAGS : WORD,               ! DUP FLAGS
;        4975        CCTLR : word,                   ! NUMBER OF "CURRENT" CONTROLLER
;        4976        CDISK : word,                   ! CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;        4977        CUOFF : word,                   ! CURRENT UNIT CST OFFSET
;        4978        CTLR_CNT : word,                ! TOTAL NUMBER OF CONFIGURED CONTROLLERS
;        4979        DUR : vector [MAX_UNITS, byte], ! DROP UNIT REASON
;        4980        QIO : vector [MAX_CTLR, byte],  ! NUMBER OF OUTSTANDING QIOS PER CONTROLLER
;        4981        FREE_MEM_ADDR,                  ! START OF FREE MEMORY
;        4982        BYTS_PER_QIO : word,            ! SIZE (BYTES) OF AN I/O BUFFER
;        4983        ST_CODE : word,                 ! CURRENT STATUS CODE
;        4984        SB_CODE : word,                 ! CURRENT SUB-CODE
;        4985        STEP : word,                    ! CURRENT STEP IN HARD_INIT
;        4986        OF_RC : signed word,            ! OFFSET (0 OR 2) TO READ IP OR SA
;        4987        SA_REG : word,                  ! STORAGE FOR SA REGISTER READS AND WRITES
;        4988        CMD_TIME : word,                ! COMMAND TIMEOUT VALUE (IN SECONDS)
;        4989        NEX : word,                     ! NON-EXISTENT MEMORY TRAP INDICATOR
;        4990        CRN_LOW : word,                 ! COMMAND REF NUMBER OF LAST COMMAND SENT
;        4991        CRN_HIGH : word,                ! COMMAND REF NUMBER (HI ORDER)
;        4992        P_INDEX : signed word,      ! CURRENT message PACKET INDEX
;        4993        S_DUPPKT : WORD,            ! DBN BYTE COUNTER
;        4994        S_PATTERN : WORD,           ! THE PATTERN WRITTEN TO DBN'S
;        4995        CREDIT_BAL : word,              ! CREDIT BALANCE
;        4996        NEXT_PKT_USE : byte,            ! POINTER TO NEXT ENTRY IN PKT_USE TABLE
;        4997        DBM5,
;        4998        DBM107,
;        4999        DU_MSG,
;        5000        DU_RSN : vector [12],
;        5001        ERR_COD : vector [14],
;        5002        ELG_FMT : vector [5],
;        5003    !
;        5004        HWQ1,
;        5005        HWQ2,
;        5006        HWQ3,
;        5007        HWQ4,
;        5008        HWQ5,
;        5009        HWQ6,
;        5010        HWQ7,
```

```
;          5011          HWQ8,
;          5012          HWQ9,
;          5013          HWQ10,
;          5014          HWQ11,
;          5015          SWQ1,
;          5016          SWQ2,
;          5017          SWQ4,
;          5018          SWQ7,
;          5019          SWQ9,
;          5020          SWQ10,
;          5021          SWQ11,
;          5022          SWQ12,
;          5023          SWQ13,
;          5024          SWQ14,
;          5025          SWQ15,
;          5026          SWQ17,
;          5027          SWQ19,
;          5028          SWQ22,
;          5029
;          5030          SWM1,
;          5031          NULL,
;          5032          MSG_01,
;          5033          MSG_02,
;          5034          MSG_03,
;          5035          RPT1,
;          5036          RPT2,
;          5037          RPT3,
;          5038          RPT4,
;          5039          RPT5,
;          5040          RPT6,
;          5041          RPT7,
;          5042          RPT8,
;          5043          RPT9,
;          5044          RPT10,
;          5045          RPT11,
;          5046          RPT12,
;          5047          RPT13,
;          5048          RPT14,
;          5049          RPT15,
;          5050          RPT16,
;          5051          EGS_01,
;          5052          EGS_02,
;          5053          EGD_10,
;          5054          EGD_11,
;          5055          EGD_12,
;          5056          EGD_13,
;          5057          EGD_14,
;          5058          EGD_15,
;          5059          EGD_16,
;          5060          EGD_17,
;          5061          EGD_18,
;          5062          EGD_20,
;          5063          EGD_21,
```

```
;          5064      EGD_22,
;          5065      EGD_23,
;          5066      EGH_30,
;          5067      EH_0,
;          5068      EH_1,
;          5069      EH_2,
;          5070      EH_3,
;          5071      EH_4,
;          5072      EH_5,
;          5073      EH_6,
;          5074      EH_7,
;          5075      EH_8,
;          5076      EH_9,
;          5077      EH_10,
;          5078      EH_12,
;          5079      EH_13,
;          5080
;          5081      EBS_01,
;          5082      EBD_10,
;          5083      EBD_12,
;          5084      EBD_13,
;          5085      EBD_14,
;          5086      EBD_18,
;          5087      EBD_19,
;          5088          ELG_00,
;          5089          EX_BDR,
;          5090      EX_BDW,
;          5091      EX_LBR,
;          5092      EX_LBW,
;          5093      EX_RBN,
;          5094      EX_CBR,
;          5095      EX_CBW,
;          5096  XX13,
;          5097  XX14,
;          5098  XX15,
;          5099  XX16,
;          5100  XX17,
;          5101  XX18,
;          5102  XX19,
;          5103  XX20,
;          5104  XX21,
;          5105  XX22,
;          5106  XX23,
;          5107  XX24,
;          5108  XX25,
;          5109  XX26,
;          5110  XX27,
;          5111  XX29,
;          5112  XX30,
;          5113  XX31,
;          5114  XX32,
;          5115  XX33,
;          5116  XX34,
```

```
;          5117  XX35,
;          5118  !XX36,
;          5119  XX37,
;          5120  XX38,
;          5121  XX39,
;          5122  XX40,
;          5123  XX41,
;          5124  EB_DCT,
;          5125  EB_COMM,
;          5126  EB_PKT,
;          5127  EB_RAL,
;          5128  EB_ADDR,
;          5129  EB_NEX1,
;          5130  EB_NEX2,
;          5131  EB_NEX3,
;          5132  CER_01,
;          5133  CER_02,
;          5134      EX_SEQ,
;          5135      EX_CRD,
;          5136      EX_MTN,
;          5137      EX_DGM,
;          5138      EX_RD,
;          5139      EX_WRT,
;          5140      EX_ACC,
;          5141      EX_ONL,
;          5142      EX_SCC,
;          5143      EX_GDS,
;          5144      EX_ESP,
;          5145      EX_ELP,
;          5146      EX_SDD,
;          5147      EX_RCD,
;          5148      EX_ABP,
;          5149      SC_SDI,
;          5150      SC_CON,
;          5151      SC_DUP,
;          5152      SC_ONL,
;          5153      SC_SON,
;          5154      SC_UNK,
;          5155      SC_VOL,
;          5156      SC_IOP,
;          5157      SC_DIS,
;          5158      SC_FER,
;          5159      SC_FE2,
;          5160      SC_ISH,
;          5161      SC_IS2,
;          5162      SC_DST,
;          5163      SC_DS2,
;          5164      SC_ECC,
;          5165      SC_ECD,
;          5166      SC_RCT,
;          5167      SC_FUL,
;          5168      SC_576,
;          5169      SC_FCT,
```

```
;          5170      SC_EC1,
;          5171      SC_EC2,
;          5172      SC_EC3,
;          5173      SC_EC4,
;          5174      SC_EC5,
;          5175      SC_EC6,
;          5176      SC_EC7,
;          5177      SC_EC8,
;          5178      SC_EC9,
;          5179      SC_SWP,
;          5180      SC_HWP,
;          5181      SC_ODA,
;          5182      SC_ODB,
;          5183      SC_NXM,
;          5184      SC_PAR,
;          5185      SC_CTO,
;          5186      SC_SDS,
;          5187      SC_EDC,
;          5188      SC_IDS,
;          5189      SC_SRT,
;          5190      SC_SRI,
;          5191      SC_POE,
;          5192      SC_RDY,
;          5193      SC_CLK,
;          5194      SC_RSP,
;          5195      SC_SUR,
;          5196      SC_PSP,
;          5197  F_1,
;          5198  F_2,
;          5199  F_3,
;          5200  F_4,
;          5201  F_5,
;          5202  F_6,
;          5203  F_7,
;          5204  F_8,
;          5205  F_9,
;          5206  F_10,
;          5207  F_11,
;          5208  F_12,
;          5209  F_13,
;          5210  F_14,
;          5211  F_15,
;          5212  F_16,
;          5213  F_17,
;          5214  F_18,
;          5215  F_19,
;          5216  F_20,
;          5217  F_21,
;          5218      EBH_30,
;          5219      EBH_44,
;          5220      EBH_45,
;          5221      EBH_46,
;          5222      EBH_47,
```

```
;          5223      EBH_48,
;          5224      EBH_49,
;          5225      df_0,
;          5226      df_1,
;          5227      df_2,
;          5228      df_3,
;          5229      df_4,
;          5230      df_5,
;          5231      df_6,
;          5232      df_7,
;          5233      T_QUE,
;          5234      T_DEF,
;          5235      T_INF,
;          5236      T_TER,
;          5237      T_FAT,
;          5238      T_SPL,
;          5239      E_UNT,
;          5240      E_BLK,
;          5241      E_DEV,
;          5242      E_ZER,
;          5243      M_ASC,
;          5244      M_BIN,
;          5245      M_TER,
;          5246      M_COD,
;          5247      M_DAT,
;          5248      M_UR ,
;          5249      M_URP,
;          5250      M_UP,
;          5251      M_UL,
;          5252      CNTR_ERR : vector [23],
;          5253      RDRX_ERR : vector [7],
;          5254      EX_WRD,
;          5255      EX_OP,
;          5256      SPACE4,
;          5257      CRLF,
;          5258      DASH,
;          5259      ASTERISK,
;          5260      SWP_FLAGS : word,
;          5261      L$HIMEM,
;          5262      L$LUN,
;          5263      L$UNIT;
;          5264
;          5265
;          5266  own
;          5267      TBL_SUC : vector [17] initial (NULL, SC_SDI, SC_CON, NULL, SC_DUP, NULL, NULL,
;          5268                NULL, SC_ONL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, SC_SON),
;          5269      TBL_OFL : vector [9] initial (SC_UNK, SC_VOL, SC_IOP, NULL, SC_DUP, NULL, NULL,
;          5270                NULL, SC_DIS),
;          5271      TBL_MFE : vector [11] initial (SC_FER, NULL, SC_ISH, SC_DST, SC_EC9, SC_576,
;          5272                SC_FCT, SC_ECC, SC_RCT, SC_FUL, SC_ECT),
;          5273      TBL_WPT : vector [3] initial (NULL, SC_SWP, SC_HWP),
;          5274      TBL_DAT : vector [16] initial (SC_FE2, NULL, SC_IS2, SC_DS2, SC_EC9, NULL, NULL,
;          5275                SC_ECD, SC_EC1, SC_EC2, SC_EC3, SC_EC4, SC_EC5, SC_EC6, SC_EC7, SC_EC8),
```

```
;         5276      TBL_HST : vector [5] initial (NULL, SC_ODA, SC_ODB, SC_NXM, SC_PAR),
;         5277      TBL_CNT : vector [4] initial (SC_CTO, SC_SDS, SC_EDC, SC_IDS),
;         5278      TBL_DRV : vector [9] initial (NULL, SC_SRT, SC_SRI, SC_POE, SC_RDY, SC_CLK, SC_RSP,
;         5279                SC_SUR, SC_PSP);
```

```
;         5280  %sbttl 'TYPE AND DESCRIPTION'
;         5281
;         5282  EQUALS;
;         5283
;         5284  DEVTYP (%asciz'RQDX1');                ! NAME OF DEVICE SUPPORTED BY PROGRAM
;         5285  DESCRIPT (%asciz'RD/RX EXERCISER');    ! TEST DESCRIPTION
```

```
;          5286  %sbttl 'HARDWARE PARAMETER CODING SECTION'
;          5287
;          5288  !+
;          5289  ! THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
;          5290  ! THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
;          5291  ! MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
;          5292  ! INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
;          5293  ! MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
;          5294  ! WITH THE OPERATOR.
;          5295  !-
;          5296
;          5297  BGNHRD;
;          5298
;          5299  GPRMA (HWQ1, 0, 0, %o'160000', %o'177777', YES, 1);                ! IP ADDRESS
;          5300  GPRMA (HWQ2, 2, 0, %o'4', %o'774', YES, 1);                        ! VECTOR
;          5301  GPRMD (HWQ3, 4, 0, %o'377', %o'0', %o'7', YES, 1);                 ! BR LEVEL
;          5302  GPRMD (HWQ4, 6, D, %o'3', %decimal'0', %decimal'3', YES, 1);       ! RDRX DRIVE NUMBER
;          5303  GPRML (HWQ5, 6, %o'4', YES, 1);                                    ! UNIT TYPE
;          5304  XFERF (HW1);
;          5305  GPRML (HWQ10, 6, %o'000010', YES, 1);                             ! run dup exerciser
;          5306  XFERF (NODU);
;          5307  GPRML (HWQ11, 6, %o'000020', YES, 1);                             ! WRITE TO DBN'S
;          5308  $L (NODU);
;          5309  GPRMD (HWQ6, 8, D, %o'177777', %decimal'0', RD_MAX_LBN, YES, 1);  ! STARTING LBN
;          5310  GPRMD (HWQ7, 10, D, %o'177777', GP$ATLO (8), RD_MAX_LBN, YES, 1); ! ENDING LBN
;          5311  XFER (HW2);
;          5312  $L (HW1);
;          5313  GPRMD (HWQ6, 8, D, %o'177777', %decimal'0', RX_MAX_LBN, YES, 1);  ! STARTING LBN
;          5314  GPRMD (HWQ7, 10, D, %o'177777', GP$ATLO (8), RX_MAX_LBN, YES, 1); ! ENDING LBN
;          5315  $L (HW2);
;          5316  GPRML (HWQ8, 6, %o'100000', NO, 1);                               ! EXER ON CUST DATA AREA
;          5317  XFERF (HWDONE);                                                   ! NO - DONE
;          5318  GPRML (HWQ9, 6, %o'100000', NO, 1);                               ! ** WARNING / CONFIRM
;          5319  $L (HWDONE);
;          5320
;          5321  ENDHRD;
```

```
;       5322  %sbttl 'SOFTWARE PARAMETER CODING SECTION'
;       5323
;       5324  !+
;       5325  ! THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
;       5326  ! THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
;       5327  ! MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
;       5328  ! INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
;       5329  ! MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
;       5330  ! WITH THE OPERATOR.
;       5331  !-
;       5332
;       5333  BGNSFT;
;       5334
;       5335  !GPRML (SWQ16, 4, SWF_TRC, YES, 1);                    ! ENABLE DIAGNOSTIC TRACE
;       5336  GPRMD (SWQ1, 0, D, %o'177777', 0, 65535, YES, 1);      ! ERROR LIMIT
;       5337  GPRMD (SWQ2, 2, D, %o'177777', 0, 99, YES, 1);        ! TRANSFER LIMIT
;       5338  GPRMD (SWQ17, 10, D, %o'177777', 0, 100, YES, 1);     ! PERCENT OF RD OPERATIONS
;       5339  GPRMD (SWQ22, 12, D, %o'177777', 0, 144, YES, 1);     ! NUMBER OF DBN'S WRITTEN AT ONE TIME


;       5340  GPRML (SWQ15, 4, SWF_CST, YES, 1);                    ! CLEAR STATISTICAL TABLES ?
;       5341  !GPRML (SWQ20, 4, SWF_FER, YES, 1);                   ! REWRITE BLOCKS WHEN "FORCED ERROR" BIT SET?
;       5342  !GPRML (SWQ21, 4, SWF_HOE, YES, 1);                   ! HALT ON HARD/SOFT ERRORS WITH 'HOE' FLAG SET?
;       5343  GPRML (SWQ4, 4, SWF_RDM, YES, 1);                     ! RANDOM SEEK MODE ?
;       5344  XFERF (SW1);                                          ! IF NO, DO NEXT QUESTION
;       5345  XFER (SW2);
;       5346  $L (SW1);
;       5347  GPRML (SWQ19, 4, SWF_SEQ, YES, 1);                    ! FIXED OR RANDOM SEQUENTAIL STEPPING ?
;       5348  $L (SW2);
;       5349  GPRML (SWQ7, 4, SWF_CRC, YES, 1);                     ! READ-COMPARES AT CONTROLLER ?
;       5350  DISPLAY (SWM1);                                       ! REMAINING QUESTIONS ONLY APPLY ...
;       5351  GPRML (SWQ9, 4, SWF_CWC, YES, 1);                     ! WRITE-COMPARES AT CONTROLLER ?
;       5352  XFERF (SW3);                                          ! IF NO, DO NEXT QUESTION
;       5353  XFER (SW4);
;       5354  $L (SW3);
;       5355  GPRML (SWQ10, 4, SWF_HWC, YES, 1);                    ! CHECK WRITES AT HOST BY READING ?
;       5356  $L (SW4);
;       5357  GPRML (SWQ11, 4, SWF_UDP, YES, 1);                    ! USER-DEFINED DATA PATTERN ?
;       5358  XFERF (SW5);                                          ! IF NO, DO NEXT QUESTION
;       5359  XFER (SW6);
;       5360  $L (SW5);
;       5361  GPRMD (SWQ12, 6, D, %o'177777', 0, DP_CNT, YES, 1);   ! SELECT PRE-DEFINED DATA PATTERN
;       5362  XFER (SW7);                                           ! DONE
;       5363  $L (SW6);
;       5364  GPRMD (SWQ13, 8, D, %o'177777', 1, MAX_UDP_CNT, YES, 1);! NO. OF WORDS IN USER DATA PATTERN
;       5365  GPRMD (SWQ14, 14, O, %o'177777', 0, %o'177777', NO, 8); ! PATTERN VALUES
;       5366  $L (SW7);
;       5367
;       5368  ENDSFT;
```

```
;        5369  %sbttl 'REPORT CODING SECTION'
;        5370
;        5371  !+
;        5372  ! THE REPORT CODING SECTION CONTAINS THE
;        5373  ! "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
;        5374  !-
;        5375
;        5376  BGNRPT;
;        5377                                                          ! PRINTS MSCP DATA
;        5378  PRINTS (RPT1);
;        5379  PRINTS (RPT2);
;        5380  PRINTS (RPT3);
;        5381  PRINTS (RPT4);
;        5382  PRINTS (RPT5);
;        5383  PRINTS (RPT6);
;        5384
;        5385  incr CTLR from 0 to MAX_CTLR - 1 do
;        5386      begin
;        5387      SET_CPAR (.CTLR);
;        5388
;        5389      incr DISK from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;        5390          begin
;        5391          SET_UPAR (.DISK);
;        5392
;        5393          if (.CST_ADDR [.DISK, D_TYPE] eql RX_50) and
;        5394             (.CST_ADDR [.DISK, D_PRES] eql PRESENT)
;        5395          then
;        5396              PRINTS (RPT7, .L$LUN, .CST_ADDR [.DISK, D_DISK_NUM]);
;        5397
;        5398          if (.CST_ADDR [.DISK, D_TYPE] eql RD_51) and
;        5399             (.CST_ADDR [.DISK, D_PRES] eql PRESENT)
;        5400          then
;        5401              PRINTS (RPT8, .L$LUN, .CST_ADDR [.DISK, D_DISK_NUM]);
;        5402
;        5403          if .CST_ADDR [.DISK, D_PRES] eql PRESENT
;        5404          then
;        5405              begin
;      P 5406              PRINTS (RPT9,
;      P 5407                  .T_ADDR [TOT_READS_HI], .T_ADDR [TOT_READS_LO],
;        5408                  .T_ADDR [MTOT_BYT_RED], .T_ADDR [TOT_BYT_RED_HI], .T_ADDR [TOT_BYT_RED_LO]);
;      P 5409              PRINTS (RPT9,
;      P 5410                  .T_ADDR [TOT_WRITES_HI], .T_ADDR [TOT_WRITES_LO],
;        5411                  .T_ADDR [MTOT_BYT_WRT], .T_ADDR [TOT_BYT_WRT_HI], .T_ADDR [TOT_BYT_WRT_LO]);
;      P 5412              PRINTS (RPT10,
;      P 5413                  .T_ADDR [ERR_HRD_SEK], .T_ADDR [ERR_HRD_DAT], .T_ADDR [ERR_HRD_DRV], .T_ADDR [ERR_HRD_HST],
;        5414                  .T_ADDR [ERR_SFT_SEK], .T_ADDR [ERR_SFT_DAT], .T_ADDR [ERR_SFT_DRV], .T_ADDR [ERR_SFT_HST]);
;        5415              end;
;        5416
;        5417          end;
;        5418
;        5419      if .CST [.CTLR, STATE] eql PRESENT
;        5420      then
;        5421          begin
```

```
;         5422          PRINTS (RPT11);
;         5423          PRINTS (RPT12, .C_ERR_TBL [.CTLR, C_ERR_HRD], .C_ERR_TBL [.CTLR, C_ERR_SFT]);
;         5424          end;
;         5425
;         5426      PRINTS (CRLF);
;         5427
;         5428      end;
;         5429  begin                                                          ! PRINTS DUP DATA
;         5430  prints(crlf);
;         5431  PRINTS(RPT13);!^
;         5432  PRINTS(RPT14);
;         5433  PRINTS(RPT15);
;         5434  INCR CTLR FROM 0 TO MAX_CTLR-1 DO
;         5435          BEGIN
;         5436          SET_CPAR(.CTLR);
;         5437          INCR DISK FROM (0+OF_UN) TO (3*UNIT_SIZE+OF_UN) BY UNIT_SIZE DO
;         5438                  BEGIN
;         5439                  SET_UPAR(.DISK);
;         5440                  IF .CST_ADDR[.DISK, D_TYPE] EQLU RD_51 and .CST_ADDR [.DISK, D_PRES] eql PRESENT
;         5441                  THEN
;       P 5442              PRINTS (RPT16,
;       P 5443                  .L$LUN, .CST_ADDR [.DISK, D_DISK_NUM],
;         5444                  .T_ADDR [T_DBN_RD], .T_ADDR [T_BLK_RD], .T_ADDR [T_DBN_WT], .T_ADDR [T_BLK_WT]);
;         5445                  END;
;         5446          END;
;         5447  end;
;         5448
;         5449  PRINTS (CRLF);
;         5450
;         5451  ENDRPT;


                                        .TITLE  ZRQAM2 RD/RX EXERCISER
                                        .IDENT  /V01.2/
                                        .ENABL  AMA


                                        .PSECT  $CODE$,  RO
000000                          L$DVTYP::
000000  122  121  104                   .ASCII  /RQD/
000003  130  061  000                   .ASCII  /X1/<00>
000006                                  .BLKB   2
000010  122  104  057           L$DESC::.ASCII  /RD/<57>
000013  122  130  040                   .ASCII  /RX /
000016  105  130  105                   .ASCII  /EXE/
000021  122  103  111                   .ASCII  /RCI/
000024  123  105  122                   .ASCII  /SER/
000027  000                            .ASCII  <00>
000030                                  .BLKB   2
000032  000000C                 L$HRDLN::
                                        .WORD   <<<L$NDHRD-L$HRDLN>/2>-1>
000034  000031                  GP$1::  .WORD   31
000036  000000G                         .WORD   HWQ1
```

```
000040  160000                          .WORD   -20000
000042  177777                          .WORD   -1
000044  001031                  GP$2::  .WORD   1031
000046  000000G                         .WORD   HWQ2
000050  000004                          .WORD   4
000052  000774                          .WORD   774
000054  002032                  GP$3::  .WORD   2032
000056  000000G                         .WORD   HWQ3
000060  000377                          .WORD   377
000062  000000                          .WORD   0
000064  000007                          .WORD   7
000066  003052                  GP$4::  .WORD   3052
000070  000000G                         .WORD   HWQ4
000072  000003                          .WORD   3
000074  000000                          .WORD   0
000076  000003                          .WORD   3
000100  003130                  GP$5::  .WORD   3130
000102  000000G                         .WORD   HWQ5
000104  000004                          .WORD   4
000106  000000C                  $HW1:   .WORD   <<<<$LHW1-$HW1>*400>+4>+40>
000110  003130                  GP$6::  .WORD   3130
000112  000000G                         .WORD   HWQ10
000114  000010                          .WORD   10
000116  000000C                  $NODU:  .WORD   <<<<$LNODU-$NODU>*400>+4>+40>
000120  003130                  GP$7::  .WORD   3130
000122  000000G                         .WORD   HWQ11
000124  000020                          .WORD   20
000126  001004                  $LNODU: .WORD   1004
000130  004052                  GP$8::  .WORD   4052
000132  000000G                         .WORD   HWQ6
000134  177777                          .WORD   -1
000136  000000                          .WORD   0
000140  052137                          .WORD   52137
000142  005452                  GP$9::  .WORD   5452
000144  000000G                         .WORD   HWQ7
000146  177777                          .WORD   -1
000150  000004                          .WORD   4
000152  052137                          .WORD   52137
000154  000001                          .WORD   1
000156  000000C                  $HW2:   .WORD   <<<$LHW2-$HW2>*400>+4>
000160  001004                  $LHW1:  .WORD   1004
000162  004052                  GP$10:: .WORD   4052
000164  000000G                         .WORD   HWQ6
000166  177777                          .WORD   -1
000170  000000                          .WORD   0
000172  001437                          .WORD   1437
000174  005452                  GP$11:: .WORD   5452
000176  000000G                         .WORD   HWQ7
000200  177777                          .WORD   -1
000202  000004                          .WORD   4
000204  001437                          .WORD   1437
000206  000001                          .WORD   1
000210  001004                  $LHW2:  .WORD   1004
```

```
000212  003120                          GP$12:: .WORD   3120
000214  000000G                                 .WORD   HWQ8
000216  100000                                  .WORD   -100000
000220  000000C                         $HWDONE:.WORD   <<<<$LHWDONE-$HWDONE>*400>+4>+40>
000222  003120                          GP$13:: .WORD   3120
000224  000000G                                 .WORD   HWQ9
000226  100000                                  .WORD   -100000
000230  001004                          $LHWDONE:
                                                .WORD   1004
000232                                  L$NDHRD::
                                                .BLKW   1
000234  000000C                         L$SFTLN::
                                                .WORD   <<<L$NDSFT-L$SFTLN>/2>-1>
000236  000052                          GP$14:: .WORD   52
000240  000000G                                 .WORD   SWQ1
000242  177777                                  .WORD   -1
000244  000000                                  .WORD   0
000246  177777                                  .WORD   -1
000250  001052                          GP$15:: .WORD   1052
000252  000000G                                 .WORD   SWQ2
000254  177777                                  .WORD   -1
000256  000000                                  .WORD   0
000260  000143                                  .WORD   143
000262  005052                          GP$16:: .WORD   5052
000264  000000G                                 .WORD   SWQ17
000266  177777                                  .WORD   -1
000270  000000                                  .WORD   0
000272  000144                                  .WORD   144
000274  006052                          GP$17:: .WORD   6052
000276  000000G                                 .WORD   SWQ22
000300  177777                                  .WORD   -1
000302  000000                                  .WORD   0
000304  000220                                  .WORD   220
000306  002130                          GP$18:: .WORD   2130
000310  000000G                                 .WORD   SWQ15
000312  000200                                  .WORD   200
000314  002130                          GP$19:: .WORD   2130
000316  000000G                                 .WORD   SWQ4
000320  000002                                  .WORD   2
000322  000000C                         $SW1:   .WORD   <<<<$LSW1-$SW1>*400>+4>+40>
000324  000000C                         $SW2:   .WORD   <<<$LSW2-$SW2>*400>+4>
000326  001004                          $LSW1:  .WORD   1004
000330  002130                          GP$20:: .WORD   2130
000332  000000G                                 .WORD   SWQ19
000334  001000                                  .WORD   1000
000336  001004                          $LSW2:  .WORD   1004
000340  002130                          GP$21:: .WORD   2130
000342  000000G                                 .WORD   SWQ7
000344  000004                                  .WORD   4
000346  000003                          GP$DISP::
                                                .WORD   3
000350  000000G                                 .WORD   SWM1
000352  002130                          GP$22:: .WORD   2130
```

```
000354  000000G                         .WORD   SWQ9
000356  000020                         .WORD   20
000360  000000C                 $SW3:   .WORD   <<<<$LSW3-$SW3>*400>+4>+40>
000362  000000C                 $SW4:   .WORD   <<<$LSW4-$SW4>*400>+4>
000364  001004                  $LSW3:  .WORD   1004
000366  002130                  GP$23:: .WORD   2130
000370  000000G                         .WORD   SWQ10
000372  000040                         .WORD   40
000374  001004                  $LSW4:  .WORD   1004
000376  002130                  GP$24:: .WORD   2130
000400  000000G                         .WORD   SWQ11
000402  000100                         .WORD   100
000404  000000C                 $SW5:   .WORD   <<<<$LSW5-$SW5>*400>+4>+40>
000406  000000C                 $SW6:   .WORD   <<<$LSW6-$SW6>*400>+4>
000410  001004                  $LSW5:  .WORD   1004
000412  003052                  GP$25:: .WORD   3052
000414  000000G                         .WORD   SWQ12
000416  177777                         .WORD   -1
000420  000000                         .WORD   0
000422  000025                         .WORD   25
000424  000000C                 $SW7:   .WORD   <<<$LSW7-$SW7>*400>+4>
000426  001004                  $LSW6:  .WORD   1004
000430  004052                  GP$26:: .WORD   4052
000432  000000G                         .WORD   SWQ13
000434  177777                         .WORD   -1
000436  000001                         .WORD   1
000440  000020                         .WORD   20
000442  007222                  GP$27:: .WORD   7222
000444  000000G                         .WORD   SWQ14
000446  177777                         .WORD   -1
000450  000000                         .WORD   0
000452  177777                         .WORD   -1
000454  000004                         .WORD   4
000456  001004                  $LSW7:  .WORD   1004
000460                          L$NDSFT::
                                .BLKW   1


000000                                  .PSECT  $OWN$,  D
000000  000000G                 TBL.SUC:.WORD   NULL
000002  000000G                         .WORD   SC.SDI
000004  000000G                         .WORD   SC.CON
000006  000000G                         .WORD   NULL
000010  000000G                         .WORD   SC.DUP
000012  000000G                         .WORD   NULL
000014  000000G                         .WORD   NULL
000016  000000G                         .WORD   NULL
000020  000000G                         .WORD   SC.ONL
000022  000000G                         .WORD   NULL
000024  000000G                         .WORD   NULL
000026  000000G                         .WORD   NULL
000030  000000G                         .WORD   NULL
```

```
000032  000000G                         .WORD   NULL
000034  000000G                         .WORD   NULL
000036  000000G                         .WORD   NULL
000040  000000G                         .WORD   SC.SON
000042  000000G                TBL.OFL: .WORD   SC.UNK
000044  000000G                         .WORD   SC.VOL
000046  000000G                         .WORD   SC.IOP
000050  000000G                         .WORD   NULL
000052  000000G                         .WORD   SC.DUP
000054  000000G                         .WORD   NULL
000056  000000G                         .WORD   NULL
000060  000000G                         .WORD   NULL
000062  000000G                         .WORD   SC.DIS
000064  000000G                TBL.MFE: .WORD   SC.FER
000066  000000G                         .WORD   NULL
000070  000000G                         .WORD   SC.ISH
000072  000000G                         .WORD   SC.DST
000074  000000G                         .WORD   SC.EC9
000076  000000G                         .WORD   SC.576
000100  000000G                         .WORD   SC.FCT
000102  000000G                         .WORD   SC.ECC
000104  000000G                         .WORD   SC.RCT
000106  000000G                         .WORD   SC.FUL
000110  000000G                         .WORD   SC.EC1
000112  000000G                TBL.WPT: .WORD   NULL
000114  000000G                         .WORD   SC.SWP
000116  000000G                         .WORD   SC.HWP
000120  000000G                TBL.DAT: .WORD   SC.FE2
000122  000000G                         .WORD   NULL
000124  000000G                         .WORD   SC.IS2
000126  000000G                         .WORD   SC.DS2
000130  000000G                         .WORD   SC.EC9
000132  000000G                         .WORD   NULL
000134  000000G                         .WORD   NULL
000136  000000G                         .WORD   SC.ECD
000140  000000G                         .WORD   SC.EC1
000142  000000G                         .WORD   SC.EC2
000144  000000G                         .WORD   SC.EC3
000146  000000G                         .WORD   SC.EC4
000150  000000G                         .WORD   SC.EC5
000152  000000G                         .WORD   SC.EC6
000154  000000G                         .WORD   SC.EC7
000156  000000G                         .WORD   SC.EC8
000160  000000G                TBL.HST: .WORD   NULL
000162  000000G                         .WORD   SC.ODA
000164  000000G                         .WORD   SC.ODB
000166  000000G                         .WORD   SC.NXM
000170  000000G                         .WORD   SC.PAR
000172  000000G                TBL.CNT: .WORD   SC.CTO
000174  000000G                         .WORD   SC.SDS
000176  000000G                         .WORD   SC.EDC
000200  000000G                         .WORD   SC.IDS
000202  000000G                TBL.DRV: .WORD   NULL
```

```
000204  000000G                 .WORD   SC.SRT
000206  000000G                 .WORD   SC.SRI
000210  000000G                 .WORD   SC.POE
000212  000000G                 .WORD   SC.RDY
000214  000000G                 .WORD   SC.CLK
000216  000000G                 .WORD   SC.RSP
000220  000000G                 .WORD   SC.SUR
000222  000000G                 .WORD   SC.PSP

                        .GLOBL  CST, CST.ADDR, DCT, DCT.ADDR, RDRX.ADDR
                        .GLOBL  IRDRX.ADDR, DUPPKT, TALLY, T.ADDR
                        .GLOBL  C.ERR.TBL, MSCP.PKT, IPKT.ADDR
                        .GLOBL  PKT.USE, RETPKT, RP.USE, RP.INDX
                        .GLOBL  RP.ADDR, ELOG.PKT, BUFF.ADDR, BUFF.OWN
                        .GLOBL  IODQ, IODQ.IN, IODQ.OUT, ENTRY.REASON
                        .GLOBL  EOP.FLAG, DUP.FLAGS, CCTLR, CDISK
                        .GLOBL  CUOFF, CTLR.CNT, DUR, QIO, FREE.MEM.ADDR
                        .GLOBL  BYTS.PER.QIO, ST.CODE, SB.CODE
                        .GLOBL  STEP, OF.RC, SA.REG, CMD.TIME
                        .GLOBL  NEX, CRN.LOW, CRN.HIGH, P.INDEX
                        .GLOBL  S.DUPPKT, S.PATTERN, CREDIT.BAL
                        .GLOBL  NEXT.PKT.USE, DBM5, DBM107, DU.MSG
                        .GLOBL  DU.RSN, ERR.COD, ELG.FMT, HWQ1
                        .GLOBL  HWQ2, HWQ3, HWQ4, HWQ5, HWQ6, HWQ7
                        .GLOBL  HWQ8, HWQ9, HWQ10, HWQ11, SWQ1
                        .GLOBL  SWQ2, SWQ4, SWQ7, SWQ9, SWQ10
                        .GLOBL  SWQ11, SWQ12, SWQ13, SWQ14, SWQ15
                        .GLOBL  SWQ17, SWQ19, SWQ22, SWM1, NULL
                        .GLOBL  MSG.01, MSG.02, MSG.03, RPT1, RPT2
                        .GLOBL  RPT3, RPT4, RPT5, RPT6, RPT7, RPT8
                        .GLOBL  RPT9, RPT10, RPT11, RPT12, RPT13
                        .GLOBL  RPT14, RPT15, RPT16, EGS.01, EGS.02
                        .GLOBL  EGD.10, EGD.11, EGD.12, EGD.13
                        .GLOBL  EGD.14, EGD.15, EGD.16, EGD.17
                        .GLOBL  EGD.18, EGD.20, EGD.21, EGD.22
                        .GLOBL  EGD.23, EGH.30, EH.0, EH.1, EH.2
                        .GLOBL  EH.3, EH.4, EH.5, EH.6, EH.7, EH.8
                        .GLOBL  EH.9, EH.10, EH.12, EH.13, EBS.01
                        .GLOBL  EBD.10, EBD.12, EBD.13, EBD.14
                        .GLOBL  EBD.18, EBD.19, ELG.00, EX.BDR
                        .GLOBL  EX.BDW, EX.LBR, EX.LBW, EX.RBN
                        .GLOBL  EX.CBR, EX.CBW, XX13, XX14, XX15
                        .GLOBL  XX16, XX17, XX18, XX19, XX20, XX21
                        .GLOBL  XX22, XX23, XX24, XX25, XX26, XX27
                        .GLOBL  XX29, XX30, XX31, XX32, XX33, XX34
                        .GLOBL  XX35, XX37, XX38, XX39, XX40, XX41
                        .GLOBL  EB.DCT, EB.COMM, EB.PKT, EB.RAL
                        .GLOBL  EB.ADDR, EBNEX1, EB.NEX2, EBNEX3
                        .GLOBL  CER.01, CER.02, EX.SEQ, EX.CRD
                        .GLOBL  EX.MTN, EX.DGM, EX.RD, EX.WRT
                        .GLOBL  EX.ACC, EX.ONL, EX.SCC, EX.GDS
                        .GLOBL  EX.ESP, EX.ELP, EX.SDD, EX.RCD
```

```
                        .GLOBL  EX.ABP, SC.SDI, SC.CON, SC.DUP
                        .GLOBL  SC.ONL, SC.SON, SC.UNK, SC.VOL
                        .GLOBL  SC.IOP, SC.DIS, SC.FER, SC.FE2
                        .GLOBL  SC.ISH, SC.IS2, SC.DST, SC.DS2
                        .GLOBL  SC.ECC, SC.ECD, SC.RCT, SC.FUL
                        .GLOBL  SC.576, SC.FCT, SC.EC1, SC.EC2
                        .GLOBL  SC.EC3, SC.EC4, SC.EC5, SC.EC6
                        .GLOBL  SC.EC7, SC.EC8, SC.EC9, SC.SWP
                        .GLOBL  SC.HWP, SC.ODA, SC.ODB, SC.NXM
                        .GLOBL  SC.PAR, SC.CTO, SC.SDS, SC.EDC
                        .GLOBL  SC.IDS, SC.SRT, SC.SRI, SC.POE
                        .GLOBL  SC.RDY, SC.CLK, SC.RSP, SC.SUR
                        .GLOBL  SC.PSP, F.1, F.2, F.3, F.4, F.5
                        .GLOBL  F.6, F.7, F.8, F.9, F.10, F.11
                        .GLOBL  F.12, F.13, F.14, F.15, F.16, F.17
                        .GLOBL  F.18, F.19, F.20, F.21, EBH.30
                        .GLOBL  EBH.44, EBH.45, EBH.46, EBH.47
                        .GLOBL  EBH.48, EBH.49, DF.0, DF.1, DF.2
                        .GLOBL  DF.3, DF.4, DF.5, DF.6, DF.7, T.QUE
                        .GLOBL  T.DEF, T.INF, T.TER, T.FAT, T.SPL
                        .GLOBL  E.UNT, E.BLK, E.DEV, E.ZER, M.ASC
                        .GLOBL  M.BIN, M.TER, M.COD, M.DAT, M.UR
                        .GLOBL  M.URP, M.UP, M.UL, CNTR.ERR, RDRX.ERR
                        .GLOBL  EX.WRD, EX.OP, SPACE4, CRLF, DASH
                        .GLOBL  ASTERISK, SWP.FLAGS, L$HIMEM, L$LUN
                        .GLOBL  L$UNIT

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=           1000
000400                  BIT8=           400
000200                  BIT7=           200
000100                  BIT6=           100
000040                  BIT5=           40
000020                  BIT4=           20
000010                  BIT3=           10
000004                  BIT2=           4
000002                  BIT1=           2
```

```
000001          BIT0==          1
000040          EF.START==      40
000037          EF.RESTART==    37
000036          EF.CONTINUE==   36
000035          EF.NEW==        35
000034          EF.PWR==        34
000340          PRI07==         340
000300          PRI06==         300
000240          PRI05==         240
000200          PRI04==         200
000140          PRI03==         140
000100          PRI02==         100
000040          PRI01==         40
000000          PRI00==         0
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
000034'         L$HARD==        L$HRDLN+2
000236'         L$SOFT==        L$SFTLN+2


                        .SBTTL  LRPT REPORT CODING SECTION
000462                  .PSECT  $CODE$,  RO

000000  004137  000000G LRPT:   JSR     R1,$SAVE3                       ;                       5368
000004  012746  000000G         MOV     #RPT1,-(SP)                     ;                       5378
000010  012746  000001          MOV     #1,-(SP)
000014  010600                  MOV     SP,R0                           ; SP,*
000016  104416                  TRAP    16
000020  012716  000000G         MOV     #RPT2,(SP)                      ;                       5379
000024  012746  000001          MOV     #1,-(SP)
000030  010600                  MOV     SP,R0                           ; SP,*
000032  104416                  TRAP    16
000034  012716  000000G         MOV     #RPT3,(SP)                      ;                       5380
000040  012746  000001          MOV     #1,-(SP)
000044  010600                  MOV     SP,R0                           ; SP,*
000046  104416                  TRAP    16
000050  012716  000000G         MOV     #RPT4,(SP)                      ;                       5381
000054  012746  000001          MOV     #1,-(SP)
000060  010600                  MOV     SP,R0                           ; SP,*
000062  104416                  TRAP    16
000064  012716  000000G         MOV     #RPT5,(SP)                      ;                       5382
```

```
000070  012746  000001                    MOV     #1,-(SP)
000074  010600                            MOV     SP,R0                   ; SP,*
000076  104416                            TRAP    16
000100  012716  000000G                   MOV     #RPT6,(SP)              ;                       5383
000104  012746  000001                    MOV     #1,-(SP)
000110  010600                            MOV     SP,R0                   ; SP,*
000112  104416                            TRAP    16
000114  005003                            CLR     R3                      ; CTLR                  5385
000116  010316                    1$:     MOV     R3,(SP)                 ; CTLR,*                5387
000120  004737  000000V                   JSR     PC,SET.CPAR
000124  012702  000003                    MOV     #3,R2                   ; *,DISK                5389
000130  010216                    2$:     MOV     R2,(SP)                 ; DISK,*                5391
000132  004737  000000V                   JSR     PC,SET.UPAR
000136  010201                            MOV     R2,R1                   ; DISK,*                5393
000140  006301                            ASL     R1
000142  010100                            MOV     R1,R0
000144  063700  000000G                   ADD     CST.ADDR,R0
000150  132710  000004                    BITB    #4,(R0)
000154  001020                            BNE     3$
000156  032710  040000                    BIT     #40000,(R0)             ;                       5394
000162  001415                            BEQ     3$
000164  111016                            MOVB    (R0),(SP)               ;                       5396
000166  042716  177774                    BIC     #177774,(SP)
000172  013746  000000G                   MOV     L$LUN,-(SP)
000176  012746  000000G                   MOV     #RPT7,-(SP)
000202  012746  000003                    MOV     #3,-(SP)
000206  010600                            MOV     SP,R0                   ; SP,*
000210  104416                            TRAP    16
000212  062706  000006                    ADD     #6,SP
000216  010100                    3$:     MOV     R1,R0                   ;                       5398
000220  063700  000000G                   ADD     CST.ADDR,R0
000224  132710  000004                    BITB    #4,(R0)
000230  001420                            BEQ     4$
000232  032710  040000                    BIT     #40000,(R0)             ;                       5399
000236  001415                            BEQ     4$
000240  111016                            MOVB    (R0),(SP)               ;                       5401
000242  042716  177774                    BIC     #177774,(SP)
000246  013746  000000G                   MOV     L$LUN,-(SP)
000252  012746  000000G                   MOV     #RPT8,-(SP)
000256  012746  000003                    MOV     #3,-(SP)
000262  010600                            MOV     SP,R0                   ; SP,*
000264  104416                            TRAP    16
000266  062706  000006                    ADD     #6,SP
000272  010100                    4$:     MOV     R1,R0                   ;                       5403
000274  063700  000000G                   ADD     CST.ADDR,R0
000300  032710  040000                    BIT     #40000,(R0)
000304  001506                            BEQ     5$
000306  013700  000000G                   MOV     T.ADDR,R0               ;                       5408
000312  016016  000032                    MOV     32(R0),(SP)
000316  016046  000034                    MOV     34(R0),-(SP)
000322  016046  000036                    MOV     36(R0),-(SP)
000326  016046  000016                    MOV     16(R0),-(SP)
000332  016046  000020                    MOV     20(R0),-(SP)
```

```
000336  012746  000000G         MOV     #RPT9,-(SP)
000342  012746  000006          MOV     #6,-(SP)
000346  010600                  MOV     SP,R0                   ; SP,*
000350  104416                  TRAP    16
000352  013700  000000G         MOV     T.ADDR,R0               ;                       5411
000356  016016  000040          MOV     40(R0),(SP)
000362  016046  000042          MOV     42(R0),-(SP)
000366  016046  000044          MOV     44(R0),-(SP)
000372  016046  000024          MOV     24(R0),-(SP)
000376  016046  000026          MOV     26(R0),-(SP)
000402  012746  000000G         MOV     #RPT9,-(SP)
000406  012746  000006          MOV     #6,-(SP)
000412  010600                  MOV     SP,R0                   ; SP,*
000414  104416                  TRAP    16
000416  013700  000000G         MOV     T.ADDR,R0               ;                       5414
000422  005016                  CLR     (SP)
000424  116016  000067          MOVB    67(R0),(SP)
000430  005046                  CLR     -(SP)
000432  116016  000066          MOVB    66(R0),(SP)
000436  005046                  CLR     -(SP)
000440  116016  000065          MOVB    65(R0),(SP)
000444  005046                  CLR     -(SP)
000446  116016  000064          MOVB    64(R0),(SP)
000452  005046                  CLR     -(SP)
000454  116016  000063          MOVB    63(R0),(SP)
000460  005046                  CLR     -(SP)
000462  116016  000062          MOVB    62(R0),(SP)
000466  005046                  CLR     -(SP)
000470  116016  000061          MOVB    61(R0),(SP)
000474  005046                  CLR     -(SP)
000476  116016  000060          MOVB    60(R0),(SP)
000502  012746  000000G         MOV     #RPT10,-(SP)
000506  012746  000011          MOV     #11,-(SP)
000512  010600                  MOV     SP,R0                   ; SP,*
000514  104416                  TRAP    16
000516  062706  000052          ADD     #52,SP                  ;                       5405
000522  062702  000005  5$:     ADD     #5,R2                   ; *,DISK                5389
000526  020227  000022          CMP     R2,#22                  ; DISK,*
000532  003002                  BGT     6$
000534  000137  000612'        JMP     2$
000540  010316          6$:     MOV     R3,(SP)                 ; CTLR,*                5419
000542  012746  000056          MOV     #56,-(SP)
000546  004737  000000G         JSR     PC,BL$MUL
000552  005726                  TST     (SP)+
000554  005760  000002G         TST     CST+2(R0)
000560  100026                  BPL     7$
000562  012716  000000G         MOV     #RPT11,(SP)             ;                       5422
000566  012746  000001          MOV     #1,-(SP)
000572  010600                  MOV     SP,R0                   ; SP,*
000574  104416                  TRAP    16
000576  010300                  MOV     R3,R0                   ; CTLR,*                5423
000600  006300                  ASL     R0
000602  005016                  CLR     (SP)
```

```
000604  116016  000001G                         MOVB    C.ERR.TBL+1(R0),(SP)
000610  005046                                  CLR     -(SP)
000612  116016  000000G                         MOVB    C.ERR.TBL(R0),(SP)
000616  012746  000000G                         MOV     #RPT12,-(SP)
000622  012746  000003                          MOV     #3,-(SP)
000626  010600                                  MOV     SP,R0                   ; SP,*
000630  104416                                  TRAP    16
000632  062706  000010                          ADD     #10,SP                  ;                       5421
000636  012716  000000G         7$:             MOV     #CRLF,(SP)              ;                       5426
000642  012746  000001                          MOV     #1,-(SP)
000646  010600                                  MOV     SP,R0                   ; SP,*
000650  104416                                  TRAP    16
000652  005726                                  TST     (SP)+                   ;                       5386
000654  005203                                  INC     R3                      ; CTLR                  5385
000656  000243                                  .WORD CLV!CLC

000660  003002                                  BGT     8$
000662  000137  000600'                         JMP     1$
000666  012716  000000G         8$:             MOV     #CRLF,(SP)              ;                       5430
000672  012746  000001                          MOV     #1,-(SP)
000676  010600                                  MOV     SP,R0                   ; SP,*
000700  104416                                  TRAP    16
000702  012716  000000G                         MOV     #RPT13,(SP)             ;                       5431
000706  012746  000001                          MOV     #1,-(SP)
000712  010600                                  MOV     SP,R0                   ; SP,*
000714  104416                                  TRAP    16
000716  012716  000000G                         MOV     #RPT14,(SP)             ;                       5432
000722  012746  000001                          MOV     #1,-(SP)
000726  010600                                  MOV     SP,R0                   ; SP,*
000730  104416                                  TRAP    16
000732  012716  000000G                         MOV     #RPT15,(SP)             ;                       5433
000736  012746  000001                          MOV     #1,-(SP)
000742  010600                                  MOV     SP,R0                   ; SP,*
000744  104416                                  TRAP    16
000746  005002                                  CLR     R2                      ; CTLR                  5434
000750  010216                          9$:     MOV     R2,(SP)                 ; CTLR,*                5436
000752  004737  000000V                         JSR     PC,SET.CPAR
000756  012701  000003                          MOV     #3,R1                   ; *,DISK                5437
000762  010116                          10$:    MOV     R1,(SP)                 ; DISK,*                5439
000764  004737  000000V                         JSR     PC,SET.UPAR
000770  010100                                  MOV     R1,R0                   ; DISK,*                5440
000772  006300                                  ASL     R0
000774  063700  000000G                         ADD     CST.ADDR,R0
001000  132710  000004                          BITB    #4,(R0)
001004  001432                                  BEQ     11$
001006  032710  040000                          BIT     #40000,(R0)
001012  001427                                  BEQ     11$
001014  013703  000000G                         MOV     T.ADDR,R3               ;                       5444
001020  016316  000050                          MOV     50(R3),(SP)
001024  016346  000052                          MOV     52(R3),-(SP)
001030  016346  000054                          MOV     54(R3),-(SP)
001034  016346  000056                          MOV     56(R3),-(SP)
001040  111046                                  MOVB    (R0),-(SP)
```

```
001042  042716  177774                          BIC     #177774,(SP)
001046  013746  000000G                         MOV     L$LUN,-(SP)
001052  012746  000000G                         MOV     #RPT16,-(SP)
001056  012746  000007                          MOV     #7,-(SP)
001062  010600                                  MOV     SP,R0           ; SP,*
001064  104416                                  TRAP    16
001066  062706  000016                          ADD     #16,SP
001072  062701  000005              11$:        ADD     #5,R1           ; *,DISK                5437
001076  020127  000022                          CMP     R1,#22          ; DISK,*
001102  003727                                  BLE     10$
001104  005202                                  INC     R2              ; CTLR                  5434
001106  000243                                  .WORD CLV!CLC

001110  003717                                  BLE     9$
001112  012716  000000G                         MOV     #CRLF,(SP)      ;                       5449
001116  012746  000001                          MOV     #1,-(SP)
001122  010600                                  MOV     SP,R0           ; SP,*
001124  104416                                  TRAP    16
001126  062706  000030                          ADD     #30,SP          ;                       5368
001132  000207                                  RTS     PC

; Routine Size:  302 words,     Routine Base:  $CODE$ + 0462
; Maximum stack depth per invocation:  34 words



                                        .SBTTL  L$RPT REPORT CODING SECTION
000000  004737  000462'                L$RPT:: JSR     PC,LRPT         ;                       5449
000004  104425                                  TRAP    25
000006  000207                                  RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 1616
; Maximum stack depth per invocation:  2 words
```

```
;        5452  %sbttl 'INITIALIZE SECTION'
;        5453
;        5454  BGNINIT;
;        5455
;        5456  local
;        5457      DELAY_MULT : word,
;        5458      FLAG : byte,
;        5459      TEMP : word,
;        5460      HWPT_REF : ref block [HWPT_LEN, word] field (HWP_FIELDS),
;        5461      CLEAR_TABLES : byte;
;        5462
;        5463  SETPRI (PRI07);                          ! PRIORITY 7 - NO INTERRUPTS ALLOWED DURING INIT
;        5464
;        5465  if READEF (EF_NEW)                       ! IS THIS A NEW PASS?
;        5466  then
;        5467      begin
;        5468      ENTRY_REASON = NEW_PASS;
;        5469
;        5470      if (.SWP_FLAGS and SWF_CST) neq SWF_CST
;        5471      then
;        5472          CLEAR_TABLES = FALSE
;        5473      else
;        5474          CLEAR_TABLES = TRUE;
;        5475
;        5476      end;
;        5477
;        5478  if READEF (EF_START)                     ! IS THIS A START?
;        5479  then
;        5480      begin
;        5481      BRESET;
;        5482      ENTRY_REASON = START;
;        5483      CLEAR_TABLES = TRUE;
;        5484      end;
;        5485
;        5486  if READEF (EF_RESTART)                   ! IS THIS A RESTART?
;        5487  then
;        5488      begin
;        5489      ENTRY_REASON = RESTART;
;        5490      CLEAR_TABLES = TRUE;
;        5491      end;
;        5492
;        5493  if READEF (EF_CONTINUE)                  ! IS THIS A CONTINUE?
;        5494  then
;        5495      begin
;        5496      ENTRY_REASON = CONT;
;        5497
;        5498      if (.SWP_FLAGS and SWF_CST) neq SWF_CST
;        5499      then
;        5500          CLEAR_TABLES = FALSE
;        5501      else
;        5502          CLEAR_TABLES = TRUE;
;        5503
;        5504      end;
```

```
5505
5506  if READEF (EF_PWR)                          ! ARE WE HERE BECAUSE OF POWER FAIL?
5507  then
5508      begin
5509      ENTRY_REASON = PWR_FAIL;
5510      CLEAR_TABLES = TRUE;
5511      PRINTF (MSG_01);                        ! "POWER DELAY - WAITING"
5512
5513      incr COUNT from 0 to 60 do              ! WAIT APPROX. 60 SECONDS
5514          begin
5515          DELAY_MULT = 333;
5516          DELAY (.DELAY_MULT);
5517          BREAK;                              ! BREAK FOR ACT
5518          end;
5519
5520      end;
5521
5522  !SETVEC (O_TVEC, O_BRK, PRI07);             ! SET ODT TRAP VECTOR
5523
5524  !+
5525  !       MAKE SURE THAT NOT MORE THAN MAX_UNITS HAVE BEEN SPECIFIED.
5526  !       IF THERE ARE TOO MANY, NOTIFY USER AND RETURN TO SUPERVISOR.
5527  !       (DIAGNOSTIC IS ABORTED).
5528  !-
5529
5530  if .L$UNIT gtru MAX_UNITS
5531  then
5532      begin
5533      ERRSF (1, EGS_01, EMS_01);
5534      DOCLN;
5535      end;
5536
5537  !+
5538  !       THE FOLLOWING CODE IS EXECUTED FOR ALL ENTRY REASONS EXCEPT NEW_PASS.
5539  !       ALL RUN-TIME CONTROLLER STATUS TABLES (CSTs) ARE CLEARED TO 0, THEN
5540  !       LOADED WITH CONFIGURATION DATA FROM THE HARDWARE P-TABLES.
5541  !-
5542
5543  if (.ENTRY_REASON neq NEW_PASS)
5544  then
5545      begin
5546
5547      incr COUNT from 0 to ((MAX_CTLR * CST_LEN * 2) - 2) by 2 do
5548          (CST + .COUNT) = 0;
5549
5550      incr UNIT from 0 to (.L$UNIT - 1) do        ! LOOP THROUGH ALL UNITS
5551
5552          if GPHARD (.UNIT, HWPT_REF) neqa 0      ! IF HWP TABLE FOUND
5553          then
5554              begin
5555              FLAG = NOT_FOUND;
5556
5557              incr CTLR from 0 to (MAX_CTLR - 1) do       ! LOOP THROUGH ALL CSTs
```

```
;        5558
;        5559                    if .CST [.CTLR, IP_ADDR] eqla .HWPT_REF [HWP_IP_ADDR]
;        5560                    then
;        5561
;        5562                        if .CST [.CTLR, .HWPT_REF [HWP_DISK_NUM] * UNIT_SIZE + OF_UN, D_PRES] eql NOT_PRESENT
;        5563                        then
;        5564                            begin                                               ! IF EMPTY SLOT FOUND
;        5565                            TEMP = .HWPT_REF [HWP_DISK_NUM] * UNIT_SIZE + OF_UN;
;        5566                            CST [.CTLR, .TEMP, D_ALL] = .HWPT_REF [HWP_DISK];       ! COPY DISK ADDR AND PROT BIT
;        5567                            CST [.CTLR, .TEMP, D_UNIT] = .UNIT;
;        5568                            CST [.CTLR, .TEMP, D_FATAL] = FALSE;
;        5569                            CST [.CTLR, .TEMP, D_PRES] = PRESENT;
;        5570                            CST [.CTLR, .TEMP + 1, D_BEG] = .HWPT_REF [HWP_BEG_TRK];
;        5571                            CST [.CTLR, .TEMP + 2, D_END] = .HWPT_REF [HWP_END_TRK];
;        5572                            CST [.CTLR, .TEMP + 3, D_DBN] = 0;
;        5573                            CST [.CTLR, .TEMP + 3, NODUPMEDIA] = NOT(.HWPT_REF [HWP_DUPEX]);
;        5574                            CST [.CTLR, .TEMP + 3, DUPWRITE] = (.HWPT_REF [HWP_DUPWT]);
;        5575                            CST [.CTLR, .TEMP + 4, D_COUNT] = 0;
;        5576
;        5577                            if (.CST [.CTLR, .TEMP, D_TYPE] eql RX_50) and
;        5578                               (.CST [.CTLR, .TEMP + 2, D_END] gtru RX_MAX_LBN)
;        5579                            then
;        5580                                CST [.CTLR, .TEMP + 2, D_END] = RX_MAX_LBN;
;        5581
;        5582                            FLAG = FOUND;
;        5583                            exitloop;
;        5584                            end
;        5585                        else
;        5586                            begin                               ! DUPLICATE UNIT
;        5587                            PRINTF (CER_01, .HWPT_REF [HWP_DISK_NUM], .HWPT_REF [HWP_IP_ADDR]);
;        5588                                                                ! "DUPLICATE UNIT: XX AT IP: XXXXXX"
;        5589                            DUR [.UNIT] = DU_CONF;              ! CONFIGURATION ERROR
;        5590                            DODU (.UNIT);                       ! DROP UNIT
;        5591                            FLAG = FOUND;
;        5592                            exitloop;
;        5593                            end;
;        5594
;        5595                    if .FLAG eql NOT_FOUND                      ! IF NO IP ADDR MATCH TO EXISTING CST
;        5596                    then
;        5597                        begin
;        5598
;        5599                        incr CTLR from 0 to (MAX_CTLR - 1) do   ! LOOP THROUGH EACH CST
;        5600
;        5601                            if .CST [.CTLR, IP_ADDR] eqla 0     ! IF EMPTY CST FOUND
;        5602                            then
;        5603                                begin
;        5604                                FLAG = FOUND;
;        5605                                CST [.CTLR, IP_ADDR] = .HWPT_REF [HWP_IP_ADDR];
;        5606                                CST [.CTLR, VEC_ADDR] = .HWPT_REF [HWP_VECTOR];
;        5607                                CST [.CTLR, BR_LEV] = .HWPT_REF [HWP_BR_LEVEL];
;        5608                                TEMP = .HWPT_REF [HWP_DISK_NUM] * UNIT_SIZE + OF_UN;
;        5609                                CST [.CTLR, .TEMP, D_ALL] = .HWPT_REF [HWP_DISK];       !COPY DISK ADDR AND PROT BIT
;        5610                                CST [.CTLR, .TEMP, D_UNIT] = .UNIT;
```

```
5611                        CST [.CTLR, .TEMP, D_FATAL] = FALSE;
5612                        CST [.CTLR, .TEMP, D_PRES] = PRESENT;
5613                        CST [.CTLR, .TEMP + 1, D_BEG] = .HWPT_REF [HWP_BEG_TRK];
5614                        CST [.CTLR, .TEMP + 2, D_END] = .HWPT_REF [HWP_END_TRK];
5615                        CST [.CTLR, .TEMP + 3, D_DBN] = 0;
5616                        CST [.CTLR, .TEMP + 3, NODUPMEDIA] = NOT(.HWPT_REF [HWP_DUPEX]);
5617                              ! CHECK TO SEE IF PROGRAMMER WANTS TO NOT WRITE TO DBNs.
5618
5619                        CST [.CTLR, .TEMP + 3, DUPWRITE] = (.HWPT_REF [HWP_DUPWT]);
5620                        CST [.CTLR, .TEMP + 4, D_COUNT] = 0;
5621
5622                        if (.CST [.CTLR, .TEMP, D_TYPE] eql RX_50) and
5623                           (.CST [.CTLR, .TEMP + 2, D_END] gtru RX_MAX_LBN)
5624                        then
5625                            CST [.CTLR, .TEMP + 2, D_END] = RX_MAX_LBN;
5626
5627                        exitloop;
5628                        end;                            ! IF EMPTY CST FOUND
5629
5630                    if .FLAG eql NOT_FOUND              ! IF NO EMPTY CST FOUND
5631                    then
5632                        begin
5633                        PRINTF (CER_02, MAX_CTLR);      ! "MORE THAN X DIFFERENT IP ADDRESSES."
5634                        DUR [.UNIT] = DU_CONF;          ! CONFIGURATION ERROR
5635                        DODU (.UNIT);                   ! DROP UNIT
5636                        end;
5637
5638                    end;                                ! IF NO IP ADDR MATCH IN CST
5639
5640                end;                                    ! IF GPHARD RETURNS A HWP TABLE
5641            !
5642            ! CONFIGURATON CHECK FOR LEGAL RDRX UNIT MIX BECAUSE WE HAVE DIFFERENT
5643            ! DRIVES :  THE RD51, AND RX50.
5644            ! (NEEDED?)
5645            !
5646            end;                                        ! END OF "NON NEW_PASS" INITIALIZATION
5647
5648    if .ENTRY_REASON eql NEW_PASS
5649    then
5650        begin
5651
5652        incr UNIT from 0 to (.L$UNIT - 1) do
5653            GPHARD (.UNIT, HWPT_REF);                   ! DUMMY GPHARDs FOR NEW PASS
5654
5655        incr CTLR from 0 to (MAX_CTLR - 1) do
5656            begin
5657            CST [.CTLR, U_CNT] = 0;                     ! REINITIALIZE UNIT COUNT
5658
5659            incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
5660                CST [.CTLR, .OFFSET, D_STAT] = OFFLINE; ! START EACH UNIT AS OFFLINE
5661
5662            end;
5663
```

```
5664        end;
5665
5666    if .ENTRY_REASON eql START
5667    then
5668        begin
5669        CTLR_CNT = 0;                               ! NUMBER OF CONFIGURED CONTROLLERS
5670
5671        incr CTLR from 0 to (MAX_CTLR - 1) do
5672
5673            if .CST [.CTLR, IP_ADDR] neqa 0         ! IF CONTROLLER IS PRESENT
5674            then
5675                CTLR_CNT = .CTLR_CNT + 1;           ! INCREMENT CONTROLLER COUNT
5676
5677        MEMORY (FREE_MEM_ADDR);                     ! GET START OF FREE MEMORY
5678
5679        end;                                        ! END OF "START" INITIALIZATION
5680
5681    !+
5682    !       CLEAR STATISTICS TABLES
5683    !-
5684
5685    incr UNITS from 0 to MAX_UNITS - 1 do           ! CLEAR CURRENT STATISTICS
5686        incr COUNT from 0 to TALLY_CLEAR - 1 do
5687            TALLY [.UNITS * TALLY_LEN + .COUNT] = 0;
5688
5689    if .CLEAR_TABLES                                ! IF CLEAR TABLES ON EVERY PASS
5690    then
5691        incr UNITS from 0 to MAX_UNITS - 1 do
5692            incr COUNT from TALLY_CLEAR to TALLY_LEN - 1 do         ! INITIALIZE TOTALS
5693                TALLY [.UNITS * TALLY_LEN + .COUNT] = 0;            !
5694
5695    if .CLEAR_TABLES
5696    then
5697        incr CTLR from 0 to MAX_CTLR - 1 do
5698            begin
5699            C_ERR_TBL [.CTLR, C_ERR_HRD] = 0;       ! INITALIZE CONTROLLER ERRORS
5700            C_ERR_TBL [.CTLR, C_ERR_SFT] = 0;       !
5701            end;
5702
5703    !+
5704    !       MISCELLANEOUS INITIALIZATON
5705    !-
5706
5707    incr CTLR from 0 to (MAX_CTLR - 1) do           ! INITIALIZE NO. OF OUTSTANDING QIOS
5708        QIO [.CTLR] = 0;
5709
5710    CRN_LOW = CRN_HIGH = 0;                         ! INITIALIZE COMMAND REF NUMBER
5711
5712    SETPRI (PRI00);                                 ! SET PROGRAM PRIORITY TO 0
5713
5714    ENDINIT;
```

```
                                        .GLOBL  L$DLY


                                        .SBTTL  LINIT INITIALIZE SECTION
000000 004137 000000G          LINIT:   JSR     R1,$SAVE5                      ;                                       5451
000004 162706 000020                    SUB     #20,SP
000010 012700 000340                    MOV     #340,R0                        ;                                       5463
000014 104441                           TRAP    41
000016 012700 000035                    MOV     #35,R0                         ;                                       5465
000022 104447                           TRAP    47
000024 103012                           BHIS    2$
000026 112737 000005 000000G            MOVB    #5,ENTRY.REASON                ;                                       5468
000034 105737 000000G                   TSTB    SWP.FLAGS                      ;                                       5470
000040 100402                           BMI     1$
000042 105005                           CLRB    R5                             ; CLEAR.TABLES                          5472
000044 000402                           BR      2$                             ;                                       5470
000046 112705 000001           1$:      MOVB    #1,R5                          ; *,CLEAR.TABLES                        5474
000052 012700 000040           2$:      MOV     #40,R0                         ;                                       5478
000056 104447                           TRAP    47
000060 103006                           BHIS    3$
000062 104433                           TRAP    33                             ;                                       5480
000064 112737 000001 000000G            MOVB    #1,ENTRY.REASON                ;                                       5482
000072 112705 000001                    MOVB    #1,R5                          ; *,CLEAR.TABLES                        5483
000076 012700 000037           3$:      MOV     #37,R0                         ;                                       5486
000102 104447                           TRAP    47
000104 103005                           BHIS    4$
000106 112737 000002 000000G            MOVB    #2,ENTRY.REASON                ;                                       5489
000114 112705 000001                    MOVB    #1,R5                          ; *,CLEAR.TABLES                        5490
000120 012700 000036           4$:      MOV     #36,R0                         ;                                       5493
000124 104447                           TRAP    47
000126 103012                           BHIS    6$
000130 112737 000003 000000G            MOVB    #3,ENTRY.REASON                ;                                       5496
000136 105737 000000G                   TSTB    SWP.FLAGS                      ;                                       5498
000142 100402                           BMI     5$
000144 105005                           CLRB    R5                             ; CLEAR.TABLES                          5500
000146 000402                           BR      6$                             ;                                       5498
000150 112705 000001           5$:      MOVB    #1,R5                          ; *,CLEAR.TABLES                        5502
000154 012700 000034           6$:      MOV     #34,R0                         ;                                       5506
000160 104447                           TRAP    47
000162 103036                           BHIS    12$
000164 112737 000004 000000G            MOVB    #4,ENTRY.REASON                ;                                       5509
000172 112705 000001                    MOVB    #1,R5                          ; *,CLEAR.TABLES                        5510
000176 012746 000000G                   MOV     #MSG.01,-(SP)                  ;                                       5511
000202 012746 000001                    MOV     #1,-(SP)
000206 010600                           MOV     SP,R0                          ; SP,*
000210 104417                           TRAP    17
000212 012702 000075                    MOV     #75,R2                         ; *,COUNT                               5513
000216 012703 000515           7$:      MOV     #515,R3                        ; *,DELAY.MULT                          5515
000222 010301                           MOV     R3,R1                          ; DELAY.MULT,$$TMP2                     5516
000224 001411                  8$:      BEQ     11$
000226 013700 000000G                   MOV     L$DLY,R0                       ; *,$$TMP1
000232 001404                           BEQ     10$
000234 005066 000022           9$:      CLR     22(SP)                         ; $$TMP
```

```
000240  005300                          DEC     R0                              ; $$TMP1
000242  001374                          BNE     9$
000244  005301                  10$:     DEC     R1                              ; $$TMP2
000246  000766                          BR      8$
000250  104422                  11$:     TRAP    22
000252  005302                          DEC     R2                              ; COUNT                         5513
000254  001360                          BNE     7$
000256  022626                          CMP     (SP)+,(SP)+                     ;                               5508
000260  023727  000000G 000004   12$:     CMP     L$UNIT,#4                       ;                               5530
000266  101405                          BLOS    13$
000270  104454                          TRAP    54                              ;                               5533
000272  000001                          .WORD   1
000274  000000G                         .WORD   EGS.01
000276  000000V                         .WORD   EMS.01
000300  104444                          TRAP    44
000302  123727  000000G 000005   13$:     CMPB    ENTRY.REASON,#5                 ;                               5543
000310  001002                          BNE     14$
000312  000137  003506'                 JMP     37$
000316  005000                  14$:     CLR     R0                              ; COUNT                         5547
000320  005060  000000G         15$:     CLR     CST(R0)                         ; *(COUNT)                      5548
000324  062700  000002                  ADD     #2,R0                           ; *,COUNT                       5547
000330  020027  000054                  CMP     R0,#54                          ; COUNT,*
000334  003771                          BLE     15$
000336  013766  000000G 000014          MOV     L$UNIT,14(SP)                   ;                               5550
000344  005066  000010                  CLR     10(SP)                          ; UNIT
000350  000137  003462'                 JMP     35$
000354  016600  000010          16$:     MOV     10(SP),R0                       ; UNIT,*                        5552
000360  104442                          TRAP    42
000362  010066  000012                  MOV     R0,12(SP)                       ; *,HWPT.REF
000366  001002                          BNE     17$
000370  000137  003456'                 JMP     34$
000374  105066  000006          17$:     CLRB    6(SP)                           ; FLAG                          5555
000400  005004                          CLR     R4                              ; CTLR                          5557
000402  010446                  18$:     MOV     R4,-(SP)                        ; CTLR,*                        5559
000404  012746  000056                  MOV     #56,-(SP)
000410  004737  000000G                 JSR     PC,BL$MUL
000414  022626                          CMP     (SP)+,(SP)+
000416  026076  000000G 000012          CMP     CST(R0),@12(SP)                 ; *,HWPT.REF
000424  001402                          BEQ     19$
000426  000137  002716'                 JMP     24$
000432  012766  000001  000002  19$:     MOV     #1,2(SP)                        ;                               5582
000440  112766  000001  000006          MOVB    #1,6(SP)                        ; *,FLAG
000446  012700  000006                  MOV     #6,R0                           ;                               5562
000452  066600  000012                  ADD     12(SP),R0                       ; HWPT.REF,*
000456  010066  000004                  MOV     R0,4(SP)
000462  111046                          MOVB    (R0),-(SP)
000464  042716  177774                  BIC     #177774,(SP)
000470  012746  000005                  MOV     #5,-(SP)
000474  004737  000000G                 JSR     PC,BL$MUL
000500  010003                          MOV     R0,R3
000502  005726                          TST     (SP)+
000504  010416                          MOV     R4,(SP)                         : CTLR,*
000506  012746  000027                  MOV     #27,-(SP)
```

```
000512  004737  000000G                        JSR     PC,BL$MUL
000516  010001                                  MOV     R0,R1
000520  022626                                  CMP     (SP)+,(SP)+
000522  060300                                  ADD     R3,R0
000524  006300                                  ASL     R0
000526  032760  040000  000006G                BIT     #40000,CST+6(R0)
000534  001130                                  BNE     23$
000536  010302                                  MOV     R3,R2                   ; *,TEMP                5565
000540  062702  000003                          ADD     #3,R2                   ; *,TEMP
000544  010100                                  MOV     R1,R0                   ;                       5566
000546  060200                                  ADD     R2,R0                   ; TEMP,*
000550  006300                                  ASL     R0
000552  012703  000000G                        MOV     #CST,R3
000556  060003                                  ADD     R0,R3
000560  017613  000004                          MOV     @4(SP),(R3)
000564  016600  000010                          MOV     10(SP),R0               ; UNIT,*                5567
000570  000300                                  SWAB    R0
000572  042700  170377                          BIC     #170377,R0
000576  042713  007400                          BIC     #7400,(R3)
000602  050013                                  BIS     R0,(R3)
000604  042713  010000                          BIC     #10000,(R3)             ;                       5568
000610  052713  040000                          BIS     #40000,(R3)             ;                       5569
000614  010100                                  MOV     R1,R0                   ;                       5570
000616  060200                                  ADD     R2,R0                   ; TEMP,*
000620  006300                                  ASL     R0
000622  016646  000012                          MOV     12(SP),-(SP)            ; HWPT.REF,*
000626  062716  000010                          ADD     #10,(SP)
000632  013660  000002G                        MOV     @(SP)+,CST+2(R0)
000636  010100                                  MOV     R1,R0                   ;                       5571
000640  060200                                  ADD     R2,R0                   ; TEMP,*
000642  006300                                  ASL     R0
000644  012716  000004G                        MOV     #CST+4,(SP)
000650  060016                                  ADD     R0,(SP)
000652  016600  000012                          MOV     12(SP),R0               ; HWPT.REF,*
000656  016076  000012  000000                  MOV     12(R0),@0(SP)           ; *(HWPT.REF),*
000664  010100                                  MOV     R1,R0                   ;                       5572
000666  060200                                  ADD     R2,R0                   ; TEMP,*
000670  006300                                  ASL     R0
000672  062700  000006G                        ADD     #CST+6,R0
000676  105010                                  CLRB    (R0)
000700  005046                                  CLR     -(SP)                   ;                       5573
000702  132776  000010  000006                  BITB    #10,@6(SP)
000710  001401                                  BEQ     20$
000712  005216                                  INC     (SP)
000714  005116                          20$:    COM     (SP)
000716  011646                                  MOV     (SP),-(SP)
000720  042710  100000                          BIC     #100000,(R0)
000724  006026                                  ROR     (SP)+
000726  103002                                  BCC     21$
000730  052710  100000                          BIS     #100000,(R0)
000734  117616  000006                  21$:    MOVB    @6(SP),(SP)             ;                       5574
000740  042710  010000                          BIC     #10000,(R0)
000744  032726  000020                          BIT     #20,(SP)+
```

```
000750  001402                          BEQ     22$
000752  052710  010000                  BIS     #10000,(R0)
000756  010100                  22$:     MOV     R1,R0                   ;                               5575
000760  060200                          ADD     R2,R0                   ; TEMP,*
000762  006300                          ASL     R0
000764  005060  000010G                 CLR     CST+10(R0)
000770  132713  000004                  BITB    #4,(R3)                 ;                               5577
000774  001042                          BNE     25$
000776  027627  000000  001437          CMP     @0(SP),#1437            ;                               5578
001004  101436                          BLOS    25$
001006  012776  001437  000000          MOV     #1437,@0(SP)            ;                               5580
001014  000432                          BR      25$                     ;                               5564
001016  017646  000012          23$:    MOV     @12(SP),-(SP)           ; HWPT.REF,*                    5587
001022  117646  000006                  MOVB    @6(SP),-(SP)
001026  042716  177774                  BIC     #177774,(SP)
001032  012746  000000G                 MOV     #CER.01,-(SP)
001036  012746  000003                  MOV     #3,-(SP)
001042  010600                          MOV     SP,R0                   ; SP,*
001044  104417                          TRAP    17
001046  062706  000010                  ADD     #10,SP
001052  016600  000010                  MOV     10(SP),R0               ; UNIT,*                        5589
001056  112760  000001  000000G         MOVB    #1,DUR(R0)              ; *,*(UNIT)
001064  104451                          TRAP    51                      ;                               5590
001066  000405                          BR      25$                     ;                               5586
001070  005204                  24$:    INC     R4                      ; CTLR                          5557
001072  000243                          .WORD CLV!CLC

001074  003002                          BGT     25$
001076  000137  002230'                JMP     18$
001102  105766  000006          25$:    TSTB    6(SP)                   ; FLAG                          5595
001106  001402                          BEQ     26$
001110  000137  003456'                JMP     34$
001114  005004                  26$:    CLR     R4                      ; CTLR                          5599
001116  010446                  27$:    MOV     R4,-(SP)                ; CTLR,*                        5601
001120  012746  000056                  MOV     #56,-(SP)
001124  004737  000000G                 JSR     PC,BL$MUL
001130  022626                          CMP     (SP)+,(SP)+
001132  005760  000000G                 TST     CST(R0)
001136  001402                          BEQ     28$
001140  000137  003376'                JMP     32$
001144  112766  000001  000006  28$:    MOVB    #1,6(SP)                ; *,FLAG                        5604
001152  017660  000012  000000G         MOV     @12(SP),CST(R0)         ; HWPT.REF,*                    5605
001160  016603  000012                  MOV     12(SP),R3               ; HWPT.REF,*                    5606
001164  016301  000002                  MOV     2(R3),R1                ; *(HWPT.REF),*
001170  042701  177000                  BIC     #177000,R1
001174  042760  000777  000002G         BIC     #777,CST+2(R0)
001202  050160  000002G                 BIS     R1,CST+2(R0)
001206  010301                          MOV     R3,R1                   ; HWPT.REF,*                    5607
001210  116160  000004  000004G         MOVB    4(R1),CST+4(R0)         ; *(HWPT.REF),*
001216  012700  000006                  MOV     #6,R0                   ;                               5608
001222  060300                          ADD     R3,R0                   ; HWPT.REF,*
001224  010066  000004                  MOV     R0,4(SP)
001230  111046                          MOVB    (R0),-(SP)
```

```
001232  042716  177774                  BIC     #177774,(SP)
001236  012746  000005                  MOV     #5,-(SP)
001242  004737  000000G                 JSR     PC,BL$MUL
001246  005726                          TST     (SP)+
001250  010002                          MOV     R0,R2                   ; *,TEMP
001252  062702  000003                  ADD     #3,R2                   ; *,TEMP
001256  010416                          MOV     R4,(SP)                 ; CTLR,*                        5609
001260  012746  000027                  MOV     #27,-(SP)
001264  004737  0000C0G                 JSR     PC,BL$MUL
001270  010001                          MOV     R0,R1
001272  005726                          TST     (SP)+
001274  060200                          ADD     R2,R0                   ; TEMP,*
001276  006300                          ASL     R0
001300  012703  000000G                 MOV     #CST,R3
001304  060003                          ADD     R0,R3
001306  017613  000006                  MOV     @6(SP),(R3)
001312  016600  000012                  MOV     12(SP),R0               ; UNIT,*                        5610
001316  000300                          SWAB    R0
001320  042700  170377                  BIC     #170377,R0
001324  042713  007400                  BIC     #7400,(R3)
001330  050013                          BIS     R0,(R3)
001332  042713  010000                  BIC     #10000,(R3)             ;                               5611
001336  052713  040000                  BIS     #40000,(R3)             ;                               5612
001342  010100                          MOV     R1,R0                   ;                               5613
001344  060200                          ADD     R2,R0                   ; TEMP,*
001346  006300                          ASL     R0
001350  016616  000014                  MOV     14(SP),(SP)             ; HWPT.REF,*
001354  062716  000010                  ADD     #10,(SP)
001360  013660  000002G                 MOV     @(SP)+,CST+2(R0)
001364  010100                          MOV     R1,R0                   ;                               5614
001366  060200                          ADD     R2,R0                   ; TEMP,*
001370  006300                          ASL     R0
001372  012766  000004G 000002          MOV     #CST+4,2(SP)
001400  060066  000002                  ADD     R0,2(SP)
001404  016600  000012                  MOV     12(SP),R0               ; HWPT.REF,*
001410  016076  000012  000002          MOV     12(R0),@2(SP)           ; *(HWPT.REF),*
001416  010100                          MOV     R1,R0                   ;                               5615
001420  060200                          ADD     R2,R0                   ; TEMP,*
001422  006300                          ASL     R0
001424  062700  000006G                 ADD     #CST+6,R0
001430  105010                          CLRB    (R0)
001432  005046                          CLR     -(SP)                   ;                               5616
001434  132776  000010  000006          BITB    #10,@6(SP)
001442  001401                          BEQ     29$
001444  005216                          INC     (SP)
001446  005116                  29$:    COM     (SP)
001450  011646                          MOV     (SP),-(SP)
001452  042710  100000                  BIC     #100000,(R0)
001456  006026                          ROR     (SP)+
001460  103002                          BCC     30$
001462  052710  100000                  BIS     #100000,(R0)
001466  117616  000006          30$:    MOVB    @6(SP),(SP)             :                               5619
001472  042710  010000                  BIC     #10000,(R0)
```

```
001476  032726  000020                      BIT     #20,(SP)+
001502  001402                              BEQ     31$
001504  052710  010000                      BIS     #10000,(R0)
001510  010100                      31$:     MOV     R1,R0                   ;                       5620
001512  060200                              ADD     R2,R0                   ; TEMP,*
001514  006300                              ASL     R0
001516  005060  000010G                     CLR     CST+10(R0)
001522  132713  000004                      BITB    #4,(R3)                 ;                       5622
001526  001015                              BNE     33$
001530  027627  000002  001437              CMP     @2(SP),#1437            ;                       5623
001536  101411                              BLOS    33$
001540  012776  001437  000002              MOV     #1437,@2(SP)            ;                       5625
001546  000405                              BR      33$                     ;                       5603
001550  005204                      32$:     INC     R4                      ; CTLR                  5599
001552  000243                              .WORD CLV!CLC

001554  003002                              BGT     33$
001556  000137  002744'                     JMP     27$
001562  105766  000006              33$:     TSTB    6(SP)                   ; FLAG                  5630
001566  001020                              BNE     34$
001570  012746  000001                      MOV     #1,-(SP)                ;                       5633
001574  012746  000000G                     MOV     #CER.02,-(SP)
001600  012746  000002                      MOV     #2,-(SP)
001604  010600                              MOV     SP,R0                   ; SP,*
001606  104417                              TRAP    17
001610  016600  000016                      MOV     16(SP),R0               ; UNIT,*                5634
001614  112760  000001  000000G             MOVB    #1,DUR(R0)              ; *,*(UNIT)
001622  104451                              TRAP    51                      ;                       5635
001624  062706  000006                      ADD     #6,SP                   ;                       5632
001630  005266  000010              34$:     INC     10(SP)                  ; UNIT                  5550
001634  026666  000010  000014      35$:     CMP     10(SP),14(SP)           ; UNIT,*
001642  002002                              BGE     36$
001644  000137  002202'                     JMP     16$
001650  123727  000000G 000005      36$:     CMPB    ENTRY.REASON,#5         ;                       5648
001656  001052                              BNE     42$
001660  013701  000000G              37$:     MOV     L$UNIT,R1               ;                       5652
001664  005002                              CLR     R2                      ; UNIT
001666  000405                              BR      39$
001670  010200                      38$:     MOV     R2,R0                   ; UNIT,*                5653
001672  104442                              TRAP    42
001674  010066  000012                      MOV     R0,12(SP)               ; *,HWPT.REF
001700  005202                              INC     R2                      ; UNIT                  5652
001702  020201                      39$:     CMP     R2,R1                   ; UNIT,*
001704  002771                              BLT     38$
001706  005004                              CLR     R4                      ; CTLR                  5655
001710  010446                      40$:     MOV     R4,-(SP)                ; CTLR,*                5657
001712  012746  000056                      MOV     #56,-(SP)
001716  004737  000000G                     JSR     PC,BL$MUL
001722  105060  000005G                     CLRB    CST+5(R0)
001726  010416                              MOV     R4,(SP)                 ; CTLR,*                5660
001730  012746  000027                      MOV     #27,-(SP)
001734  004737  000000G                     JSR     PC,BL$MUL
001740  012702  000003                      MOV     #3,R2                   ; *,OFFSET              5659
```

```
001744  010001                  41$:    MOV     R0,R1                           ;                                       5660
001746  060201                          ADD     R2,R1                           ; OFFSET,*
001750  006301                          ASL     R1
001752  042761  020000  000000G         BIC     #20000,CST(R1)
001760  062702  000005                  ADD     #5,R2                           ; *,OFFSET                              5659
001764  020227  000022                  CMP     R2,#22                          ; OFFSET,*
001770  003765                          BLE     41$
001772  062706  000006                  ADD     #6,SP                           ;                                       5656
001776  005204                          INC     R4                              ; CTLR                                  5655
002000  000243                          .WORD CLV!CLC

002002  003742                          BLE     40$
002004  123727  000000G 000001  42$:    CMPB    ENTRY.REASON,#1                 ;                                       5666
002012  001017                          BNE     45$
002014  005037  000000G                 CLR     CTLR.CNT                        ;                                       5669
002020  005000                          CLR     R0                              ; CTLR                                  5671
002022  005760  000000G         43$:    TST     CST(R0)                         ; *(CTLR)                               5673
002026  001402                          BEQ     44$
002030  005237  000000G                 INC     CTLR.CNT                        ;                                       5675
002034  062700  000056          44$:    ADD     #56,R0                          ; *,CTLR                                5671
002040  000243                          .WORD CLV!CLC

002042  003767                          BLE     43$
002044  104431                          TRAP    31                              ;                                       5677
002046  010037  000000G                 MOV     R0,FREE.MEM.ADDR
002052  005002                  45$:    CLR     R2                              ; UNITS                                 5685
002054  005001                  46$:    CLR     R1                              ; COUNT                                 5686
002056  010100                  47$:    MOV     R1,R0                           ; COUNT,*                               5687
002060  060200                          ADD     R2,R0                           ; UNITS,*
002062  006300                          ASL     R0
002064  005060  000000G                 CLR     TALLY(R0)
002070  005201                          INC     R1                              ; COUNT                                 5686
002072  020127  000006                  CMP     R1,#6                           ; COUNT,*
002076  003767                          BLE     47$
002100  062702  000034                  ADD     #34,R2                          ; *,UNITS                               5685
002104  020227  000124                  CMP     R2,#124                         ; UNITS,*
002110  003761                          BLE     46$
002112  032705  000001                  BIT     #1,R5                           ; *,CLEAR.TABLES                        5689
002116  001421                          BEQ     50$
002120  005002                          CLR     R2                              ; UNITS                                 5691
002122  012701  000007          48$:    MOV     #7,R1                           ; *,COUNT                               5692
002126  010100                  49$:    MOV     R1,R0                           ; COUNT,*                               5693
002130  060200                          ADD     R2,R0                           ; UNITS,*
002132  006300                          ASL     R0
002134  005060  000000G                 CLR     TALLY(R0)
002140  005201                          INC     R1                              ; COUNT                                 5692
002142  020127  000033                  CMP     R1,#33                          ; COUNT,*
002146  003767                          BLE     49$
002150  062702  000034                  ADD     #34,R2                          ; *,UNITS                               5691
002154  020227  000124                  CMP     R2,#124                         ; UNITS,*
002160  003760                          BLE     48$
002162  006005                  50$:    ROR     R5                              : CLEAR.TABLES                          5695
002164  103011                          BCC     52$
```

```
002166  005000                          CLR     R0                      ; CTLR                  5697
002170  105060  000000G          51$:   CLRB    C.ERR.TBL(R0)           ; *(CTLR)               5699
002174  105060  000001G                 CLRB    C.ERR.TBL+1(R0)         ; *(CTLR)               5700
002200  062700  000002                  ADD     #2,R0                   ; *,CTLR                5697
002204  000243                          .WORD CLV!CLC

002206  003770                          BLE     51$
002210  005000                  52$:    CLR     R0                      ; CTLR                  5707
002212  105060  000000G          53$:   CLRB    QIO(R0)                 ; *(CTLR)               5708
002216  005200                          INC     R0                      ; CTLR                  5707
002220  000243                          .WORD CLV!CLC

002222  003773                          BLE     53$
002224  005037  000000G                 CLR     CRN.HIGH                ;                       5710
002230  005037  000000G                 CLR     CRN.LOW
002234  005000                          CLR     R0                      ;                       5712
002236  104441                          TRAP    41
002240  062706  000020                  ADD     #20,SP                  ;                       5451
002244  000207                          RTS     PC

; Routine Size:  595 words,     Routine Base:  $CODE$ + 1626
; Maximum stack depth per invocation:  20 words




                                        .SBTTL  L$INIT INITIALIZE SECTION
000000  004737  001626'         L$INIT::JSR     PC,LINIT                ;                       5712
000004  104411                          TRAP    11
000006  000207                          RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4074
; Maximum stack depth per invocation:  2 words
```

```
;          5715  %sbttl 'AUTODROP SECTION'
;          5716
;          5717  !+
;          5718  ! THIS CODE IS EXECUTED IMMEDIATELY AFTER THE INITIALIZE CODE IF
;          5719  ! THE "ADR" FLAG WAS SET.  THE UNIT(S) UNDER TEST ARE CHECKED TO
;          5720  ! SEE IF THEY WILL RESPOND.  THOSE THAT DON'T ARE IMMEDIATELY
;          5721  ! DROPPED FROM TESTING.
;          5722  !-
;          5723
;          5724  BGNAUTO;
;          5725
;          5726  !if (.SWP_FLAGS and SWF_TRC) eql SWF_TRC
;          5727  !then   PRINTF (DBM3);
;          5728
;          5729  ENDAUTO;
; INFO#219         L1:5714
; Empty compound expression is illegal
; Error occurred expanding macro ENDAUTO, called from source

                                        .SBTTL  LAUTO AUTODROP SECTION
000000  000207                  LAUTO:  RTS     PC                      ;       5714

; Routine Size:  1 word,        Routine Base:  $CODE$ + 4104
; Maximum stack depth per invocation:  0 words



                                        .SBTTL  L$AUTO AUTODROP SECTION
000000  004737  004104'         L$AUTO::JSR     PC,LAUTO                ;       5724
000004  104461                          TRAP    61
000006  000207                          RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4106
; Maximum stack depth per invocation:  2 words
```

```
;        5730  %sbttl 'CLEANUP CODING SECTION'
;        5731
;        5732  !+
;        5733  ! THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
;        5734  ! AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.
;        5735  !-
;        5736
;        5737  BGNCLN;
;        5738
;        5739  DORPT;
;        5740
;        5741  !CLRVEC (0_TVEC);                          ! RETURN ODT TRAP TO DIAGNOSTIC SUPERVISER
;        5742
;        5743  incr CTLR from 0 to (MAX_CTLR - 1) do      ! FOR EACH CONTROLLER
;        5744
;        5745      if (RDRX_ADDR = .CST [.CTLR, IP_ADDR]) neqa 0       ! IF CONTROLLER EXISTS
;        5746      then
;        5747          begin
;        5748          CLRVEC (.CST[.CTLR, VEC_ADDR]);     ! RETURN CONTROLLER'S TRAP VECTOR TO SUPERVISOR
;        5749          WRT_RDRX (RCIP, RC_ALL, ALL_ONES);  ! WRITE IP TO STOP DEVICE
;        5750          end;
;        5751
;        5752  ENDCLN;


                                        .SBTTL  LCLEAN CLEANUP CODING SECTION
000000  010146                  LCLEAN: MOV     R1,-(SP)                        ;                       5729
000002  104424                          TRAP    24                              ;                       5737
000004  005001                          CLR     R1                              ; CTLR                  5743
000006  016137  000000G 000000G  1$:     MOV     CST(R1),RDRX.ADDR               ; *(CTLR),*             5745
000014  001411                          BEQ     2$
000016  016100  000002G                  MOV     CST+2(R1),R0                    ; *(CTLR),*             5748
000022  042700  177000                   BIC     #177000,R0
000026  104436                          TRAP    36
000030  012700  177777                   MOV     #-1,R0                          ; *,RC.REG              5749
000034  010077  000000G                  MOV     R0,@RDRX.ADDR                   ; RC.REG,*
000040  062701  000056          2$:     ADD     #56,R1                          ; *,CTLR                5743
000044  000243                          .WORD CLV!CLC

000046  003757                          BLE     1$
000050  012601                          MOV     (SP)+,R1                        ;                       5729
000052  000207                          RTS     PC

; Routine Size:  22 words,      Routine Base:  $CODE$ + 4116
; Maximum stack depth per invocation:  3 words



                                        .SBTTL  L$CLEAN CLEANUP CODING SECTION
000000  004737  004116'         L$CLEAN::
                                        JSR     PC,LCLEAN                       ;                       5750
000004  104412                          TRAP    12
```

```
000006  000207                                  RTS     PC

; Routine Size:  4 words,      Routine Base:  $CODE$ + 4172
; Maximum stack depth per invocation:  2 words
```

```
;       5753  %sbttl 'DROP UNIT SECTION'
;       5754
;       5755  !+
;       5756  ! THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
;       5757  ! TO NO LONGER BE TESTED.
;       5758  !-
;       5759
;       5760  BGNDU;
;       5761
;       5762  local
;       5763      UNIT : word,                                          ! NUMBER OF UNIT BEING DROPPED
;       5764      PRINT : byte initial (byte (FALSE));                  ! NO PRINTING OF DROP UNIT MESSAGE
;       5765
;       5766  label
;       5767      SEARCH;
;       5768
;       5769  begin
;       5770
;       5771  register
;       5772      INPUT = 0;                                            ! UNIT NUMBER APPEARS IN R0 UPON ENTRY
;       5773
;       5774  UNIT = .INPUT;                                           ! GET UNIT NUMBER
;       5775  end;                                                     ! UNDECLARE REGISTER
;       5776
;       5777  if (.SWP_FLAGS and SWF_TRC) eql SWF_TRC
;       5778  then
;       5779      PRINTF (DBM5, .UNIT);
;       5780
;       5781  SEARCH :                                                 ! BEGIN SEARCH BLOCK
;       5782  begin
;       5783
;       5784  incr CTLR from 0 to (MAX_CTLR - 1) do                    ! FOR EACH CONTROLLER
;       5785
;       5786      incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do     ! FOR EACH UNIT ENTRY IN CST
;       5787
;       5788          if (.CST [.CTLR, .OFFSET, D_UNIT] eql .UNIT) and                        ! IF UNIT MATCHES CST ENTRY
;       5789             (.CST [.CTLR, .OFFSET, D_PRES] eql PRESENT)
;       5790          then
;       5791              begin
;       5792
;       5793              if (.CST [.CTLR, .OFFSET, D_STAT] eql ONLINE) or    ! IF UNIT IS STILL ALIVE
;       5794                 (.DUR [.UNIT] eql DU_ONLINE) or
;       5795                 (.DUR [.UNIT] eql DU_PROTECT)
;       5796              then
;       5797                  begin
;       5798                  PRINT = TRUE;                                   ! O.K. TO PRINT MESSAGE
;       5799
;       5800                  if (.CST [.CTLR, U_CNT] gtru 0) and
;       5801                     (.CST [.CTLR, .OFFSET, D_STAT] eql ONLINE)
;       5802                  then
;       5803                      CST [.CTLR, U_CNT] = .CST [.CTLR, U_CNT] - 1;! DECREMENT UNIT COUNT
;       5804
;       5805                  if (.CST [.CTLR, U_CNT] eql 0) and
```

```
;       5806                        (.CST [.CTLR, .OFFSET, D_STAT] eql ONLINE)
;       5807                    then
;       5808                        EOP_FLAG = TRUE;                    ! DECLARE END OF PASS IF ALL UNITS OFFLINE
;       5809
;       5810                    CST [.CTLR, .OFFSET, D_STAT] = OFFLINE; ! MARK UNIT OFFLINE
;       5811                    end;                                    ! IF UNIT WAS STILL ALIVE
;       5812
;       5813                leave SEARCH;                               ! EXIT SEARCH BLOCK
;       5814                end;                                        ! IF UNIT FOUND IN CST
;       5815
;       5816    end;
;       5817
;       5818    if .PRINT or                                            ! IF OK TO PRINT MESSAGE
;       5819       (.DUR [.UNIT] eql DU_CONF) or
;       5820       (.DUR [.UNIT] eql DU_INIT) or
;       5821       (.DUR [.UNIT] eql DU_ONLINE) or
;       5822       (.DUR [.UNIT] eql DU_AV) or
;       5823       (.DUR [.UNIT] eql DU_PROTECT)
;       5824    then
;       5825        begin
;       5826        PRINTF (DU_MSG, .UNIT);                             ! "UNIT XX DROPPED - "
;       5827        PRINTF (.DU_RSN [.DUR [.UNIT]]);                    ! REASON
;       5828        end;
;       5829
;       5830    ENDDU;

                                .SBTTL  LDU DROP UNIT SECTION
000000  004137  000000G   LDU:  JSR     R1,$SAVE5                       ;                       5752
000004  024646                 CMP     -(SP),-(SP)
000006  105004                 CLRB    R4                              ; PRINT
000010  010003                 MOV     R0,R3                           ; INPUT,UNIT            5774
000012  032737  000001  000000G BIT     #1,SWP.FLAGS                    ;                       5777
000020  001411                 BEQ     1$
000022  010346                 MOV     R3,-(SP)                        ; UNIT,*                5779
000024  012746  000000G        MOV     #DBM5,-(SP)
000030  012746  000002         MOV     #2,-(SP)
000034  010600                 MOV     SP,R0                           ; SP,*
000036  104417                 TRAP    17
000040  062706  000006         ADD     #6,SP
000044  005005           1$:   CLR     R5                              ; CTLR                  5784
000046  010546           2$:   MOV     R5,-(SP)                        ; CTLR,*                5788
000050  012746  000027         MOV     #27,-(SP)
000054  004737  000000G        JSR     PC,BL$MUL
000060  010066  000006         MOV     R0,6(SP)
000064  012701  000003         MOV     #3,R1                           ; *,OFFSET              5786
000070  010100           3$:   MOV     R1,R0                           ; OFFSET,*              5788
000072  066600  000006         ADD     6(SP),R0
000076  006300                 ASL     R0
000100  012766  000000G 000004 MOV     #CST,4(SP)
000106  060066  000004         ADD     R0,4(SP)
000112  010302                 MOV     R3,R2                           ; UNIT,*
000114  017600  000004         MOV     @4(SP),R0
```

```
000120  000300                          SWAB    R0
000122  042700  177760                  BIC     #177760,R0
000126  020002                          CMP     R0,R2
000130  001060                          BNE     9$
000132  032776  040000  000004          BIT     #40000,@4(SP)                   ;                               5789
000140  001454                          BEQ     9$
000142  005002                          CLR     R2                              ;                               5793
000144  032776  020000  000004          BIT     #20000,@4(SP)
000152  001402                          BEQ     4$
000154  005202                          INC     R2
000156  000410                          BR      5$
000160  126327  000000G 000007  4$:     CMPB    DUR(R3),#7                      ; *(UNIT),*                     5794
000166  001404                          BEQ     5$
000170  126327  000000G 000011          CMPB    DUR(R3),#11                     ; *(UNIT),*                     5795
000176  001033                          BNE     8$
000200  112704  000001          5$:     MOVB    #1,R4                           ; *,PRINT                       5798
000204  010516                          MOV     R5,(SP)                         ; CTLR,*                        5800
000206  012746  000056                  MOV     #56,-(SP)
000212  004737  000000G                 JSR     PC,BL$MUL
000216  005726                          TST     (SP)+
000220  062700  000004G                 ADD     #CST+4,R0
000224  105760  000001                  TSTB    1(R0)
000230  001404                          BEQ     6$
000232  006002                          ROR     R2                              ;                               5801
000234  105660  000001                  SBCB    1(R0)                           ;                               5803
000240  001007                          BNE     7$                              ;                               5805
000242  032776  020000  000004  6$:     BIT     #20000,@4(SP)                   ;                               5806
000250  001403                          BEQ     7$
000252  112737  000001  000000G         MOVB    #1,EOP.FLAG                     ;                               5808
000260  042776  020000  000004  7$:     BIC     #20000,@4(SP)                   ;                               5810
000266  022626                  8$:     CMP     (SP)+,(SP)+                     ;                               5791
000270  000411                          BR      10$
000272  062701  000005          9$:     ADD     #5,R1                           ; *,OFFSET                      5786
000276  020127  000022                  CMP     R1,#22                          ; OFFSET,*
000302  003672                          BLE     3$
000304  022626                          CMP     (SP)+,(SP)+
000306  005205                          INC     R5                              ; CTLR                          5784
000310  000243                          .WORD CLV!CLC

000312  003655                          BLE     2$
000314  006004                  10$:    ROR     R4                              ; PRINT                         5818
000316  103424                          BLO     11$
000320  126327  000000G 000001          CMPB    DUR(R3),#1                      ; *(UNIT),*                     5819
000326  001420                          BEQ     11$
000330  126327  000000G 000002          CMPB    DUR(R3),#2                      ; *(UNIT),*                     5820
000336  001414                          BEQ     11$
000340  126327  000000G 000007          CMPB    DUR(R3),#7                      ; *(UNIT),*                     5821
000346  001410                          BEQ     11$
000350  126327  000000G 000013          CMPB    DUR(R3),#13                     ; *(UNIT),*                     5822
000356  001404                          BEQ     11$
000360  126327  000000G 000011          CMPB    DUR(R3),#11                     ; *(UNIT),*                     5823
000366  001023                          BNE     12$
000370  010346                  11$:    MOV     R3,-(SP)                        ; UNIT,*                        5826
```

```
000372  012746  000000G                        MOV     #DU.MSG,-(SP)
000376  012746  000002                        MOV     #2,-(SP)
000402  010600                                MOV     SP,R0                   ; SP,*
000404  104417                                TRAP    17                              5827
000406  005000                                CLR     R0                      ;
000410  156300  000000G                        BISB    DU.R(R3),R0             ; *(UNIT),*
000414  006300                                ASL     R0
000416  016016  000000G                        MOV     DU.RSN(R0),(SP)
000422  012746  000001                        MOV     #1,-(SP)
000426  010600                                MOV     SP,R0                   ; SP,*
000430  104417                                TRAP    17
000432  062706  000010                        ADD     #10,SP                  ;               5825
000436  022626                        12$:    CMP     (SP)+,(SP)+             ;               5752
000440  000207                                RTS     PC

; Routine Size:  145 words,     Routine Base:  $CODE$ + 4202
; Maximum stack depth per invocation:  14 words



                                              .SBTTL  L$DU DROP UNIT SECTION
000000  004737  004202'                L$DU::  JSR     PC,I.DU                 ;               5828
000004  104453                                TRAP    53
000006  000207                                RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4644
; Maximum stack depth per invocation:  2 words
```

```
;          5831  %sbttl 'ADD UNIT SECTION'
;          5832
;          5833  !+
;          5834  ! THE ADD-UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
;          5835  !  TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
;          5836  !  TO THE TEST CYCLE.
;          5837  !-
;          5838
;          5839  BGNAU;
;          5840
;          5841  local
;          5842      STINDX : word,
;          5843      ENDIDX : word;
;          5844
;          5845  register
;          5846      UNIT = 0;                                   ! UNIT NUMBER APPEARS IN R0 UPON ENTRY
;          5847
;          5848  if (.SWP_FLAGS and SWF_CST) eql SWF_CST
;          5849  then
;          5850      begin                                       ! IF CLEAR STAT. TABLES TRUE....
;          5851      STINDX = .UNIT * TALLY_LEN;                 ! ZERO OUT
;          5852      ENDIDX = .STINDX + TALLY_LEN - 1;           !    ADDED
;          5853
;          5854      incr COUNT from .STINDX to .ENDIDX do       !    UNIT'S
;          5855          TALLY [.COUNT] = 0;                     !    STATISTICS
;          5856
;          5857      end;
;          5858
;          5859  ENDAU;


                                        .SBTTL  LAU ADD UNIT SECTION
000000  004137  000000G          LAU:   JSR     R1,$SAVE2            ;                          5830
000004  105737  000000G                 TSTB    SWP.FLAGS            ;                          5848
000010  100023                          BPL     3$
000012  010046                          MOV     R0,-(SP)             ; UNIT,*                   5851
000014  012746  000034                  MOV     #34,-(SP)
000020  004737  000000G                 JSR     PC,BL$MUL
000024  010002                          MOV     R0,R2                ; STINDX,ENDIDX            5852
000026  062702  000033                  ADD     #33,R2               ; *,ENDIDX
000032  010001                          MOV     R0,R1                ; STINDX,COUNT             5854
000034  005301                          DEC     R1                   ; COUNT
000036  000404                          BR      2$
000040  010100                  1$:     MOV     R1,R0                ; COUNT,*                  5855
000042  006300                          ASL     R0
000044  005060  000000G                 CLR     TALLY(R0)
000050  005201                  2$:     INC     R1                   ; COUNT                    5854
000052  020102                          CMP     R1,R2                ; COUNT,ENDIDX
000054  003771                          BLE     1$
000056  022626                          CMP     (SP)+,(SP)+          ;                          5850
000060  000207                  3$:     RTS     PC                   ;                          5830

; Routine Size:  25 words,      Routine Base:  $CODE$ + 4654
```

```
; Maximum stack depth per invocation:  6 words



                                        .SBTTL  L$AU ADD UNIT SECTION
000000  004737  004654'          L$AU::  JSR     PC,LAU                  ;                5857
000004  104452                          TRAP    52
000006  000207                          RTS     PC

; Routine Size:  4 words,      Routine Base:  $CODE$ + 4736
; Maximum stack depth per invocation:  2 words
```

```
;          5860  %sbttl 'NON-EXISTENT MEMORY TRAP HANDLER'
;          5861
;          5862  !+
;          5863  !       THIS TRAP HANDLER IS VECTORED FROM LOCATION 4 FOR ALL UNIBUS TIMEOUT
;          5864  !       ERRORS, INDICATING THAT AN ATTEMPT WAS MADE TO REFERENCE A NON-EXISTENT
;          5865  !       MEMORY LOCATION. ITS MAIN PURPOSE IS TO SET A FLAG FOR THE RDRX
;          5866  !       REGISTER EXISTENCE TEST, INDICATING THE ABSENCE OF A DEVICE REGISTER.
;          5867  !-
;          5868
;          5869  BGNSRV (NEX_TRAP);
;          5870
;          5871  NEX = TRUE;                                   ! NEX TRAP OCCURRED
;          5872
;          5873  ENDSRV;


                                        .SBTTL  NEX.TRAP NON-EXISTENT MEMORY TRAP HANDLER
000000  012737  000001  000000G  NEX.TRAP::
                                        MOV     #1,NEX                          ;  5871
000006  000002                          RTI                                     ;  5869

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4746
; Maximum stack depth per invocation:  0 words
```

```
;        5874  %sbttl 'GLOBAL ROUTINES'
;        5875
;        5876  global routine SET_CPAR (CTLR) : novalue =
;        5877
;        5878  !+
;        5879  !       THIS ROUTINE SETS UP THE COMMONLY-USED CONTROLLER-RELATED DATA ITEMS
;        5880  !       FOR THE GIVEN CONTROLLER NUMBER.
;        5881  !
;        5882  !       INPUTS:
;        5883  !               CTLR - CONTROLLER NUMBER
;        5884  !
;        5885  !       IMPLICIT OUTPUTS:
;        5886  !               CCTLR - CURRENT CONTROLLER NUMBER
;        5887  !               CST_ADDR - ADDRESS OF CONTROLLER'S STATUS TABLE
;        5888  !               DCT_ADDR - ADDRESS OF CONTROLLER'S DRIVER TABLE
;        5889  !               RDRX_ADDR - ADDRESS OF CONTROLLER'S IP REGISTER
;        5890  !-
;        5891
;        5892      begin
;        5893      CCTLR = .CTLR;                          ! SET CURRENT CONTROLLER NUMBER
;        5894      CST_ADDR = CST + (.CTLR * CST_LEN * 2); ! CALCULATE ADDRESS OF CONTROLLER'S CST
;        5895      DCT_ADDR = DCT + (.CTLR * DCT_LEN * 2); ! CALCULATE ADDRESS OF CONTROLLER'S DCT
;        5896      RDRX_ADDR = .CST_ADDR [IP_ADDR];        ! GET CONTROLLER'S DEVICE ADDRESS
;        5897      end;

                                .SBTTL  SET.CPAR GLOBAL ROUTINES
000000  010146                SET.CPAR::
                                MOV     R1,-(SP)                        ;                       5876
000002  016601  000004          MOV     4(SP),R1                        ; CTLR,*                5893
000006  010137  000000G         MOV     R1,CCTLR
000012  010146                  MOV     R1,-(SP)                        ;                       5894
000014  012746  000056          MOV     #56,-(SP)
000020  004737  000000G         JSR     PC,BL$MUL
000024  062700  000000G         ADD     #CST,R0
000030  010037  000000G         MOV     R0,CST.ADDR
000034  010116                  MOV     R1,(SP)                         ;                       5895
000036  012746  000022          MOV     #22,-(SP)
000042  004737  000000G         JSR     PC,BL$MUL
000046  062700  000000G         ADD     #DCT,R0
000052  010037  000000G         MOV     R0,DCT.ADDR
000056  017737  000000G 000000G  MOV     @CST.ADDR,RDRX.ADDR             ;                       5896
000064  062706  000006          ADD     #6,SP                           ;                       5892
000070  012601                  MOV     (SP)+,R1                        ;                       5876
000072  000207                  RTS     PC

; Routine Size:  30 words,      Routine Base:  $CODE$ + 4756
; Maximum stack depth per invocation:  5 words
```

```
;          5898  global routine SET_UPAR (OFFSET) : novalue =
;          5899
;          5900  !+
;          5901  !       THIS ROUTINE SETS UP THE COMMONLY-USED UNIT-RELATED DATA ITEMS FOR
;          5902  !       THE CURRENT CONTROLLER AND GIVEN CST OFFSET.
;          5903  !
;          5904  !       INPUTS:
;          5905  !               OFFSET - WORD OFFSET INTO CURRENT CONTROLLER'S CST WHICH
;          5906  !                        DESCRIBES A UNIT
;          5907  !
;          5908  !       IMPLICIT INPUTS:
;          5909  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;          5910  !
;          5911  !       IMPLICIT OUTPUTS:
;          5912  !               CUOFF - CURRENT UNIT'S CST OFFSET
;          5913  !               CDISK - CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;          5914  !               L$LUN - CURRENT UNIT NUMBER (DRS UNIT NUMBER)
;          5915  !               T_ADDR - ADDRESS OF CURRENT UNIT'S STATISTICS BLOCK (TALLY)
;          5916  !-
;          5917
;          5918      begin
;          5919      CUOFF = .OFFSET;
;          5920      CDISK = .CST_ADDR [.OFFSET, D_DISK_NUM];
;          5921      L$LUN = .CST_ADDR [.OFFSET, D_UNIT];
;          5922      T_ADDR = TALLY + (.L$LUN * TALLY_LEN * 2);
;          5923      end;

                                        .SBTTL  SET.UPAR GLOBAL ROUTINES
000000  010146                  SET.UPAR::
                                        MOV     R1,-(SP)                ;                       5898
000002  016637  000004  000000G         MOV     4(SP),CUOFF             ; OFFSET,*              5919
000010  016600  000004                  MOV     4(SP),R0                ; CUOFF,*               5920
000014  006300                          ASL     R0
000016  063700  000000G                 ADD     CST.ADDR,R0
000022  111037  000000G                 MOVB    (R0),CDISK
000026  042737  177774  000000G         BIC     #177774,CDISK
000034  011001                          MOV     (R0),R1                 ;                       5921
000036  000301                          SWAB    R1
000040  042701  177760                  BIC     #177760,R1
000044  010137  000000G                 MOV     R1,L$LUN
000050  010146                          MOV     R1,-(SP)                ; L$LUN,*               5922
000052  012746  000070                  MOV     #70,-(SP)
000056  004737  000000G                 JSR     PC,BL$MUL
000062  062700  000000G                 ADD     #TALLY,R0
000066  010037  000000G                 MOV     R0,T.ADDR
000072  022626                          CMP     (SP)+,(SP)+             ;                       5918
000074  012601                          MOV     (SP)+,R1                ;                       5898
000076  000207                          RTS     PC

; Routine Size:  32 words,      Routine Base:  $CODE$ + 5052
; Maximum stack depth per invocation:  4 words
;          5924
```

```
;         5925
;         5926  global routine GET_PKT (CTLR) =
;         5927
;         5928  !+
;         5929  !       THIS ROUTINE SEARCHES THE MSCP PACKET POOL ALLOCATION TABLE (PKT_USE)
;         5930  !       FOR A FREE MSCP PACKET TO ALLOCATE TO THE GIVEN CONTROLLER. IF ONE IS
;         5931  !       FOUND, THE PACKET IS ZEROED OUT, AND THE PACKET INDEX IS RETURNED
;         5932  !       TO THE CALLER. OTHERWISE, A -1 IS RETURNED INDICATING NONE AVAILABLE.
;         5933  !
;         5934  !       INPUTS:
;         5935  !               CTLR - CONTROLLER NUMBER REQUESTING ALLOCATION
;         5936  !-
;         5937
;         5938      begin
;         5939
;         5940      local
;         5941          index : signed word initial (-1),
;         5942          RING_ADDR : word,
;         5943          PACKET_OWNED : byte,
;         5944          NEXT_PACKET : byte;
;         5945
;         5946      NEXT_PACKET = .NEXT_PKT_USE;                ! NEXT PACKET TO TRY
;         5947
;         5948      incr COUNT from 0 to (PKT_CNT - 1) do       ! FOR EACH ENTRY IN ALLOCATION TABLE
;         5949          begin
;         5950          PACKET_OWNED = FALSE;
;         5951
;         5952          if .PKT_USE [.NEXT_PACKET] lss 0        ! IF ENTRY INDICATES FREE PACKET
;         5953          then
;         5954              begin
;         5955              RING_ADDR = .DCT_ADDR [RR_BEG];     ! FIRST RESPONSE PACKET'S ADDRESS
;         5956
;         5957              incr I from 1 to (RRING_LEN + CRING_LEN) do ! FOR EACH PACKET ADDRESS
;         5958                  begin
;         5959
;         5960                  if (..RING_ADDR eqla .MSCP_PKT [.NEXT_PACKET, PKT_LO]) and
;         5961                     (((.RING_ADDR + 2) and ED_OWN) eql ED_OWN)
;         5962                  then
;         5963                      begin                       ! CHECK ADDRESS AND OWNERSHIP
;         5964                      PACKET_OWNED = TRUE;        ! PACKET OWNED BY CONTROLLER
;         5965                      exitloop;
;         5966                      end
;         5967                  else
;         5968                      RING_ADDR = .RING_ADDR + 4; ! ADDRESS OF NEXT PACKET IN RING
;         5969
;         5970                  end;
;         5971
;         5972              if not .PACKET_OWNED                ! IF NOT ALREADY USED
;         5973              then
;         5974                  begin
;         5975                  PKT_USE [.NEXT_PACKET] = .CTLR; ! ALLOCATE PACKET TO CONTROLLER
;         5976                  index = .NEXT_PACKET;
;         5977
```

```
;          5978                    incr J from 2 to (PKT_LEN - 1) do       ! ZERO OUT PACKET
;          5979                        MSCP_PKT [.NEXT_PACKET, .J, 0, 16, 0] = 0;
;          5980
;          5981                    exitloop;                               ! DONE
;          5982                    end;
;          5983
;          5984                end;
;          5985
;          5986            NEXT_PACKET = .NEXT_PACKET + 1;                 ! TRY NEXT PACKET IN RING
;          5987
;          5988            if .NEXT_PACKET gequ PKT_CNT
;          5989            then
;          5990                NEXT_PACKET = 0;                            ! IF BEYOND ALL PACKETS, START AT THE TOP
;          5991
;          5992            end;
;          5993
;          5994        if (.index geq 0) and                               ! IF PACKET FOUND
;          5995           (.PKT_USE [.index] geq 0)
;          5996        then
;          5997            begin
;          5998            MSCP_PKT [.index, MSGLEN] = SZ_GEN;             ! PACKET SIZE - ONLY ONLINE AND SCC CHANGE IT
;          5999            MSCP_PKT [.index, CREDITS] = 1;                 ! CREDIT SIZE
;          6000            NEXT_PKT_USE = .NEXT_PACKET + 1;                ! NEXT PACKET TO ALLOCATE
;          6001
;          6002            if.NEXT_PKT_USE gequ PKT_CNT
;          6003            then
;          6004                NEXT_PKT_USE = 0;                           ! CYCLE BACK TO BEGINNING IF AT END
;          6005
;          6006            end;
;          6007
;          6008        return .index;
;          6009        end;


                                        .SBTTL  GET.PKT GLOBAL ROUTINES
000000  004137  000000G                GET.PKT::
                                        JSR     R1,$SAVE5               ;                                       5926
000004  162706  000006                  SUB     #6,SP
000010  012704  177777                  MOV     #-1,R4                  ; *,INDEX                               5938
000014  113705  000000G                 MOVB    NEXT.PKT.USE,R5         ; *,NEXT.PACKET                         5946
000020  012766  000014  000004          MOV     #14,4(SP)               ; *,COUNT                               5948
000026  105066  000002          1$:     CLRB    2(SP)                   ; PACKET.OWNED                          5950
000032  005002                          CLR     R2                      ;                                       5952
000034  150502                          BISB    R5,R2                   ; NEXT.PACKET,*
000036  105762  000000G                 TSTB    PKT.USE(R2)
000042  002076                          BGE     7$
000044  013700  000000G                 MOV     DCT.ADDR,R0             ;                                       5955
000050  016016  000004                  MOV     4(R0),(SP)              ; *,RING.ADDR
000054  010246                          MOV     R2,-(SP)                ;                                       5960
000056  012746  000104                  MOV     #104,-(SP)
000062  004737  000000G                 JSR     PC,BL$MUL
000066  012703  000010                  MOV     #10,R3                  ; *,I                                   5957
000072  027660  000004  000000G  2$:     CMP     @4(SP),MSCP.PKT(R0)     ; RING.ADDR,*                           5960
```

```
000100  001016                          BNE     3$
000102  012746  000002                  MOV     #2,-(SP)                                        ;                                               5961
000106  066616  000006                  ADD     6(SP),(SP)                                      ; RING.ADDR,*
000112  012601                          MOV     (SP)+,R1
000114  042701  077777                  BIC     #77777,R1
000120  020127  100000                  CMP     R1,#-100000
000124  001004                          BNE     3$
000126  112766  000001  000006          MOVB    #1,6(SP)                                        ; *,PACKET.OWNED                                5964
000134  000405                          BR      4$                                              ;                                               5963
000136  062766  000004  000004  3$:     ADD     #4,4(SP)                                        ; *,RING.ADDR                                   5968
000144  005303                          DEC     R3                                              ; I                                             5957
000146  001351                          BNE     2$
000150  032766  000001  000006  4$:     BIT     #1,6(SP)                                        ; *,PACKET.OWNED                                5972
000156  001027                          BNE     6$
000160  116662  000030  000000G         MOVB    30(SP),PKT.USE(R2)                              ; CTLR,*                                        5975
000166  010204                          MOV     R2,R4                                           ; *,INDEX                                       5976
000170  010216                          MOV     R2,(SP)                                         ;                                               5979
000172  012746  000042                  MOV     #42,-(SP)
000176  004737  000000G                 JSR     PC,BL$MUL
000202  005726                          TST     (SP)+
000204  012701  000002                  MOV     #2,R1                                           ; *,J                                           5978
000210  010003                  5$:     MOV     R0,R3                                           ;                                               5979
000212  060103                          ADD     R1,R3                                           ; J,*
000214  006303                          ASL     R3
000216  005063  000000G                 CLR     MSCP.PKT(R3)
000222  005201                          INC     R1                                              ; J                                             5978
000224  020127  000041                  CMP     R1,#41                                          ; J,*
000230  003767                          BLE     5$
000232  022626                          CMP     (SP)+,(SP)+                                     ;                                               5974
000234  000411                          BR      9$
000236  022626                  6$:     CMP     (SP)+,(SP)+                                     ;                                               5954
000240  105205                  7$:     INCB    R5                                              ; NEXT.PACKET                                   5986
000242  120527  000014                  CMPB    R5,#14                                          ; NEXT.PACKET,*                                 5988
000246  103401                          BLO     8$
000250  105005                          CLRB    R5                                              ; NEXT.PACKET                                   5990
000252  005366  000004          8$:     DEC     4(SP)                                           ; COUNT                                         5948
000256  001263                          BNE     1$
000260  005704                  9$:     TST     R4                                              ; INDEX                                         5994
000262  002434                          BLT     11$
000264  105764  000000G                 TSTB    PKT.USE(R4)                                     ; *(INDEX)                                      5995
000270  002431                          BLT     11$
000272  010446                          MOV     R4,-(SP)                                        ; INDEX,*                                       5998
000274  012746  000104                  MOV     #104,-(SP)
000300  004737  000000G                 JSR     PC,BL$MUL
000304  012760  000040  000006G         MOV     #40,MSCP.PKT+6(R0)
000312  142760  000017  000010G         BICB    #17,MSCP.PKT+10(R0)                             ;                                               5999
000320  152760  000001  000010G         BISB    #1,MSCP.PKT+10(R0)
000326  005000                          CLR     R0                                              ;                                               6000
000330  150500                          BISB    R5,R0                                           ; NEXT.PACKET,*
000332  005200                          INC     R0
000334  110037  000000G                 MOVB    R0,NEXT.PKT.USE
000340  120027  000014                  CMPB    R0,#14                                          ; NEXT.PKT.USE,*                                6002
000344  103402                          BLO     10$
```

```
000346  105037  000000G                        CLRB    NEXT.PKT.USE                    ;                               6004
000352  022626                          10$:    CMP     (SP)+,(SP)+                     ;                               5997
000354  010400                          11$:    MOV     R4,R0                           ; INDEX,*                       5938
000356  062706  000006                        ADD     #6,SP                           ;                               5926
000362  000207                                RTS     PC

; Routine Size:  122 words,    Routine Base:  $CODE$ + 5152
; Maximum stack depth per invocation:  13 words


;        6010
```

```
;        6011  global routine PUT_PKT (index) : novalue =
;        6012
;        6013  !+
;        6014  !       THE MSCP PACKET DESIGNATED BY "INDEX" IS RETURNED TO THE POOL BY THIS
;        6015  !       ROUTINE.
;        6016  !-
;        6017
;        6018      begin
;        6019
;        6020      local
;        6021          RING_ADDR : word,
;        6022          OWNER : word;
;        6023
;        6024      RING_ADDR = .DCT_ADDR [RR_BEG];                     ! ADDRESS IN FIRST RESPONSE RING
;        6025
;        6026      incr COUNT from 1 to (RRING_LEN + CRING_LEN) do     ! FOR EACH ADDRESS IN THE RINGS
;        6027          begin
;        6028
;        6029          if .MSCP_PKT [.index, PKT_LO] eqla ..RING_ADDR  ! IF ADDRESS MATCHES
;        6030          then
;        6031              begin
;        6032              OWNER = .RING_ADDR + 2;                     ! ADDRESS OF OWNERSHIP WORD
;        6033              .OWNER = ..OWNER and (not (ED_OWN)) and (not (ED_FLAG)) ; ! GIVE OWNERSHIP TO HOST
;        6034              end;
;        6035
;        6036          RING_ADDR = .RING_ADDR + 4;                     ! LOOK AT NEXT PACKET ADDRESS IN RING
;        6037          end;
;        6038
;        6039      PKT_USE [.index] = -1;
;        6040      end;


                                .SBTTL  PUT.PKT GLOBAL ROUTINES
000000  004137  000000G    PUT.PKT::
                            JSR     R1,$SAVE4                   ;                               6011
000004  013700  000000G            MOV     DCT.ADDR,R0         ;                               6024
000010  016003  000004             MOV     4(R0),R3            ; *,RING.ADDR
000014  016601  000014             MOV     14(SP),R1           ; INDEX,*                       6029
000020  010146                     MOV     R1,-(SP)
000022  012746  000104             MOV     #104,-(SP)
000026  004737  000000G            JSR     PC,BL$MUL
000032  012702  000010             MOV     #10,R2              ; *,COUNT                       6026
000036  026013  000000G    1$:     CMP     MSCP.PKT(R0),(R3)   ; *,RING.ADDR                   6029
000042  001005                     BNE     2$
000044  012704  000002             MOV     #2,R4               ; *,OWNER                       6032
000050  060304                     ADD     R3,R4               ; RING.ADDR,OWNER
000052  042714  140000             BIC     #140000,(R4)        ; *,OWNER                       6033
000056  062703  000004    2$:     ADD     #4,R3               ; *,RING.ADDR                   6036
000062  005302                     DEC     R2                  ; COUNT                         6026
000064  001364                     BNE     1$
000066  112761  000377  000000G    MOVB    #377,PKT.USE(R1)    ;                               6039
000074  022626                     CMP     (SP)+,(SP)+         ;                               6018
000076  000207                     RTS     PC                  ;                               6011
```

```
; Routine Size:  32 words,      Routine Base:  $CODE$ + 5536
; Maximum stack depth per invocation:  8 words
```

```
;         6041  routine PUTA_PKT (CTLR) : novalue =
;         6042
;         6043  !+
;         6044  !       THIS ROUTINE DEALLOCATES ALL MSCP PACKETS WHICH HAVE BEEN ALLOCATED
;         6045  !       TO A PARTICULAR CONTROLLER.
;         6046  !
;         6047  !       INPUTS:
;         6048  !               CTLR - CONTROLLER NUMBER
;         6049  !-
;         6050
;         6051      incr COUNT from 0 to (PKT_CNT - 1) do       ! FOR EACH ENTRY IN ALLOCATION TABLE
;         6052
;         6053          if .PKT_USE [.COUNT] eql .CTLR          ! IF PACKET IS ALLOCATED TO GIVEN CONTROLLER
;         6054          then
;         6055              PKT_USE [.COUNT] = -1;              ! DEALLOCATE IT


                                        .SBTTL  PUTA.PKT GLOBAL ROUTINES
000000  010146                  PUTA.PKT:
                                        MOV     R1,-(SP)                ;                               6041
000002  005000                          CLR     R0                      ; COUNT                         6051
000004  116001  000000G         1$:     MOVB    PKT.USE(R0),R1          ; *(COUNT),*                    6053
000010  020166  000004                  CMP     R1,4(SP)                ; *,CTLR
000014  001003                          BNE     2$
000016  112760  000377  000000G        MOVB    #377,PKT.USE(R0)        ; *,*(COUNT)                    6055
000024  005200                  2$:     INC     R0                      ; COUNT                         6051
000026  020027  000013                  CMP     R0,#13                  ; COUNT,*
000032  003764                          BLE     1$
000034  012601                          MOV     (SP)+,R1                ;                               6041
000036  000207                          RTS     PC

; Routine Size:  16 words,      Routine Base:  $CODE$ + 5636
; Maximum stack depth per invocation:  2 words
```

```
;        6056  global routine GET_RETPKT (CTLR) =
;        6057
;        6058  !+
;        6059  !       THIS ROUTINE SEARCHES THE RETURN PACKET POOL ALLOCATION TABLE (RP_USE)
;        6060  !       FOR A FREE RETURN PACKET TO ALLOCATE TO THE GIVEN CONTROLLER. IF ONE IS
;        6061  !       FOUND, THE PACKET IS ZEROED OUT, AND THE PACKET INDEX IS RETURNED TO
;        6062  !       THE CALLER. OTHERWISE, A -1 IS RETURNED INDICATING NONE AVAILABLE.
;        6063  !
;        6064  !       INPUTS:
;        6065  !               CTLR - CONTROLLER NUMBER REQUESTING ALLOCATION
;        6066  !-
;        6067
;        6068      begin
;        6069
;        6070      local
;        6071          index : signed word initial (-1);       ! ASSUME NONE AVAILABLE
;        6072
;        6073      incr COUNT from 0 to (RP_CNT - 1) do        ! FOR EACH ENTRY IN TABLE
;        6074
;        6075          if .RP_USE [.COUNT] lss 0               ! IF FREE RETPKT IS FOUND
;        6076          then
;        6077              begin
;        6078              RP_USE [.COUNT] = .CTLR;            ! ALLOCATE RETURN PACKET TO CONTROLLER
;        6079              index = .COUNT;
;        6080
;        6081              incr J from 0 to (RP_LEN - 1) do    ! ZERO OUT RETPKT
;        6082                  RETPKT [.COUNT, .J, 0, 16, 0] = 0;
;        6083
;        6084              exitloop;                           ! DONE
;        6085              end;
;        6086
;        6087      return .index;                              ! RETURN PACKET INDEX (OR -1) TO CALLER
;        6088      end;


                                        .SBTTL  GET.RETPKT GLOBAL ROUTINES
000000  004137  000000G          GET.RETPKT::
                                        JSR     R1,$SAVE4                       ;                       6056
000004  012704  177777                  MOV     #-1,R4                          ; *,INDEX               6068
000010  005003                          CLR     R3                              ; COUNT                 6073
000012  105763  000000G         1$:     TSTB    RP.USE(R3)                      ; *(COUNT)              6075
000016  002025                          BGE     3$
000020  116663  000014  000000G        MOVB    14(SP),RP.USE(R3)               ; CTLR,*(COUNT)         6078
000026  010304                          MOV     R3,R4                           ; COUNT,INDEX           6079
000030  010346                          MOV     R3,-(SP)                        ; COUNT,*               6082
000032  012746  000030                  MOV     #30,-(SP)
000036  004737  000000G                JSR     PC,BL$MUL
000042  022626                          CMP     (SP)+,(SP)+
000044  005002                          CLR     R2                              ; J                     6081
000046  010001                  2$:     MOV     R0,R1                           ;                       6082
000050  060201                          ADD     R2,R1                           ; J,*
000052  006301                          ASL     R1
000054  005061  000000G                CLR     RETPKT(R1)
```

```
000060  005202                          INC     R2                      ; J                             6081
000062  020227  000027                  CMP     R2,#27                  ; J,*
000066  003767                          BLE     2$
000070  000404                          BR      4$                      ;                               6077
000072  005203                  3$:     INC     R3                      ; COUNT                         6073
000074  020327  000003                  CMP     R3,#3                   ; COUNT,*
000100  003744                          BLE     1$
000102  010400                  4$:     MOV     R4,R0                   ; INDEX,*                       6068
000104  000207                          RTS     PC                      ;                               6056

; Routine Size:  35 words,      Routine Base:  $CODE$ + 5676
; Maximum stack depth per invocation:  8 words
```

```
;          6089  global routine PUT_RETPKT (index) : novalue =
;          6090
;          6091  !+
;          6092  !       THE RETURN PACKET DESIGNATED BY "INDEX" IS RETURNED TO THE POOL BY THIS
;          6093  !       ROUTINE.
;          6094  !-
;          6095
;          6096      RP_USE [.index] = -1;

                                        .SBTTL  PUT.RETPKT GLOBAL ROUTINES
000000  016600  000002          PUT.RETPKT::
                                        MOV     2(SP),R0                        ; INDEX,*                       6096
000004  112760  000377  000000G        MOVB    #377,RP.USE(R0)
000012  000207                          RTS     PC                              ;                               6089

; Routine Size:  6 words,       Routine Base:  $CODE$ + 6004
; Maximum stack depth per invocation:  0 words
```

```
;       6097  global routine GET_IO_BUFF (ADDR) : novalue =
;       6098
;       6099  !+
;       6100  !       THIS ROUTINE HANDLES THE ALLOCATION OF AN I/O BUFFER FROM THE BUFFER
;       6101  !       POOL.
;       6102  !
;       6103  !       INPUTS:
;       6104  !               ADDR - ADDRESS TO STORE THE 2-WORD BUFFER DESCRIPTOR
;       6105  !
;       6106  !       IMPLICIT INPUTS:
;       6107  !               CCTLR - CURRENT CONTROLLER NUMBER
;       6108  !
;       6109  !       OUTPUTS:
;       6110  !               THE ALLOCATED BUFFER'S DESCRIPTOR IS LOADED INTO THE TWO
;       6111  !               WORDS AT ''ADDR'' AND ''ADDR + 2''. OTHERWISE, A ZERO IS RETURNED
;       6112  !               AT ''ADDR'' IF NO BUFFERS ARE AVAILABLE.
;       6113  !-
;       6114
;       6115      begin
;       6116      .ADDR = 0;                                    ! ASSUME FAILURE
;       6117
;       6118      incr COUNT from 0 to (QIO_PER_CTLR * MAX_CTLR - 1) do       ! FOR EACH ENTRY IN BUFFER TABLE
;       6119
;       6120          if .BUFF_OWN [.COUNT] lss 0               ! IF BUFFER IS FREE
;       6121          then
;       6122              begin
;       6123              BUFF_OWN [.COUNT] = .CCTLR;           ! ALLOCATE BUFFER TO CONTROLLER
;       6124              .ADDR = .BUFF_ADDR [.COUNT];          ! RETURN BUFFER DESCRIPTOR
;       6125              exitloop;                             ! DONE
;       6126              end;
;       6127
;       6128      end;                                          ! ROUTINE GET_IO_BUFF


                                        .SBTTL  GET.IO.BUFF GLOBAL ROUTINES
000000  010146                  GET.IO.BUFF::
                                        MOV     R1,-(SP)                ;                               6097
000002  005076  000004                  CLR     @4(SP)                  ; ADDR                          6116
000006  005001                          CLR     R1                      ; COUNT                         6118
000010  105761  000000G         1$:     TSTB    BUFF.OWN(R1)            ; *(COUNT)                      6120
000014  002011                          BGE     2$
000016  113761  000000G 000000G       MOVB    CCTLR,BUFF.OWN(R1)      ; *,*(COUNT)                    6123
000024  010100                          MOV     R1,R0                   ; COUNT,*                       6124
000026  006300                          ASL     R0
000030  016076  000000G 000004          MOV     BUFF.ADDR(R0),@4(SP)    ; *,ADDR
000036  000404                          BR      3$                      ;                               6122
000040  005201                  2$:     INC     R1                      ; COUNT                         6118
000042  020127  000007                  CMP     R1,#7                   ; COUNT,*
000046  003760                          BLE     1$
000050  012601                  3$:     MOV     (SP)+,R1                ;                               6097
000052  000207                          RTS     PC

; Routine Size:  22 words,      Routine Base:  $CODE$ + 6020
```

```
; Maximum stack depth per invocation:  2 words
```

```
;        6129  global routine PUT_IO_BUFF (ADDR) : novalue =
;        6130
;        6131  !+
;        6132  !       THIS ROUTINE HANDLES THE DEALLOCATION OF AN I/O BUFFER, RETURNING IT
;        6133  !       TO THE BUFFER POOL.
;        6134  !
;        6135  !       INPUTS:
;        6136  !               ADDR - ADDRESS OF THE 2-WORD BUFFER DESCRIPTOR TO BE
;        6137  !                      DEALLOCATED
;        6138  !-
;        6139
;        6140      incr COUNT from 0 to (QIO_PER_CTLR * MAX_CTLR - 1) do       ! FOR EACH ENTRY IN BUFFER TABLE
;        6141
;        6142          if .BUFF_ADDR [.COUNT] eqla ..ADDR      ! IF THIS IS THE BUFFER'S ENTRY
;        6143          then
;        6144              begin
;        6145              BUFF_OWN [.COUNT] = -1;             ! DEALLOCATE BUFFER
;        6146              exitloop;                           ! DONE
;        6147              end;


                                .SBTTL  PUT.IO.BUFF GLOBAL ROUTINES
000000  010146            PUT.IO.BUFF::
                                MOV     R1,-(SP)                        ;                               6129
000002  005001                  CLR     R1                              ; COUNT                         6140
000004  010100          1$:     MOV     R1,R0                           ; COUNT,*                       6142
000006  006300                  ASL     R0
000010  026076  000000G 000004          CMP     BUFF.ADDR(R0),@4(SP)    ; *,ADDR
000016  001004                  BNE     2$
000020  112761  000377  000000G         MOVB    #377,BUFF.OWN(R1)       ; *,*(COUNT)                    6145
000026  000404                  BR      3$                              ;                               6144
000030  005201          2$:     INC     R1                              ; COUNT                         6140
000032  020127  000007          CMP     R1,#7                           ; COUNT,*
000036  003762                  BLE     1$
000040  012601          3$:     MOV     (SP)+,R1                        ;                               6129
000042  000207                  RTS     PC

; Routine Size:  18 words,      Routine Base:  $CODE$ + 6074
; Maximum stack depth per invocation:  2 words
```

```
;        6148  global routine PUTA_BUFF : novalue =
;        6149
;        6150  !+
;        6151  !       THIS ROUTINE DEALLOCATES ALL I/O BUFFERS WHICH HAVE BEEN ALLOCATED TO
;        6152  !       THE CURRENT CONTROLLER (CCTLR).
;        6153  !-
;        6154
;        6155      incr COUNT from 0 to (QIO_PER_CTLR * MAX_CTLR - 1) do       ! FOR EACH ENTRY IN BUFFER TABLE
;        6156
;        6157          if .BUFF_OWN [.COUNT] eql .CCTLR        ! IF THIS BUFFER IS ALLOCATED TO THE CURRENT CONTROLLER
;        6158          then
;        6159              BUFF_OWN [.COUNT] = -1;             ! DEALLOCATE IT


                                .SBTTL  PUTA.BUFF GLOBAL ROUTINES
000000  010146          PUTA.BUFF::
                                MOV     R1,-(SP)                        ;                               6148
000002  005000                  CLR     R0                              ; COUNT                         6155
000004  116001  000000G 1$:     MOVB    BUFF.OWN(R0),R1                 ; *(COUNT),*                    6157
000010  020137  000000G         CMP     R1,CCTLR
000014  001003                  BNE     2$
000016  112760  000377  000000G        MOVB    #377,BUFF.OWN(R0)       ; *,*(COUNT)                    6159
000024  005200          2$:     INC     R0                              ; COUNT                         6155
000026  020027  000007          CMP     R0,#7                           ; COUNT,*
000032  003764                  BLE     1$
000034  012601                  MOV     (SP)+,R1                        ;                               6148
000036  000207                  RTS     PC

; Routine Size:  16 words,      Routine Base:  $CODE$ + 6140
; Maximum stack depth per invocation:  2 words
```

```
;        6160  global routine OUT_IODQ =
;        6161
;        6162  !+
;        6163  !       THIS ROUTINE RETURNS TO THE CALLER THE NEXT RETPKT INDEX TO BE
;        6164  !       PROCESSED FROM THE I/O DONE QUEUE (IODQ). THE "OUT" POINTER TO THE
;        6165  !       QUEUE IS ALSO UPDATED.
;        6166  !
;        6167  !       INPUTS:
;        6168  !               NONE
;        6169  !
;        6170  !       OUTPUTS:
;        6171  !               THE INDEX OF THE NEXT RETPKT TO BE PROCESSED.
;        6172  !-
;        6173
;        6174      begin
;        6175
;        6176      local
;        6177          index : word;
;        6178
;        6179      index = .IODQ [.IODQ_OUT];                ! GET NEXT RETPKT INDEX
;        6180      IODQ_OUT = .IODQ_OUT + 1;                 ! ADVANCE "OUT" POINTER
;        6181
;        6182      if .IODQ_OUT gequ IODQ_LEN                ! IF BEYOND END OF QUEUE
;        6183      then
;        6184          IODQ_OUT = 0;                         ! SET POINTER TO BEGINNING OF QUEUE
;        6185
;        6186      return .index;                            ! RETURN INDEX TO CALLER
;        6187      end;


                                        .SBTTL  OUT.IODQ GLOBAL ROUTINES
000000  013700  000000G          OUT.IODQ::
                                        MOV     IODQ.OUT,R0                     ;                       6179
000004  116000  000000G                   MOVB    IODQ(R0),R0                     ; *,INDEX
000010  042700  177400                    BIC     #177400,R0                      ; *,INDEX
000014  005237  000000G                   INC     IODQ.OUT                        ;                       6180
000020  023727  000000G 000004            CMP     IODQ.OUT,#4                     ;                       6182
000026  103402                           BLO     1$
000030  005037  000000G                   CLR     IODQ.OUT                        ;                       6184
000034  000207                  1$:      RTS     PC                              ;                       6160

; Routine Size:  15 words,      Routine Base:  $CODE$ + 6200
; Maximum stack depth per invocation:  0 words
```

```
;       6188  global routine IN_IODQ (index) : novalue =
;       6189
;       6190  !+
;       6191  !       THIS ROUTINE INSERTS A RETURN PACKET INDEX INTO THE I/O DONE QUEUE, AND
;       6192  !       UPDATES THE IODQ_IN POINTER.
;       6193  !-
;       6194
;       6195      if ((.IODQ_IN + 1) eql .IODQ_OUT) or
;       6196         (.IODQ_IN - (IODQ_LEN - 1) eql .IODQ_OUT)
;       6197      then
;       6198          return
;       6199      else
;       6200          begin
;       6201          IODQ [.IODQ_IN] = .index;               ! LOAD INDEX INTO QUEUE
;       6202          IODQ_IN = .IODQ_IN + 1;                 ! ADVANCE "IN" POINTER
;       6203
;       6204          if .IODQ_IN gequ IODQ_LEN               ! IF BEYOND END OF QUEUE
;       6205          then
;       6206              IODQ_IN = 0;                        ! CYCLE BACK TO BEGINNING OF QUEUE
;       6207
;       6208          end;                                    ! IF IODQ IS NOT FULL


                                .SBTTL  IN.IODQ GLOBAL ROUTINES
000000  010146                IN.IODQ::
                                MOV     R1,-(SP)                        ;                       6188
000002  013701  000000G         MOV     IODQ.IN,R1                      ;                       6195
000006  010100                  MOV     R1,R0
000010  005200                  INC     R0
000012  020037  000000G         CMP     R0,IODQ.OUT
000016  001421                  BEQ     1$
000020  010100                  MOV     R1,R0                           ;                       6196
000022  162700  000003          SUB     #3,R0
000026  020037  000000G         CMP     R0,IODQ.OUT
000032  001413                  BEQ     1$                              ;                       6198
000034  116661  000004  000000G MOVB    4(SP),IODQ(R1)                  ; INDEX,*               6201
000042  005237  000000G         INC     IODQ.IN                         ;                       6202
000046  023727  000000G 000004  CMP     IODQ.IN,#4                      ;                       6204
000054  103402                  BLO     1$
000056  005037  000000G         CLR     IODQ.IN                         ;                       6206
000062  012601                1$:MOV     (SP)+,R1                        ;                       6188
000064  000207                  RTS     PC

; Routine Size:  27 words,      Routine Base:  $CODE$ + 6236
; Maximum stack depth per invocation:  2 words
```

```
;       6209  global routine DROP_CTLR (CTLR, REASON) : novalue =
;       6210
;       6211  !+
;       6212  !       THIS ROUTINE DROPS ALL UNITS ASSOCIATED WITH THE CONTROLLER DESIGNATED
;       6213  !       BY "CTLR". THE REASON FOR DROPPING THE DEVICE IS LOADED INTO THE DUR
;       6214  !       VECTOR FOR EACH ATTACHED UNIT. THIS DATA IS THEN USED BY THE DROP UNIT
;       6215  !       SECTION.
;       6216  !-
;       6217
;       6218      begin
;       6219
;       6220      local
;       6221          UNIT;
;       6222
;       6223      incr N from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do  ! FOR EACH UNIT IN CST
;       6224
;       6225          if .CST [.CTLR, .N, D_PRES] eql PRESENT ! IF UNIT IS CONFIGURED
;       6226          then
;       6227              begin
;       6228              UNIT = .CST [.CTLR, .N, D_UNIT];    ! GET DRS UNIT NUMBER
;       6229              DUR [.UNIT] = .REASON;              ! SET REASON FOR DROPPING UNIT
;       6230              DODU (.UNIT);                       ! DROP UNIT
;       6231              end;
;       6232
;       6233      end;


                                        .SBTTL  DROP.CTLR GLOBAL ROUTINES
000000  004137  000000G           DROP.CTLR::
                                        JSR     R1,$SAVE3                       ;                       6209
000004  016646  000014                  MOV     14(SP),-(SP)                    ; CTLR,*                6225
000010  012746  000027                  MOV     #27,-(SP)
000014  004737  000000G                 JSR     PC,BL$MUL
000020  010003                          MOV     R0,R3
000022  012702  000003                  MOV     #3,R2                           ; *,N                   6223
000026  010300                  1$:     MOV     R3,R0                           ;                       6225
000030  060200                          ADD     R2,R0                           ; N,*
000032  006300                          ASL     R0
000034  032760  040000  000000G         BIT     #40000,CST(R0)
000042  001412                          BEQ     2$
000044  016001  000000G                 MOV     CST(R0),R1                      ; *,UNIT                6228
000050  000301                          SWAB    R1                              ; UNIT
000052  042701  177760                  BIC     #177760,R1                      ; *,UNIT
000056  116661  000016  000000G         MOVB    16(SP),DUR(R1)                  ; REASON,*(UNIT)        6229
000064  010100                          MOV     R1,R0                           ; UNIT,*                6230
000066  104451                          TRAP    51
000070  062702  000005          2$:     ADD     #5,R2                           ; *,N                   6223
000074  020227  000022                  CMP     R2,#22                          ; N,*
000100  003752                          BLE     1$
000102  022626                          CMP     (SP)+,(SP)+                     ;                       6218
000104  000207                          RTS     PC                              ;                       6209

; Routine Size:  35 words,      Routine Base:  $CODE$ + 6324
```

```
; Maximum stack depth per invocation:  8 words
```

```
;       6234  global routine DRV_CTLERR (CTLR) : novalue =
;       6235
;       6236  !+
;       6237  !       THIS ROUTINE IS CALLED BY DRV_TIMCHK AND FATAL_ERROR WHENEVER AN
;       6238  !       UNRECOVERABLE CONTROLLER ERROR HAS BEEN DETECTED. ITS PURPOSE IS TO
;       6239  !       CLEAN UP ALL CONTROLLER-RELATED DATA IN THE "DRIVER" PORTION OF THE
;       6240  !       PROGRAM. THIS INCLUDES MARKING THE CONTROLLER OFFLINE, CLEARING THE
;       6241  !       C-RING COUNT, AND DEALLOCATING MSCP PACKETS DESCRIBED IN THE RESPONSE
;       6242  !       RING.
;       6243  !
;       6244  !       INPUTS:
;       6245  !               CTLR - DYING CONTROLLER NUMBER
;       6246  !-
;       6247
;       6248      begin
;       6249
;       6250      local
;       6251          D_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS);  ! CONTROLLER'S DCT ADDRESS
;       6252
;       6253      D_ADDR = DCT + (.CTLR * DCT_LEN * 2);       ! GET CONTROLLER'S DCT ADDR
;       6254      D_ADDR [WORD0] = OFFLINE;                   ! MARK DCT OFFLINE AND CLEAR CRING_CNT
;       6255      PUTA_PKT (.CTLR);                           ! RELEASE ALL PACKETS ALLOCATED TO CONTROLLER
;       6256      DROP_CTLR (.CTLR, DU_CFATAL);               ! DROP ALL UNITS ON THE CONTROLLER
;       6257      end;                                        ! ROUTINE DRV_CTLERR

                                .SBTTL  DRV.CTLERR GLOBAL ROUTINES
000000  010146          DRV.CTLERR::
                                MOV     R1,-(SP)                        ;                       6234
000002  016601  000004          MOV     4(SP),R1                        ; CTLR,*                6253
000006  010146                  MOV     R1,-(SP)
000010  012746  000022          MOV     #22,-(SP)
000014  004737  000000G         JSR     PC,BL$MUL
000020  062700  000000G         ADD     #DCT,R0
000024  005010                  CLR     (R0)                            ; D.ADDR                6254
000026  010116                  MOV     R1,(SP)                         ;                       6255
000030  004737  005636'          JSR     PC,PUTA.RKT
000034  010116                  MOV     R1,(SP)                         ;                       6256
000036  012746  000006          MOV     #6,-(SP)
000042  004737  006324'          JSR     PC,DROP.CTLR
000046  062706  000006          ADD     #6,SP                           ;                       6248
000052  012601                  MOV     (SP)+,R1                        ;                       6234
000054  000207                  RTS     PC

; Routine Size:  23 words,      Routine Base:  $CODE$ + 6432
; Maximum stack depth per invocation:  5 words
```

```
;       6258  global routine SEND (index) =
;       6259
;       6260  !+
;       6261  !       IF THE CURRENT RDRX IS ONLINE AND ITS CRING IS NOT FULL, THEN THIS
;       6262  !       ROUTINE "SENDS" A COMMAND TO THE RDRX BY LOADING THE PACKET
;       6263  !       DESCRIPTOR OF AN MSCP PACKET INTO THE COMMAND RING AND READING THE
;       6264  !       DEVICE'S IP REGISTER.  IF THE
;       6265  !       CURRENT RDRX IN NOT ONLINE, THEN A FAILURE INDICATION IS RETURNED TO
;       6266  !       THE CALLER, AND NO ACTION IS TAKEN.
;       6267  !
;       6268  !       INPUTS:
;       6269  !               INDEX - INDEX OF MSCP PACKET CONTAINING THE COMMAND TO
;       6270  !                       BE SENT
;       6271  !
;       6272  !       IMPLICIT INPUTS:
;       6273  !               CCTLR - CURRENT CONTROLLER NUMBER
;       6274  !               DCT_ADDR - ADDRESS OF CURRENT CONTROLLER'S DCT
;       6275  !-
;       6276
;       6277      begin
;       6278
;       6279      local
;       6280          SLOT_ADDR,
;       6281          TEMP : word;
;       6282
;       6283      if ((.DCT_ADDR [STAT] eql ONLINE) and                       ! IF DEVICE IS ONLINE AND
;       6284          (.DCT_ADDR [CRING_CNT] lssu CRING_LEN)) or              ! ITS CRING IS NOT FULL
;       6285         ((.MSCP_PKT [.index, OPCODE] eql OP_SCC) and             ! OR IT IS A SET-CTRL-CHAR COMMAND
;       6286          (.DCT_ADDR [CRING_CNT] lssu CRING_LEN))                 !
;       6287      then
;       6288
;       6289          if (not ((.MSCP_PKT [.index, OPCODE] eql OP_ACC) or
;       6290                   (.MSCP_PKT [.index, OPCODE] eql OP_ONL) or
;       6291                   (.MSCP_PKT [.index, OPCODE] eql OP_RD) or
;       6292                   (.MSCP_PKT [.index, OPCODE] eql OP_SCC) or
;       6293                   (.MSCP_PKT [.index, OPCODE] eql OP_WRT) OR  !^
;       6294                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_SDD) OR
;       6295                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_RCD) OR
;       6296                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_GDS) OR
;       6297                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_ELP) OR
;       6298                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_ABP) OR
;       6299                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_ESP) ) )
;       6300          then
;       6301              begin
;       6302              PRINTF (DBM107, .MSCP_PKT [.index, OPCODE]);
;       6303              return FAILURE;
;       6304              end
;       6305          else
;       6306              begin
;       6307
;       6308              do
;       6309                  BREAK                                           ! LOOP TILL CREDIT BALANCE POSITIVE
;       6310              until ((.MSCP_PKT [.index, CMD_TYPE] eql IMM_CMD) and
```

```
;        6311                    (.CREDIT_BAL gequ 1)) or
;        6312                   (.CREDIT_BAL gtru 1);
;        6313
;        6314                MSCP_PKT [.index, CRN_LO] = (CRN_LOW = .CRN_LOW + 1);       ! ASSIGN CMD REF NUM
;        6315
;        6316                if .CRN_LOW eql 0
;        6317                then  CRN_HIGH = .CRN_HIGH + 1;
;        6318
;        6319                MSCP_PKT [.index, CRN_HI] = .CRN_HIGH;
;        6320
;        6321                SLOT_ADDR = .DCT_ADDR [CR_NEXT];                    ! ADDR OF NEXT COMMAND SLOT
;        6322
;        6323                DO BREAK
;        6324                UNTIL ((.(.SLOT_ADDR + 2) and ED_OWN) eql 0);
;        6325
;        6326                SETPRI (PRI07);
;        6327
;        6328                .SLOT_ADDR = .MSCP_PKT [.index, PKT_LO];            ! LOAD BUFF DESC (LO) INTO COMMAND SLOT
;        6329                SLOT_ADDR = .SLOT_ADDR + 2;                         ! ADVANCE TO NEXT WORD
;        6330                .SLOT_ADDR = .MSCP_PKT [.index, PKT_HI];            ! LOAD BUFF DESC (HI) INTO COMMAND SLOT
;        6331                .SLOT_ADDR = ..SLOT_ADDR and (not (ED_FLAG));       ! CLEAR INTERRUPT FLAG IN CASE SET
;        6332                .SLOT_ADDR = ..SLOT_ADDR or ED_OWN;                 ! GIVE OWNERSHIP TO CONTROLLER
;        6333                SLOT_ADDR = .SLOT_ADDR + 2;                         ! ADVANCE TO NEXT COMMAND SLOT
;        6334
;        6335                if .SLOT_ADDR gtra .DCT_ADDR [CR_END]               ! IF BEYOND END OF CRING
;        6336                then
;        6337                    SLOT_ADDR = .DCT_ADDR [CR_BEG];                 ! CYCLE BACK TO BEGINNING
;        6338
;        6339                DCT_ADDR [CR_NEXT] = .SLOT_ADDR;                    ! RESTORE CR_NEXT POINTER IN DCT
;        6340                DCT_ADDR [CRING_CNT] = .DCT_ADDR [CRING_CNT] + 1;   ! INCR # OF COMMANDS IN CRING
;        6341                CREDIT_BAL = .CREDIT_BAL - 1;                       ! DECREMENT CREDIT BALANCE
;        6342                TEMP = .RDRX_ADDR [RCIP, RC_ALL];                   ! READ IP TO FORCE PORT TO POLL
;        6343                SETPRI (PRI00);                                     ! LOWER PRIORITY
;        6344                return SUCCESS;
;        6345                end
;        6346
;        6347        else
;        6348            return FAILURE;                                         ! IF DEVICE IS NOT ONLINE
;        6349
;        6350        end;                                                        ! ROUTINE SEND


                                    .SBTTL  SEND GLOBAL ROUTINES
000000  004137  000000G     SEND::  JSR     R1,$SAVE2                   ;                   6258
000004  005746                     TST     -(SP)
000006  013701  000000G             MOV     DCT.ADDR,R1                 ;                   6283
000012  005711                     TST     (R1)
000014  100003                     BPL     1$
000016  121127  000004             CMPB    (R1),#4                     ;                   6284
000022  103416                     BLO     2$
000024  016646  000012     1$:     MOV     12(SP),-(SP)                ; INDEX,*           6285
000030  012746  000104             MOV     #104,-(SP)
000034  004737  000000G             JSR     PC,BL$MUL
```

```
000040  022626                          CMP     (SP)+,(SP)+
000042  126027  000022G 000004          CMPB    MSCP.PKT+22(R0),#4
000050  001170                          BNE     9$
000052  121127  000004                  CMPB    (R1),#4                 ;                               6286
000056  103165                          BHIS    9$
000060  016646  000012          2$:     MOV     12(SP),-(SP)            ; INDEX,*                       6289
000064  012746  000104                  MOV     #104,-(SP)
000070  004737  000000G                 JSR     PC,BL$MUL
000074  010002                          MOV     R0,R2
000076  022626                          CMP     (SP)+,(SP)+
000100  005000                          CLR     R0
000102  156200  000022G                 BISB    MSCP.PKT+22(R2),R0
000106  020027  000020                  CMP     R0,#20
000112  001450                          BEQ     3$
000114  020027  000011                  CMP     R0,#11                  ;                               6290
000120  001445                          BEQ     3$
000122  020027  000041                  CMP     R0,#41                  ;                               6291
000126  001442                          BEQ     3$
000130  020027  000004                  CMP     R0,#4                   ;                               6292
000134  001437                          BEQ     3$
000136  020027  000042                  CMP     R0,#42                  ;                               6293
000142  001434                          BEQ     3$
000144  020027  000004                  CMP     R0,#4                   ;                               6294
000150  001431                          BEQ     3$
000152  020027  000005                  CMP     R0,#5                   ;                               6295
000156  001426                          BEQ     3$
000160  020027  000001                  CMP     R0,#1                   ;                               6296
000164  001423                          BEQ     3$
000166  020027  000003                  CMP     R0,#3                   ;                               6297
000172  001420                          BEQ     3$
000174  020027  000006                  CMP     R0,#6                   ;                               6298
000200  001415                          BEQ     3$
000202  020027  000002                  CMP     R0,#2                   ;                               6299
000206  001412                          BEQ     3$
000210  010046                          MOV     R0,-(SP)                ;                               6302
000212  012746  000000G                 MOV     #DBM107,-(SP)
000216  012746  000002                  MOV     #2,-(SP)
000222  010600                          MOV     SP,R0                   ; SP,*
000224  104417                          TRAP    17
000226  062706  000006                  ADD     #6,SP                   ;                               6301
000232  000477                          BR      9$                      ;                               6289
000234  104422                  3$:     TRAP    22                      ;                               6308
000236  005762  000004G                 TST     MSCP.PKT+4(R2)          ;                               6310
000242  001003                          BNE     4$
000244  005737  000000G                 TST     CREDIT.BAL              ;                               6311
000250  001004                          BNE     5$
000252  023727  000000G 000001  4$:     CMP     CREDIT.BAL,#1           ;                               6312
000260  101765                          BLOS    3$
000262  013700  000000G          5$:     MOV     CRN.LOW,R0              ;                               6314
000266  005200                          INC     R0
000270  010037  000000G                 MOV     R0,CRN.LOW
000274  010062  000012G                 MOV     R0,MSCP.PKT+12(R2)
000300  001002                          BNE     6$                      ;                               6316
```

```
000302  005237  000000G                 INC     CRN.HIGH                        ;                                       6317
000306  013762  000000G 000014G  6$:     MOV     CRN.HIGH,MSCP.PKT+14(R2)        ;                                       6319
000314  013700  000000G                 MOV     DCT.ADDR,R0                     ;                                       6321
000320  016001  000020                  MOV     20(R0),R1                       ; *,SLOT.ADDR
000324  104422                  7$:     TRAP    22                              ;                                       6323
000326  032761  100000  000002          BIT     #-100000,2(R1)                  ; *,*(SLOT.ADDR)                        6324
000334  001373                          BNE     7$
000336  012700  000340                  MOV     #340,R0                         ;                                       6326
000342  104441                          TRAP    41
000344  016221  000000G                 MOV     MSCP.PKT(R2),(R1)+              ; *,SLOT.ADDR                           6328
000350  016211  000002G                 MOV     MSCP.PKT+2(R2),(R1)             ; *,SLOT.ADDR                           6330
000354  042711  040000                  BIC     #40000,(R1)                     ; *,SLOT.ADDR                           6331
000360  052721  100000                  BIS     #100000,(R1)+                   ; *,SLOT.ADDR                           6332
000364  013700  000000G                 MOV     DCT.ADDR,R0                     ;                                       6335
000370  020160  000012                  CMP     R1,12(R0)                       ; SLOT.ADDR,*
000374  101402                          BLOS    8$
000376  016001  000010                  MOV     10(R0),R1                       ; *,SLOT.ADDR                           6337
000402  010160  000020          8$:     MOV     R1,20(R0)                       ; SLOT.ADDR,*                           6339
000406  105210                          INCB    (R0)                            ;                                       6340
000410  005337  000000G                 DEC     CREDIT.BAL                      ;                                       6341
000414  017716  000000G                 MOV     @RDRX.ADDR,(SP)                 ; *,RC.REG                              6342
000420  005000                          CLR     R0                              ;                                       6343
000422  104441                          TRAP    41
000424  012700  000001                  MOV     #1,R0                           ;                                       6289
000430  000401                          BR      10$                             ;                                       6277
000432  005000                  9$:     CLR     R0
000434  005726                  10$:    TST     (SP)+                           ;                                       6258
000436  000207                          RTS     PC

; Routine Size:  144 words,    Routine Base:  $CODE$ + 6510
; Maximum stack depth per invocation:  9 words
```

```
;          6351  global routine WAIT : novalue =
;          6352
;          6353  !+
;          6354  !       THE PURPOSE OF THIS ROUTINE IS TO KILL TIME UNTIL AN RDRX INTERRUPT
;          6355  !       RESULTS IN A RETURN PACKET INDEX BEING DEPOSITED INTO THE I/O DONE
;          6356  !       QUEUE (IODQ).
;          6357  !-
;          6358
;          6359      do
;          6360          BREAK                                   ! BREAK FOR ACT
;          6361      until .IODQ_IN neq .IODQ_OUT;


                                        .SBTTL  WAIT GLOBAL ROUTINES
000000  104422                  WAIT::
000000                          1$:     TRAP    22                      ;  6359
000002  023737  000000G 000000G         CMP     IODQ.IN,IODQ.OUT        ;  6361
000010  001773                         BEQ     1$
000012  000207                         RTS     PC                      ;  6351

; Routine Size:  6 words,      Routine Base:  $CODE$ + 7150
; Maximum stack depth per invocation:  2 words
```

```
;        6362  %sbttl 'ERROR MESSAGE SUBROUTINES'
;        6363
;        6364  routine EMS_SA : novalue =
;        6365
;        6366  !+
;        6367  !       THIS ROUTINE PRINTS (EXTENDED) THE GLOBAL DATUM "SA_REG" WHICH CONTAINS
;        6368  !       THE CONTENTS OF THE SA REGISTER.
;        6369  !-
;        6370
;        6371      if .SA_REG eql %o'177777'                                   ! IF CONTROLLER TIME-OUT
;        6372      then
;        6373          begin
;        6374          PRINTX (CRLF);
;        6375          PRINTX (ASTERISK);
;        6376          PRINTX (.CNTR_ERR [0]);
;        6377          end
;        6378      else
;        6379
;        6380          if (.SA_REG and %o'003777') lequ 22                     ! IF GENERIC CONTROLLER ERROR
;        6381          then
;        6382              begin
;        6383              PRINTX (CRLF);
;        6384              PRINTX (ASTERISK);
;        6385              PRINTX (.CNTR_ERR [.SA_REG and %o'003777']);
;        6386              end
;        6387          else
;        6388
;        6389              if ((.SA_REG and %o'003777') - 400) lequ 6           ! IF RDRX SPECIFIC CONTROLLER ERROR
;        6390              then
;        6391                  begin
;        6392                  PRINTX (CRLF);
;        6393                  PRINTX (ASTERISK);
;        6394                  PRINTX (.RDRX_ERR [(.SA_REG and %o'003777') - 400]);
;        6395                  end
;        6396              else
;        6397                  PRINTX (XX14, .SA_REG);                         ! JUST PRINT CONTENTS OF SA


                                .SBTTL  EMS.SA ERROR MESSAGE SUBROUTINES
000000  010146          EMS.SA: MOV     R1,-(SP)                        ;       6364
000002  013701  000000G         MOV     SA.REG,R1                       ;       6371
000006  020127  177777          CMP     R1,#-1
000012  001023                  BNE     1$
000014  012746  000000G         MOV     #CRLF,-(SP)                     ;       6374
000020  012746  000001          MOV     #1,-(SP)
000024  010600                  MOV     SP,R0                           ; SP,*
000026  104415                  TRAP    15
000030  012716  000000G         MOV     #ASTERISK,(SP)                  ;       6375
000034  012746  000001          MOV     #1,-(SP)
000040  010600                  MOV     SP,R0                           ; SP,*
000042  104415                  TRAP    15
000044  013716  000000G         MOV     CNTR.ERR,(SP)                   ;       6376
000050  012746  000001          MOV     #1,-(SP)
```

```
000054  010600                          MOV     SP,R0                   ; SP,*
000056  104415                          TRAP    15
000060  000475                          BR      3$                      ;                       6373
000062  010100                  1$:     MOV     R1,R0                   ;                       6380
000064  042700  174000                  BIC     #174000,R0
000070  020027  000026                  CMP     R0,#26
000074  101030                          BHI     2$
000076  012746  000000G                 MOV     #CRLF,-(SP)             ;                       6383
000102  012746  000001                  MOV     #1,-(SP)
000106  010600                          MOV     SP,R0                   ; SP,*
000110  104415                          TRAP    15
000112  012716  000000G                 MOV     #ASTERISK,(SP)          ;                       6384
000116  012746  000001                  MOV     #1,-(SP)
000122  010600                          MOV     SP,R0                   ; SP,*
000124  104415                          TRAP    15
000126  013700  000000G                 MOV     SA.REG,R0               ;                       6385
000132  042700  174000                  BIC     #174000,R0
000136  006300                          ASL     R0
000140  016016  000000G                 MOV     CNTR.ERR(R0),(SP)
000144  012746  000001                  MOV     #1,-(SP)
000150  010600                          MOV     SP,R0                   ; SP,*
000152  104415                          TRAP    15
000154  000437                          BR      3$                      ;                       6382
000156  010100                  2$:     MOV     R1,R0                   ;                       6389
000160  042700  174000                  BIC     #174000,R0
000164  162700  000620                  SUB     #620,R0
000170  020027  000006                  CMP     R0,#6
000174  101031                          BHI     4$
000176  012746  000000G                 MOV     #CRLF,-(SP)             ;                       6392
000202  012746  000001                  MOV     #1,-(SP)
000206  010600                          MOV     SP,R0                   ; SP,*
000210  104415                          TRAP    15
000212  012716  000000G                 MOV     #ASTERISK,(SP)          ;                       6393
000216  012746  000001                  MOV     #1,-(SP)
000222  010600                          MOV     SP,R0                   ; SP,*
000224  104415                          TRAP    15
000226  013700  000000G                 MOV     SA.REG,R0               ;                       6394
000232  042700  174000                  BIC     #174000,R0
000236  006300                          ASL     R0
000240  016016  176340G                 MOV     RDRX.ERR-1440(R0),(SP)
000244  012746  000001                  MOV     #1,-(SP)
000250  010600                          MOV     SP,R0                   ; SP,*
000252  104415                          TRAP    15
000254  005726                  3$:     TST     (SP)+                   ;                       6391
000256  000407                          BR      5$                      ;                       6389
000260  010146                  4$:     MOV     R1,-(SP)                ;                       6397
000262  012746  000000G                 MOV     #XX14,-(SP)
000266  012746  000002                  MOV     #2,-(SP)
000272  010600                          MOV     SP,R0                   ; SP,*
000274  104415                          TRAP    15
000276  062706  000006          5$:     ADD     #6,SP                   ;                       6371
000302  012601                          MOV     (SP)+,R1                ;                       6364
000304  000207                          RTS     PC
```

ZRQAM2 RD/RX EXERCISER 21-Jul-1983 15:25:58 VAX-11 Bliss-16 V3-555
V01.2 ERROR MESSAGE SUBROUTINES 21-Jul-1983 15:19:20 DISK$USER2:[DIETZ.RELEASE]ZRQAB0.BL1;10 (63)

```
; Routine Size:  99 words,      Routine Base:  $CODE$ + 7164
; Maximum stack depth per invocation:  7 words
```

```
;       6398  routine EMS_SBC : novalue =
;       6399
;       6400  !+
;       6401  !       THIS ROUTINE PRINTS  THE GLOBAL DATUM "SB_CODE" (SUB-CODE) IF
;       6402  !       EITHER THE STATUS CODE (ST_CODE) OR THE SUB-CODE IS NON-ZERO. (A
;       6403  !       NON-ZERO SUB-CODE ALWAYS HAS SIGNIFICANCE, WHEREAS A ZERO SUB-CODE ONLY
;       6404  !       HAS MEANING WITH A NON-ZERO STATUS CODE).
;       6405  !-
;       6406
;       6407      begin
;       6408
;       6409      if (.ST_CODE or .SB_CODE) neq 0                                   ! PRINT SUB-CODE ONLY ON ERROR
;       6410      then
;       6411          begin
;       6412          PRINTB (XX16);                                                ! SUB-CODE :
;       6413
;       6414          case .ST_CODE from ST_SUC to ST_DRV of
;       6415              set
;       6416
;       6417              [ST_SUC]:           if .SB_CODE lequ 16                   ! SUCCESS SUB-CODES
;       6418                                  then
;       6419                                      PRINTB (.TBL_SUC [.SB_CODE]);
;       6420
;       6421              [ST_CMD]:           PRINTB (EX_OP, .SB_CODE / 8);         ! INVALID COMMAND
;       6422
;       6423              [ST_ABO]:           ;                                     ! COMMAND ABORTED
;       6424
;       6425              [ST_OFL]:           if .SB_CODE lequ 8                    ! UNIT OFFLINE
;       6426                                  then
;       6427                                      PRINTB (.TBL_OFL [.SB_CODE]);
;       6428
;       6429              [ST_AVL]:           ;                                     ! UNIT AVAILABLE
;       6430
;       6431              [ST_MFE]:           if .SB_CODE lequ 10                   ! MEDIA FORMAT ERROR
;       6432                                  then
;       6433                                      PRINTB (.TBL_MFE [.SB_CODE]);
;       6434
;       6435              [ST_WPT]:           if (.SB_CODE / 128) lequ 2            ! WRITE PROTECTED
;       6436                                  then
;       6437                                      PRINTB (.TBL_WPT [(.SB_CODE / 128)]);
;       6438
;       6439              [ST_CMP]:           ;                                     ! COMPARE ERROR
;       6440
;       6441              [ST_DAT]:           if .SB_CODE lequ 15                   ! DATA ERROR
;       6442                                  then
;       6443                                      PRINTB (.TBL_DAT [.SB_CODE]);
;       6444
;       6445              [ST_HST]:           if .SB_CODE lequ 4                    ! HOST ACCESS ERROR
;       6446                                  then
;       6447                                      PRINTB (.TBL_HST [.SB_CODE]);
;       6448
;       6449              [ST_CNT]:           if .SB_CODE lequ 3                    ! CONTROLLER ERROR
;       6450                                  then
```

```
;       6451                                    PRINTB (.TBL_CNT [.SB_CODE]);
;       6452
;       6453            [ST_DRV]:               if .SB_CODE lequ 8                      ! DRIVE ERROR
;       6454                                    then
;       6455                                        PRINTB (.TBL_DRV [.SB_CODE]);
;       6456
;       6457            [outrange]:             PRINTB (EX_OP, .SB_CODE);               ! JUST PRINT SUB-CODE IF NO MATCH
;       6458            tes;
;       6459
;       6460        end;
;       6461
;       6462    end;


                                        .SBTTL  EMS.SBC ERROR MESSAGE SUBROUTINES
000000  013700  000000G          EMS.SBC:MOV     ST.CODE,R0                      ;                                       6409
000004  053700  000000G                  BIS     SB.CODE,R0
000010  001001                           BNE     1$
000012  000207                           RTS     PC
000014  012746  000000G          1$:     MOV     #XX16,-(SP)                     ;                                       6412
000020  012746  000001                   MOV     #1,-(SP)
000024  010600                           MOV     SP,R0                           ; SP,*
000026  104414                           TRAP    14
000030  013700  000000G                  MOV     ST.CODE,R0                      ;                                       6414
000034  020027  000013                   CMP     R0,#13
000040  101003                           BHI     3$
000042  006300                           ASL     R0
000044  066007  000000'                   ADD     P.AAA(R0),PC                    ; Case dispatch
000050  013716  000000G          3$:     MOV     SB.CODE,(SP)                    ;                                       6457
000054  012746  000000G                  MOV     #EX.OP,-(SP)
000060  012746  000002                   MOV     #2,-(SP)
000064  010600                           MOV     SP,R0                           ; SP,*
000066  104414                           TRAP    14
000070  022626                           CMP     (SP)+,(SP)+
000072  000435                           BR      6$                              ;                                       6414
000074  023727  000000G 000020  4$:      CMP     SB.CODE,#20                     ;                                       6417
000102  101165                           BHI     14$
000104  013700  000000G                  MOV     SB.CODE,R0                      ;                                       6419
000110  006300                           ASL     R0
000112  016016  000000'                   MOV     TBL.SUC(R0),(SP)
000116  012746  000001                   MOV     #1,-(SP)
000122  010600                           MOV     SP,R0                           ; SP,*
000124  104414                           TRAP    14
000126  000565                           BR      15$
000130  013716  000000G          5$:     MOV     SB.CODE,(SP)                    ;                                       6421
000134  012746  000010                   MOV     #10,-(SP)
000140  004737  000000G                  JSR     PC,BL$DIV
000144  010016                           MOV     R0,(SP)
000146  012746  000000G                  MOV     #EX.OP,-(SP)
000152  012746  000002                   MOV     #2,-(SP)
000156  010600                           MOV     SP,R0                           ; SP,*
000160  104414                           TRAP    14
000162  062706  000006                   ADD     #6,SP
```

```
000166  000546                          6$:     BR      16$                             ;                               6414
000170  023727  000000G 000010         7$:     CMP     SB.CODE,#10                     ;                               6425
000176  101142                                  BHI     16$
000200  013700  000000G                         MOV     SB.CODE,R0                      ;                               6427
000204  006300                                  ASL     R0
000206  016016  000042'                         MOV     TBL.OFL(R0),(SP)
000212  012746  000001                          MOV     #1,-(SP)
000216  010600                                  MOV     SP,R0                           ; SP,*
000220  104414                                  TRAP    14
000222  000527                                  BR      15$
000224  023727  000000G 000012         8$:     CMP     SB.CODE,#12                     ;                               6431
000232  101124                                  BHI     16$
000234  013700  000000G                         MOV     SB.CODE,R0                      ;                               6433
000240  006300                                  ASL     R0
000242  016016  000064'                         MOV     TBL.MFE(R0),(SP)
000246  012746  000001                          MOV     #1,-(SP)
000252  010600                                  MOV     SP,R0                           ; SP,*
000254  104414                                  TRAP    14
000256  000511                                  BR      15$
000260  013716  000000G                 9$:     MOV     SB.CODE,(SP)                    ;                               6435
000264  012746  000200                          MOV     #200,-(SP)
000270  004737  000000G                         JSR     PC,BL$DIV
000274  005726                                  TST     (SP)+
000276  020027  000002                          CMP     R0,#2
000302  101100                                  BHI     16$
000304  006300                                  ASL     R0                              ;                               6437
000306  016016  000112'                         MOV     TBL.WPT(R0),(SP)
000312  012746  000001                          MOV     #1,-(SP)
000316  010600                                  MOV     SP,R0                           ; SP,*
000320  104414                                  TRAP    14
000322  000467                                  BR      15$
000324  023727  000000G 000017         10$:    CMP     SB.CODE,#17                     ;                               6441
000332  101064                                  BHI     16$
000334  013700  000000G                         MOV     SB.CODE,R0                      ;                               6443
000340  006300                                  ASL     R0
000342  016016  000120'                         MOV     TBL.DAT(R0),(SP)
000346  012746  000001                          MOV     #1,-(SP)
000352  010600                                  MOV     SP,R0                           ; SP,*
000354  104414                                  TRAP    14
000356  000451                                  BR      15$
000360  023727  000000G 000004         11$:    CMP     SB.CODE,#4                      ;                               6445
000366  101046                                  BHI     16$
000370  013700  000000G                         MOV     SB.CODE,R0                      ;                               6447
000374  006300                                  ASL     R0
000376  016016  000160'                         MOV     TBL.HST(R0),(SP)
000402  012746  000001                          MOV     #1,-(SP)
000406  010600                                  MOV     SP,R0                           ; SP,*
000410  104414                                  TRAP    14
000412  000433                                  BR      15$
000414  023727  000000G 000003         12$:    CMP     SB.CODE,#3                      ;                               6449
000422  101030                                  BHI     16$
000424  013700  000000G                         MOV     SB.CODE,R0                      ;                               6451
000430  006300                                  ASL     R0
```

```
000432  016016  000172'                        MOV     TBL.CNT(R0),(SP)
000436  012746  000001                        MOV     #1,-(SP)
000442  010600                                MOV     SP,R0                   ; SP,*
000444  104414                                TRAP    14
000446  000415                                BR      15$
000450  023727  000000G 000010        13$:    CMP     SB.CODE,#10             ;                       6453
000456  101012                        14$:    BHI     16$
000460  013700  000000G                        MOV     SB.CODE,R0              ;                       6455
000464  006300                                ASL     R0
000466  016016  000202'                        MOV     TBL.DRV(R0),(SP)
000472  012746  000001                        MOV     #1,-(SP)
000476  010600                                MOV     SP,R0                   ; SP,*
000500  104414                                TRAP    14
000502  005726                        15$:    TST     (SP)+
000504  022626                        16$:    CMP     (SP)+,(SP)+             ;                       6411
000506  000207                                RTS     PC                      ;                       6398

; Routine Size:  164 words,     Routine Base:  $CODE$ + 7472
; Maximum stack depth per invocation:  7 words


000000                                  .PSECT  $PLIT$,  RO ,  D

                                P.AAA:                                          ; CASE Table for EMS.SBC+0044   6414
000000  000024                  2$:     .WORD   24                              ; [4$]
000002  000060                          .WORD   60                              ; [5$]
000004  000434                          .WORD   434                             ; [16$]
000006  000120                          .WORD   120                             ; [7$]
000010  000434                          .WORD   434                             ; [16$]
000012  000154                          .WORD   154                             ; [8$]
000014  000210                          .WORD   210                             ; [9$]
000016  000434                          .WORD   434                             ; [16$]
000020  000254                          .WORD   254                             ; [10$]
000022  000310                          .WORD   310                             ; [11$]
000024  000344                          .WORD   344                             ; [12$]
000026  000400                          .WORD   400                             ; [13$]
```

```
;       6463  GLOBAL routine EMSCMD : novalue =
;       6464
;       6465  !+
;       6466  !       THIS ROUTINE PRINTS THE ENTIRE RETURN PACKET INCLUDING OPCODE,
;       6467  !       STAUTUS, SUB-STATUS,  MODFIERS OR FLAGS, AND ETC.
;       6468  !       THESE FIELDS ARE "TRANSLATED" INTO ENGLISH TEXT IF POSSIBLE
;       6469  !       RATHER THAN PRINTED AS RAW NUMBERS.
;       6470  !
;       6471  !       IMPLICIT INPUTS:
;       6472  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;       6473  !-
;       6474  begin
;       6475
;       6476  OWN
;       6477          EBH_TB1 : VECTOR [7] INITIAL (EBH_30,EBH_44,EBH_45,
;       6478                                        EBH_46,EBA_47,EBA_48,EBA_49);
;       6479
;       6480                  ! TABLE OF BASIC, HARD ERROR MESSAGE ADDRESSES, INDEXED BY STATUS CODE
;       6481
;       6482  PRINTB (XX13, .CDISK);                        ! "DISK XXX"
;       6483  !PRINTB (XX36, .CRN_LOW);                                ! EXPECTED CRN : XXXXXX
;       6484  PRINTX (XX35, .RP_ADDR [CRF_LO]);                       ! RECEIVED CRN : XXXXXX
;       6485  printX (xx29);                                          ! "message type:"
;       6486  SELECTU (.RP_ADDR [MESTYP]) OF
;       6487                  SET
;       6488                  [%o'0']: PRINTX (EX_SEQ);       ! "SEQUENTIAL"
;       6489                  [%o'1']: PRINTX (EX_DGM);       ! "DATAGRAM"
;       6490                  [%o'2']: PRINTX (EX_CRD);       ! "CREDIT NOTIFICATION"                 PACKET TYPE
;       6491                  [%o'15']: PRINTX (EX_MTN);      ! "MAINTENANCE"
;       6492                  [OTHERWISE]: PRINTX (XX37, .RP_ADDR [MESTYP]);  ! UNKOWN MESSAGE TYPE
;       6493                  TES;
;       6494
;       6495  PRINTB (XX17);                                  ! "COMMAND: "
;       6496
;       6497  SELECTU (.RP_ADDR [conid]) OF
;       6498  SET
;       6499  [%o'2']:
;       6500      BEGIN
;       6501      PRINTB (XX18);                                                  ! PRINTS -DUP-
;       6502      SELECTU (.RP_ADDR [ENDCOD]) OF
;       6503                  SET
;       6504                  [%o'201']: PRINTB (EX_GDS);
;       6505                  [%o'202']: PRINTB (EX_ESP);                       ! PRINTS A COMMAND
;       6506                  [%o'203']: PRINTB (EX_ELP);
;       6507                  [%o'204']: PRINTB (EX_RCD);
;       6508                  [%o'205']: PRINTB (EX_SDD);
;       6509                  [%o'206']: PRINTB (EX_ABP);
;       6510                  [OTHERWISE]: PRINTB (EX_OP, .RP_ADDR [ENDCOD]); ! PRINT ENDCODE VALUE
;       6511                  TES;
;       6512      printb (xx15);                                                        ! "status:"
;       6513      IF (.RP_ADDR [STSCOD] GEQU 0) AND (.RP_ADDR [STSCOD] LEQU 7)          ! IF STATUS CODE IS WITHIN RANGE
;       6514      THEN        PRINTB (.EBH_TB1 [.RP_ADDR [STSCOD]])                     ! PRINTB APPROPRIATE MESSAGE
;       6515      ELSE    PRINTB (ex_op, .RP_ADDR [STSCOD]);                  ! JUST PRINT STATUS CODE
```

```
;       6516
;       6517        IF .RP_ADDR [ENDCOD] EQL %O'204' or
;       6518           .RP_ADDR [ENDCOD] EQL %O'205'                    ! IF A SEND DATA OR RECEIVE DATA COMMAND THEN
;       6519        then
;       6520            begin
;       6521            PRINTX (XX25, .RP_ADDR [BCNT_LO]);               ! FOR ANY "ACTUAL # OF BYTES TRANSFERRED: XXXXX."
;       6522            PRINTX (XX26, .RP_ADDR [BUFF_1], .RP_ADDR [BUFF_0]);    ! "I/O BUFFER DESCRIPTOR: XXXXXX XXXXXX"
;       6523            EMS_DUP ();                                      ! prints contents of dup packet
;       6524            end;
;       6525        IF .RP_ADDR [ENDCOD] EQL %O'201'
;       6526        then
;       6527            begin
;       6528            if BIT_TST (RP_ADDR [9, 8, 1, 0], 1)
;       6529                then PRINTB (df_0);
;       6530            if BIT_TST (RP_ADDR [9, 9, 1, 0], 1)
;       6531                then PRINTB (df_1);
;       6532            if BIT_TST (RP_ADDR [9, 10, 1, 0], 1)
;       6533                then PRINTB (df_2);
;       6534            if BIT_TST (RP_ADDR [9, 11, 1, 0], 1)
;       6535                then PRINTB (df_3);
;       6536            end;
;       6537
;       6538        IF .RP_ADDR [ENDCOD] EQL %O'203'                     ! IF A GET DUST STATUS OR EXEC. LOCAL PRG COMMAND TH
;       6539        then
;       6540            begin
;       6541            if BIT_TST (RP_ADDR [9, 8, 1, 0], 1)
;       6542                then PRINTB (df_4);
;       6543            if BIT_TST (RP_ADDR [9, 9, 1, 0], 1)
;       6544                then PRINTB (df_5);
;       6545            if BIT_TST (RP_ADDR [9, 10, 1, 0], 1)
;       6546                then PRINTB (df_6);
;       6547            if BIT_TST (RP_ADDR [9, 11, 1, 0], 1)
;       6548                then PRINTB (df_7);
;       6549            end;
;       6550        PRINTX (XX23, .CST_ADDR [.CUOFF + 3, D_DBN], .CST_ADDR [.CUOFF + 3, D_DBN]);        ! "DBN: XXXXXX."
;       6551        END;
;       6552    [%o'0']:
;       6553        BEGIN
;       6554        PRINTB (XX19);                                   !PRINTS -MSCP-                                  !MSC
;       6555        SELECTU (.RP_ADDR [ENDCOD]) OF
;       6556                    SET
;       6557                    [%o'204']: PRINTB (EX_SCC);
;       6558                    [%o'211']: PRINTB (EX_ONL);
;       6559                    [%o'220']: PRINTB (EX_ACC);
;       6560                    [%o'241']: PRINTB (EX_RD);                          ! PRINTS THE COMMAND
;       6561                    [%o'242']: PRINTB (EX_WRT);
;       6562                    [OTHERWISE]: PRINTB (EX_OP, .RP_ADDR [ENDCOD]); ! PRINT ENDCODE VALUE
;       6563                    TES;
;       6564        if .RP_ADDR [CMDMOD] eql MD_CMP THEN PRINTB (XX20); ! PRINTS THE MODIFIER IF NECESSARY
;       6565
;       6566        PRINTB (XX15);                                   ! STATUS:
;       6567        if (.ST_CODE gtru 0) and                         ! IF STATUS CODE IS WITHIN RANGE
;       6568           (.ST_CODE lequ 11)
```

```
;       6569        then
;       6570            PRINTB (.ERR_COD [.ST_CODE - 1])                ! PRINTB APPROPRIATE MESSAGE
;       6571        else
;       6572
;       6573            if .ST_CODE eql ST_DIA
;       6574            then
;       6575                    PRINTB (.ERR_COD [11])                  ! MESSAGE FROM INTERNAL DIAGNOSTICS
;       6576            else
;       6577                    PRINTB (EX_OP, .ST_CODE);               ! JUST PRINT STATUS CODE WHEN NO MATCH
;       6578
;       6579        EMS_SBC ();                                         ! PRINTS STATUS SUB-CODE
;       6580
;       6581        IF .RP_ADDR [ENDCOD] EQLU %o'220' OR                                                    ! MSCP acces
;       6582              .RP_ADDR [ENDCOD] EQLU %o'241' OR                                                 ! mscp read
;       6583              .RP_ADDR [ENDCOD] EQLU %o'242'                                                    ! mscp write
;       6584        THEN
;       6585                begin
;       6586                printX (XX24, .rp_addr [CBCNT_LO]);             ! "BYTE COUNT IN COMMAND: XXXXXXX"
;       6587                PRINTX (XX25, .RP_ADDR [BCNT_LO]);              ! FOR ANY "ACTUAL # OF BYTES TRANSFERRED: XXXXX."
;       6588                PRINTX (XX26, .RP_ADDR [BUFF_1], .RP_ADDR [BUFF_0]);   ! "I/O BUFFER DESCRIPTOR: XXXXXX XXXXXX"
;       6589                if BIT_TST (RP_ADDR [FLAGS], EF_0)      ! IF BAD BLOCK REPORTED
;       6590                    then
;       6591                        PRINTB (XX21, .RP_ADDR [BBLK_LO])       ! "BAD BLOCK REPORTED: XXXXXX."
;       6592                    else
;       6593                        printX (XX22, .RP_ADDR [LBN_LO], .RP_ADDR [LBN_LO]);            ! "LBN: XXXXXX"
;       6594                PRINTB (XX41);
;       6595                if BIT_TST (RP_ADDR [FLAGS], EF_1)      ! IF BAD BLOCK UNREPORTED
;       6596                    then PRINTB (F_1);
;       6597                if BIT_TST (RP_ADDR [FLAGS], EF_2)      ! IF ERROR LOG GENERATED
;       6598                    then PRINTB (F_2);
;       6599                if BIT_TST (RP_ADDR [FLAGS], EF_3)      ! IF SERIOUS EXCEPTION
;       6600                    then PRINTB (F_3);
;       6601                END;
;       6602        IF .RP_ADDR [ENDCOD] EQLU %o'204'                                                       ! MSCP SET CTLR CHAR
;       6603        THEN
;       6604            begin
;       6605                PRINTB (XX39);
;       6606                if BIT_TST (RP_ADDR [BCNT_HI], EF_4)    ! IF
;       6607                    then PRINTB (F_4);
;       6608                if BIT_TST (RP_ADDR [BCNT_HI], EF_5)    ! IF
;       6609                    then PRINTB (F_5);
;       6610                if BIT_TST (RP_ADDR [BCNT_HI], EF_6)    ! IF
;       6611                    then PRINTB (F_6);
;       6612                if BIT_TST (RP_ADDR [BCNT_HI], EF_7)    ! IF
;       6613                    then PRINTB (F_7);
;       6614                if BIT_TST (RP_ADDR [BCNT_HI], EF_8)    ! IF
;       6615                    then PRINTB (F_8);
;       6616                if BIT_TST (RP_ADDR [BCNT_HI], EF_9)    ! IF
;       6617                    then PRINTB (F_9);
;       6618                if BIT_TST (RP_ADDR [BCNT_HI], EF_10)   ! IF
;       6619                    then PRINTB (F_10);
;       6620            end;
;       6621        IF .RP_ADDR [ENDCOD] EQLU %o'211'
```

```
;       6622        THEN                                                                    ! MSCP ONLINE comman
;       6623            begin
;       6624                    PRINTB (XX40);
;       6625                    if BIT_TST (RP_ADDR [BCNT_HI], EF_11)   ! IF
;       6626                        then PRINTB (F_11);
;       6627                    if BIT_TST (RP_ADDR [BCNT_HI], EF_12)   ! IF
;       6628                        then PRINTB (F_12);
;       6629                    if BIT_TST (RP_ADDR [BCNT_HI], EF_13)   ! IF
;       6630                        then PRINTB (F_13);
;       6631                    if BIT_TST (RP_ADDR [BCNT_HI], EF_14)   ! IF
;       6632                        then PRINTB (F_14);
;       6633                    if BIT_TST (RP_ADDR [BCNT_HI], EF_15)   ! IF
;       6634                        then PRINTB (F_15);
;       6635                    if BIT_TST (RP_ADDR [BCNT_HI], EF_16)   ! IF
;       6636                        then PRINTB (F_16);
;       6637                    if BIT_TST (RP_ADDR [BCNT_HI], EF_17)   ! IF
;       6638                        then PRINTB (F_17);
;       6639                    if BIT_TST (RP_ADDR [BCNT_HI], EF_18)   ! IF
;       6640                        then PRINTB (F_18);
;       6641                    if BIT_TST (RP_ADDR [BCNT_HI], EF_19)   ! IF
;       6642                        then PRINTB (F_19);
;       6643                    if BIT_TST (RP_ADDR [BCNT_HI], EF_20)   ! IF
;       6644                        then PRINTB (F_20);
;       6645                    if BIT_TST (RP_ADDR [BCNT_HI], EF_21)   ! IF
;       6646                        then PRINTB (F_21);
;       6647            end;
;       6648        END;
;       6649    [otherwise]:
;       6650        BEGIN                                                                   ! UNKNOWN CO
;       6651        PRINTB (XX38, .RP_ADDR [CONID]);                    ! PRINTS UNKNOWN CONNECTION ID
;       6652        SELECTU (.RP_ADDR [ENDCOD]) OF
;       6653                    SET
;       6654                    [%o'204']: PRINTB (EX_SCC);
;       6655                    [%o'211']: PRINTB (EX_ONL);             ! PRINTS THE COMMAND IF RECOGNIZED
;       6656                    [%o'220']: PRINTB (EX_ACC);
;       6657                    [%o'241']: PRINTB (EX_RD);
;       6658                    [%o'242']: PRINTB (EX_WRT);
;       6659                    [%o'211']: PRINTB (EX_ONL);
;       6660                    [%o'220']: PRINTB (EX_ACC);
;       6661                    [%o'241']: PRINTB (EX_RD);
;       6662                    [%o'242']: PRINTB (EX_WRT);
;       6663                    [OTHERWISE]: PRINTB (EX_OP, .RP_ADDR [ENDCOD]); ! PRINT ENDCODE VALUE
;       6664                    TES;
;       6665        if .RP_ADDR [CMDMOD] eql MD_CMP THEN PRINTB (XX20); ! PRINTS MODIFIER IF NECESSARY
;       6666        printb (xx15);
;       6667        PRINTB (ex_op, .RP_ADDR [STSCOD]);                  ! PRINTS STATUS CODE IN OCTAL FORM
;       6668        printb (xx16);
;       6669        PRINTB (ex_op, .RP_ADDR [SUBCOD]);                  ! PRINTS STATUS SUB-CODE IN OCTAL FORM
;       6670
;       6671
;       6672                    !+
;       6673                    !       THIS ROUTINE PRINTS (EXTENDED) BOTH BYTE COUNT FIELDS OF THE CURRENT
;       6674                    !       RETURN PACKET: THE BYTE COUNT FROM THE COMMAND ENVELOPE AND THE
```

```
                        !       ACTUAL NUMBER OF BYTES TRANSFERRED (FROM THE RESPONSE ENVELOPE).
                        !-

    IF .RP_ADDR [ENDCOD] EQLU %o'220' OR                        ! MSCP access command
       .RP_ADDR [ENDCOD] EQLU %o'241' OR                        ! mscp read
       .RP_ADDR [ENDCOD] EQLU %o'242'                   ! mscp write
    THEN
        begin
            printX (XX24, .rp_addr [CBCNT_LO]);                 ! 'BYTE COUNT IN COMMAND: XXXXXXX"
            PRINTX (XX25, .RP_ADDR [BCNT_LO]);                  ! FOR ANY "ACTUAL # OF BYTES TRANSFERRED: XXXXX."
            PRINTX (XX26, .RP_ADDR [BUFF_1], .RP_ADDR [BUFF_0]);    ! "I/O BUFFER DESCRIPTOR: XXXXXX XXXXXX"
            if BIT_TST (RP_ADDR [FLAGS], EF_0)      ! IF BAD BLOCK REPORTED
                then
                    PRINTB (XX21, .RP_ADDR [BBLK_LO])       ! 'BAD BLOCK REPORTED: XXXXXX."
                else
                    printX (xx22, .RP_ADDR [LBN_LO], .RP_ADDR [LBN_LO]);            ! "LBN: XXXXXX"
        end;
    IF .RP_ADDR [ENDCOD] EQLU %o'204' OR                        ! dup receive data
       .RP_ADDR [ENDCOD] EQLU %o'205'                   ! dup send data
    THEN
        begin
        PRINTX (XX25, .RP_ADDR [BCNT_LO]);                      ! FOR ANY "ACTUAL # OF BYTES TRANSFERRED: XXXXX."
        PRINTX (XX26, .RP_ADDR [BUFF_1], .RP_ADDR [BUFF_0]);    ! "I/O BUFFER DESCRIPTOR: XXXXXX XXXXXX"
        PRINTX (XX23, .CST_ADDR [.CUOFF + 3, D_DBN], .CST_ADDR [.CUOFF + 3, D_DBN]);            ! "DBN: XXXXXX."
        ems_dup ();                                             ! prints dup packet material
        end;
    END;
tes;                    !+
                        !       PRINTS RETURN OR MESSAGE PACKET IN A BLOCK OF OCTAL WORDS
                        !-

    !PRINTX (XX27);                                 ! "CONTENTS OF PACKET:"
    !EMS_BLK (.RP_ADDR, PKT_LEN);                   ! PRINT BLOCK OF WORDS AS LONG AS A MESSAGE PACKET INCASE A PACKET IS USED
                                                    ! INSTEAD OF A RETURN PACKET

    end;                        ! ROUTINE EMSCMD


010202                                  .PSECT  $CODE$,  RO
010202  000000G               EBH.TB1:.WORD   EBH.30
010204  000000G                       .WORD   EBH.44
010206  000000G                       .WORD   EBH.45
010210  000000G                       .WORD   EBH.46
010212  000000G                       .WORD   EBH.47
010214  000000G                       .WORD   EBH.48
010216  000000G                       .WORD   EBH.49


                                        .SBTTL  EMSCMD ERROR MESSAGE SUBROUTINES
000000  004137  000000G       EMSCMD::JSR     R1,$SAVE4                       ;                       6463
000004  013746  000000G               MOV     CDISK,-(SP)                     ;                       6482
000010  012746  000000G               MOV     #XX13,-(SP)
```

```
000014  012746  000002                    MOV     #2,-(SP)
000020  010600                            MOV     SP,R0                   ; SP,*
000022  104414                            TRAP    14
000024  013700  000000G                   MOV     RP.ADDR,R0              ;                       6484
000030  016016  000004                    MOV     4(R0),(SP)
000034  012746  000000G                   MOV     #XX35,-(SP)
000040  012746  000002                    MOV     #2,-(SP)
000044  010600                            MOV     SP,R0                   ; SP,*
000046  104415                            TRAP    15
000050  012716  000000G                   MOV     #XX29,(SP)              ;                       6485
000054  012746  000001                    MOV     #1,-(SP)
000060  010600                            MOV     SP,R0                   ; SP,*
000062  104415                            TRAP    15
000064  013700  000000G                   MOV     RP.ADDR,R0              ;                       6486
000070  116002  000002                    MOVB    2(R0),R2
000074  006202                            ASR     R2
000076  006202                            ASR     R2
000100  006202                            ASR     R2
000102  006202                            ASR     R2
000104  042702  177760                    BIC     #177760,R2
000110  012701  177777                    MOV     #-1,R1
000114  005702                            TST     R2
000116  001010                            BNE     1$
000120  005001                            CLR     R1
000122  012716  000000G                   MOV     #EX.SEQ,(SP)            ;                       6488
000126  012746  000001                    MOV     #1,-(SP)
000132  010600                            MOV     SP,R0                   ; SP,*
000134  104415                            TRAP    15
000136  005726                            TST     (SP)+
000140  020227  000001            1$:     CMP     R2,#1                   ;                       6486
000144  001010                            BNE     2$
000146  005001                            CLR     R1
000150  012716  000000G                   MOV     #EX.DGM,(SP)            ;                       6489
000154  012746  000001                    MOV     #1,-(SP)
000160  010600                            MOV     SP,R0                   ; SP,*
000162  104415                            TRAP    15
000164  005726                            TST     (SP)+
000166  020227  000002            2$:     CMP     R2,#2                   ;                       6486
000172  001010                            BNE     3$
000174  005001                            CLR     R1
000176  012716  000000G                   MOV     #EX.CRD,(SP)            ;                       6490
000202  012746  000001                    MOV     #1,-(SP)
000206  010600                            MOV     SP,R0                   ; SP,*
000210  104415                            TRAP    15
000212  005726                            TST     (SP)+
000214  020227  000015            3$:     CMP     R2,#15                  ;                       6486
000220  001010                            BNE     4$
000222  005001                            CLR     R1
000224  012716  000000G                   MOV     #EX.MTN,(SP)            ;                       6491
000230  012746  000001                    MOV     #1,-(SP)
000234  010600                            MOV     SP,R0                   ; SP,*
000236  104415                            TRAP    15
000240  005726                            TST     (SP)+
```

```
000242  005701                  4$:     TST     R1                              ;                               6486
000244  001422                          BEQ     5$
000246  013700  000000G                 MOV     RP.ADDR,R0                      ;                               6492
000252  116001  000002                  MOVB    2(R0),R1
000256  006201                          ASR     R1
000260  006201                          ASR     R1
000262  006201                          ASR     R1
000264  006201                          ASR     R1
000266  042701  177760                  BIC     #177760,R1
000272  010116                          MOV     R1,(SP)
000274  012746  000000G                 MOV     #XX37,-(SP)
000300  012746  000002                  MOV     #2,-(SP)
000304  010600                          MOV     SP,R0                           ; SP,*
000306  104415                          TRAP    15
000310  022626                          CMP     (SP)+,(SP)+
000312  012716  000000G         5$:     MOV     #XX17,(SP)                      ;                               6495
000316  012746  000001                  MOV     #1,-(SP)
000322  010600                          MOV     SP,R0                           ; SP,*
000324  104414                          TRAP    14
000326  013700  000000G                 MOV     RP.ADDR,R0                      ;                               6497
000332  005004                          CLR     R4
000334  156004  000003                  BISB    3(R0),R4
000340  012703  177777                  MOV     #-1,R3
000344  020427  000002                  CMP     R4,#2
000350  001402                          BEQ     6$
000352  000137  011706'                 JMP     26$
000356  005003                  6$:     CLR     R3
000360  012716  000000G                 MOV     #XX18,(SP)                      ;                               6501
000364  012746  000001                  MOV     #1,-(SP)
000370  010600                          MOV     SP,R0                           ; SP,*
000372  104414                          TRAP    14
000374  013700  000000G                 MOV     RP.ADDR,R0                      ;                               6502
000400  005002                          CLR     R2
000402  156002  000014                  BISB    14(R0),R2
000406  012701  177777                  MOV     #-1,R1
000412  020227  000201                  CMP     R2,#201
000416  001010                          BNE     7$
000420  005001                          CLR     R1
000422  012716  000000G                 MOV     #EX.GDS,(SP)                    ;                               6504
000426  012746  000001                  MOV     #1,-(SP)
000432  010600                          MOV     SP,R0                           ; SP,*
000434  104414                          TRAP    14
000436  005726                          TST     (SP)+
000440  020227  000202          7$:     CMP     R2,#202                         ;                               6502
000444  001010                          BNE     8$
000446  005001                          CLR     R1
000450  012716  000000G                 MOV     #EX.ESP,(SP)                    ;                               6505
000454  012746  000001                  MOV     #1,-(SP)
000460  010600                          MOV     SP,R0                           ; SP,*
000462  104414                          TRAP    14
000464  005726                          TST     (SP)+
000466  020227  000203          8$:     CMP     R2,#203                         ;                               6502
000472  001010                          BNE     9$
```

```
000474  005001                          CLR     R1
000476  012716  000000G                 MOV     #EX.ELP,(SP)            ;                       6506
000502  012746  000001                  MOV     #1,-(SP)
000506  010600                          MOV     SP,R0                   ; SP,*
000510  104414                          TRAP    14
000512  005726                          TST     (SP)+
000514  020227  000204          9$:     CMP     R2,#204                 ;                       6502
000520  001010                          BNE     10$
000522  005001                          CLR     R1
000524  012716  000000G                 MOV     #EX.RCD,(SP)            ;                       6507
000530  012746  000001                  MOV     #1,-(SP)
000534  010600                          MOV     SP,R0                   ; SP,*
000536  104414                          TRAP    14
000540  005726                          TST     (SP)+
000542  020227  000205          10$:    CMP     R2,#205                 ;                       6502
000546  001010                          BNE     11$
000550  005001                          CLR     R1
000552  012716  000000G                 MOV     #EX.SDD,(SP)            ;                       6508
000556  012746  000001                  MOV     #1,-(SP)
000562  010600                          MOV     SP,R0                   ; SP,*
000564  104414                          TRAP    14
000566  005726                          TST     (SP)+
000570  020227  000206          11$:    CMP     R2,#206                 ;                       6502
000574  001010                          BNE     12$
000576  005001                          CLR     R1
000600  012716  000000G                 MOV     #EX.ABP,(SP)            ;                       6509
000604  012746  000001                  MOV     #1,-(SP)
000610  010600                          MOV     SP,R0                   ; SP,*
000612  104414                          TRAP    14
000614  005726                          TST     (SP)+
000616  005701                  12$:    TST     R1                      ;                       6502
000620  001414                          BEQ     13$
000622  013700  000000G                 MOV     RP.ADDR,R0              ;                       6510
000626  005016                          CLR     (SP)
000630  116016  000014                  MOVB    14(R0),(SP)
000634  012746  000000G                 MOV     #EX.OP,-(SP)
000640  012746  000002                  MOV     #2,-(SP)
000644  010600                          MOV     SP,R0                   ; SP,*
000646  104414                          TRAP    14
000650  022626                          CMP     (SP)+,(SP)+
000652  012716  000000G          13$:    MOV     #XX15,(SP)              ;                       6512
000656  012746  000001                  MOV     #1,-(SP)
000662  010600                          MOV     SP,R0                   ; SP,*
000664  104414                          TRAP    14
000666  013700  000000G                 MOV     RP.ADDR,R0              ;                       6513
000672  116000  000016                  MOVB    16(R0),R0
000676  042700  177740                  BIC     #177740,R0
000702  020027  000007                  CMP     R0,#7
000706  101010                          BHI     14$
000710  006300                          ASL     R0                      ;                       6514
000712  016016  010202'                 MOV     EBH.TB1(R0),(SP)
000716  012746  000001                  MOV     #1,-(SP)
000722  010600                          MOV     SP,R0                   ; SP,*
```

```
000724  104414                          TRAP    14
000726  000410                          BR      15$                             ;                               6513
000730  010016                  14$:     MOV     R0,(SP)                         ;                               6515
000732  012746  000000G                  MOV     #EX.OP,-(SP)
000736  012746  000002                   MOV     #2,-(SP)
000742  010600                          MOV     SP,R0                           ; SP,*
000744  104414                          TRAP    14
000746  005726                          TST     (SP)+
000750  013700  000000G          15$:    MOV     RP.ADDR,R0                      ;                               6517
000754  126027  000014  000204          CMPB    14(R0),#204
000762  001404                          BEQ     16$
000764  126027  000014  000205          CMPB    14(R0),#205                     ;                               6518
000772  001032                          BNE     17$
000774  013700  000000G          16$:    MOV     RP.ADDR,R0                      ;                               6521
001000  016016  000020                   MOV     20(R0),(SP)
001004  012746  000000G                  MOV     #XX25,-(SP)
001010  012746  000002                   MOV     #2,-(SP)
001014  010600                          MOV     SP,R0                           ; SP,*
001016  104415                          TRAP    15
001020  013700  000000G                  MOV     RP.ADDR,R0                      ;                               6522
001024  016016  000024                   MOV     24(R0),(SP)
001030  016046  000026                   MOV     26(R0),-(SP)
001034  012746  000000G                  MOV     #XX26,-(SP)
001040  012746  000003                   MOV     #3,-(SP)
001044  010600                          MOV     SP,R0                           ; SP,*
001046  104415                          TRAP    15
001050  004737  000000V                  JSR     PC,EMS.DUP                      ;                               6523
001054  062706  000012                   ADD     #12,SP                          ;                               6520
001060  013700  000000G          17$:    MOV     RP.ADDR,R0                      ;                               6525
001064  126027  000014  000201          CMPB    14(R0),#201
001072  001062                          BNE     21$
001074  032760  000400  000022          BIT     #400,22(R0)                     ;                               6528
001102  001407                          BEQ     18$
001104  012716  000000G                  MOV     #DF.0,(SP)                      ;                               6529
001110  012746  000001                   MOV     #1,-(SP)
001114  010600                          MOV     SP,R0                           ; SP,*
001116  104414                          TRAP    14
001120  005726                          TST     (SP)+
001122  013700  000000G          18$:    MOV     RP.ADDR,R0                      ;                               6530
001126  032760  001000  000022          BIT     #1000,22(R0)
001134  001407                          BEQ     19$
001136  012716  000000G                  MOV     #DF.1,(SP)                      ;                               6531
001142  012746  000001                   MOV     #1,-(SP)
001146  010600                          MOV     SP,R0                           ; SP,*
001150  104414                          TRAP    14
001152  005726                          TST     (SP)+
001154  013700  000000G          19$:    MOV     RP.ADDR,R0                      ;                               6532
001160  032760  002000  000022          BIT     #2000,22(R0)
001166  001407                          BEQ     20$
001170  012716  000000G                  MOV     #DF.2,(SP)                      ;                               6533
001174  012746  000001                   MOV     #1,-(SP)
001200  010600                          MOV     SP,R0                           ; SP,*
001202  104414                          TRAP    14
```

```
001204  005726                          TST     (SP)+
001206  013700  000000G           20$:  MOV     RP.ADDR,R0              ;                                       6534
001212  032760  004000  000022        BIT     #4000,22(R0)
001220  001407                        BEQ     21$
001222  012716  000000G               MOV     #DF.3,(SP)              ;                                       6535
001226  012746  000001                MOV     #1,-(SP)
001232  010600                        MOV     SP,R0                   ; SP,*
001234  104414                        TRAP    14
001236  005726                          TST     (SP)+
001240  013700  000000G           21$:  MOV     RP.ADDR,R0              ;                                       6538
001244  126027  000014  000203        CMPB    14(R0),#203
001252  001062                        BNE     25$
001254  032760  000400  000022        BIT     #400,22(R0)             ;                                       6541
001262  001407                        BEQ     22$
001264  012716  000000G               MOV     #DF.4,(SP)              ;                                       6542
001270  012746  000001                MOV     #1,-(SP)
001274  010600                        MOV     SP,R0                   ; SP,*
001276  104414                        TRAP    14
001300  005726                          TST     (SP)+
001302  013700  000000G           22$:  MOV     RP.ADDR,R0              ;                                       6543
001306  032760  001000  000022        BIT     #1000,22(R0)
001314  001407                        BEQ     23$
001316  012716  000000G               MOV     #DF.5,(SP)              ;                                       6544
001322  012746  000001                MOV     #1,-(SP)
001326  010600                        MOV     SP,R0                   ; SP,*
001330  104414                        TRAP    14
001332  005726                          TST     (SP)+
001334  013700  000000G           23$:  MOV     RP.ADDR,R0              ;                                       6545
001340  032760  002000  000022        BIT     #2000,22(R0)
001346  001407                        BEQ     24$
001350  012716  000000G               MOV     #DF.6,(SP)              ;                                       6546
001354  012746  000001                MOV     #1,-(SP)
001360  010600                        MOV     SP,R0                   ; SP,*
001362  104414                        TRAP    14
001364  005726                          TST     (SP)+
001366  013700  000000G           24$:  MOV     RP.ADDR,R0              ;                                       6547
001372  032760  004000  000022        BIT     #4000,22(R0)
001400  001407                        BEQ     25$
001402  012716  000000G               MOV     #DF.7,(SP)              ;                                       6548
001406  012746  000001                MOV     #1,-(SP)
001412  010600                        MOV     SP,R0                   ; SP,*
001414  104414                        TRAP    14
001416  005726                          TST     (SP)+
001420  013700  000000G           25$:  MOV     CUOFF,R0                ;                                       6550
001424  006300                        ASL     R0
001426  063700  000000G               ADD     CST.ADDR,R0
001432  005016                        CLR     (SP)
001434  116016  000006                MOVB    6(R0),(SP)
001440  005046                        CLR     -(SP)
001442  116016  000006                MOVB    6(R0),(SP)
001446  012746  000000G               MOV     #XX23,-(SP)
001452  012746  000003                MOV     #3,-(SP)
001456  010600                        MOV     SP,R0                   ; SP,*
```

```
001460  104415                          TRAP    15
001462  062706  000014                  ADD     #14,SP                  ;                               6500
001466  005704                  26$:    TST     R4                      ;                               6497
001470  001402                          BEQ     27$
001472  000137  013742'                  JMP     66$
001476  005003                  27$:    CLR     R3
001500  012716  000000G                 MOV     #XX19,(SP)              ;                               6554
001504  012746  000001                  MOV     #1,-(SP)
001510  010600                          MOV     SP,R0                   ; SP,*
001512  104414                          TRAP    14
001514  013700  000000G                 MOV     RP.ADDR,R0              ;                               6555
001520  005002                          CLR     R2
001522  156002  000014                  BISB    14(R0),R2
001526  012701  177777                  MOV     #-1,R1
001532  020227  000204                  CMP     R2,#204
001536  001010                          BNE     28$
001540  005001                          CLR     R1
001542  012716  000000G                 MOV     #EX.SCC,(SP)            ;                               6557
001546  012746  000001                  MOV     #1,-(SP)
001552  010600                          MOV     SP,R0                   ; SP,*
001554  104414                          TRAP    14
001556  005726                          TST     (SP)+
001560  020227  000211          28$:    CMP     R2,#211                 ;                               6555
001564  001010                          BNE     29$
001566  005001                          CLR     R1
001570  012716  000000G                 MOV     #EX.ONL,(SP)            ;                               6558
001574  012746  000001                  MOV     #1,-(SP)
001600  010600                          MOV     SP,R0                   ; SP,*
001602  104414                          TRAP    14
001604  005726                          TST     (SP)+
001606  020227  000220          29$:    CMP     R2,#220                 ;                               6555
001612  001010                          BNE     30$
001614  005001                          CLR     R1
001616  012716  000000G                 MOV     #EX.ACC,(SP)            ;                               6559
001622  012746  000001                  MOV     #1,-(SP)
001626  010600                          MOV     SP,R0                   ; SP,*
001630  104414                          TRAP    14
001632  005726                          TST     (SP)+
001634  020227  000241          30$:    CMP     R2,#241                 ;                               6555
001640  001010                          BNE     31$
001642  005001                          CLR     R1
001644  012716  000000G                 MOV     #EX.RD,(SP)             ;                               6560
001650  012746  000001                  MOV     #1,-(SP)
001654  010600                          MOV     SP,R0                   ; SP,*
001656  104414                          TRAP    14
001660  005726                          TST     (SP)+
001662  020227  000242          31$:    CMP     R2,#242                 ;                               6555
001666  001010                          BNE     32$
001670  005001                          CLR     R1
001672  012716  000000G                 MOV     #EX.WRT,(SP)            ;                               6561
001676  012746  000001                  MOV     #1,-(SP)
001702  010600                          MOV     SP,R0                   ; SP,*
001704  104414                          TRAP    14
```

```
001706  005726                          TST     (SP)+
001710  005701                  32$:    TST     R1                      ;                       6555
001712  001414                          BEQ     33$
001714  013700  000000G                 MOV     RP.ADDR,R0              ;                       6562
001720  005016                          CLR     (SP)
001722  116016  000014                  MOVB    14(R0),(SP)
001726  012746  000000G                 MOV     #EX.OP,-(SP)
001732  012746  000002                  MOV     #2,-(SP)
001736  010600                          MOV     SP,R0                   ; SP,*
001740  104414                          TRAP    14
001742  022626                          CMP     (SP)+,(SP)+
001744  013700  000000G         33$:    MOV     RP.ADDR,R0              ;                       6564
001750  026027  000012  040000          CMP     12(R0),#40000
001756  001007                          BNE     34$
001760  012716  000000G                 MOV     #XX20,(SP)
001764  012746  000001                  MOV     #1,-(SP)
001770  010600                          MOV     SP,R0                   ; SP,*
001772  104414                          TRAP    14
001774  005726                          TST     (SP)+
001776  012716  000000G         34$:    MOV     #XX15,(SP)              ;                       6566
002002  012746  000001                  MOV     #1,-(SP)
002006  010600                          MOV     SP,R0                   ; SP,*
002010  104414                          TRAP    14
002012  013700  000000G                 MOV     ST.CODE,R0              ;                       6567
002016  001413                          BEQ     35$
002020  020027  000013                  CMP     R0,#13                  ;                       6568
002024  101010                          BHI     35$
002026  006300                          ASL     R0                      ;                       6570
002030  016016  177776G                 MOV     ERR.COD-2(R0),(SP)
002034  012746  000001                  MOV     #1,-(SP)
002040  010600                          MOV     SP,R0                   ; SP,*
002042  104414                          TRAP    14
002044  000422                          BR      37$                     ;                       6567
002046  020027  000037         35$:    CMP     R0,#37                  ;                       6573
002052  001007                          BNE     36$
002054  013716  000026G                 MOV     ERR.COD+26,(SP)         ;                       6575
002060  012746  000001                  MOV     #1,-(SP)
002064  010600                          MOV     SP,R0                   ; SP,*
002066  104414                          TRAP    14
002070  000410                          BR      37$                     ;                       6573
002072  010016                  36$:    MOV     R0,(SP)                 ;                       6577
002074  012746  000000G                 MOV     #EX.OP,-(SP)
002100  012746  000002                  MOV     #2,-(SP)
002104  010600                          MOV     SP,R0                   ; SP,*
002106  104414                          TRAP    14
002110  005726                          TST     (SP)+
002112  004737  007472'         37$:    JSR     PC,EMS.SBC              ;                       6579
002116  013700  000000G                 MOV     RP.ADDR,R0              ;                       6581
002122  116000  000014                  MOVB    14(R0),R0
002126  042700  177400                  BIC     #177400,R0
002132  020027  000220                  CMP     R0,#220
002136  001406                          BEQ     38$
002140  020027  000241                  CMP     R0,#241                 ;                       6582
```

```
002144  001403                          BEQ     38$
002146  020027  000242                  CMP     R0,#242                 ;                                       6583
002152  001147                          BNE     44$
002154  013700  000000G         38$:    MOV     RP.ADDR,R0              ;                                       6586
002160  016016  000044                  MOV     44(R0),(SP)
002164  012746  000000G                 MOV     #XX24,-(SP)
002170  012746  000002                  MOV     #2,-(SP)
002174  010600                          MOV     SP,R0                   ; SP,*
002176  104415                          TRAP    15
002200  013700  000000G                 MOV     RP.ADDR,R0              ;                                       6587
002204  016016  000020                  MOV     20(R0),(SP)
002210  012746  000000G                 MOV     #XX25,-(SP)
002214  012746  000002                  MOV     #2,-(SP)
002220  010600                          MOV     SP,R0                   ; SP,*
002222  104415                          TRAP    15
002224  013700  000000G                 MOV     RP.ADDR,R0              ;                                       6588
002230  016016  000024                  MOV     24(R0),(SP)
002234  016046  000026                  MOV     26(R0),-(SP)
002240  012746  000000G                 MOV     #XX26,-(SP)
002244  012746  000003                  MOV     #3,-(SP)
002250  010600                          MOV     SP,R0                   ; SP,*
002252  104415                          TRAP    15
002254  013700  000000G                 MOV     RP.ADDR,R0              ;                                       6589
002260  005760  000014                  TST     14(R0)
002264  100011                          BPL     39$
002266  016016  000040                  MOV     40(R0),(SP)             ;                                       6591
002272  012746  000000G                 MOV     #XX21,-(SP)
002276  012746  000002                  MOV     #2,-(SP)
002302  010600                          MOV     SP,R0                   ; SP,*
002304  104414                          TRAP    14
002306  000412                          BR      40$                     ;                                       6589
002310  016016  000050         39$:    MOV     50(R0),(SP)             ;                                       6593
002314  011646                          MOV     (SP),-(SP)
002316  012746  000000G                 MOV     #XX22,-(SP)
002322  012746  000003                  MOV     #3,-(SP)
002326  010600                          MOV     SP,R0                   ; SP,*
002330  104415                          TRAP    15
002332  005726                          TST     (SP)+
002334  012716  000000G         40$:    MOV     #XX41,(SP)              ;                                       6594
002340  012746  000001                  MOV     #1,-(SP)
002344  010600                          MOV     SP,R0                   ; SP,*
002346  104414                          TRAP    14
002350  013700  000000G                 MOV     RP.ADDR,R0              ;                                       6595
002354  032760  040000  000014         BIT     #40000,14(R0)
002362  001407                          BEQ     41$
002364  012716  000000G                 MOV     #F.1,(SP)               ;                                       6596
002370  012746  000001                  MOV     #1,-(SP)
002374  010600                          MOV     SP,R0                   ; SP,*
002376  104414                          TRAP    14
002400  005726                          TST     (SP)+
002402  013700  000000G         41$:    MOV     RP.ADDR,R0              ;                                       6597
002406  032760  020000  000014         BIT     #20000,14(R0)
002414  001407                          BEQ     42$
```

```
002416  012716  000000G                        MOV     #F.2,(SP)                       ;                                       6598
002422  012746  000001                        MOV     #1,-(SP)
002426  010600                                MOV     SP,R0                           ; SP,*
002430  104414                                TRAP    14
002432  005726                                TST     (SP)+
002434  013700  000000G                42$:    MOV     RP.ADDR,R0                      ;                                       6599
002440  032760  010000  000014                BIT     #10000,14(R0)
002446  001407                                BEQ     43$
002450  012716  000000G                        MOV     #F.3,(SP)                       ;                                       6600
002454  012746  000001                        MOV     #1,-(SP)
002460  010600                                MOV     SP,R0                           ; SP,*
002462  104414                                TRAP    14
002464  005726                                TST     (SP)+
002466  062706  000024                43$:    ADD     #24,SP                          ;                                       6585
002472  013700  000000G                44$:    MOV     RP.ADDR,R0                      ;                                       6602
002476  126027  000014  000204                CMPB    14(R0),#204
002504  001144                                BNE     52$
002506  012716  000000G                        MOV     #XX39,(SP)                      ;                                       6605
002512  012746  000001                        MOV     #1,-(SP)
002516  010600                                MOV     SP,R0                           ; SP,*
002520  104414                                TRAP    14
002522  013700  000000G                        MOV     RP.ADDR,R0                      ;                                       6606
002526  105760  000022                        TSTB    22(R0)
002532  100007                                BPL     45$
002534  012716  000000G                        MOV     #F.4,(SP)                       ;                                       6607
002540  012746  000001                        MOV     #1,-(SP)
002544  010600                                MOV     SP,R0                           ; SP,*
002546  104414                                TRAP    14
002550  005726                                TST     (SP)+
002552  013700  000000G                45$:    MOV     RP.ADDR,R0                      ;                                       6608
002556  032760  000100  000022                BIT     #100,22(R0)
002564  001407                                BEQ     46$
002566  012716  000000G                        MOV     #F.5,(SP)                       ;                                       6609
002572  012746  000001                        MOV     #1,-(SP)
002576  010600                                MOV     SP,R0                           ; SP,*
002600  104414                                TRAP    14
002602  005726                                TST     (SP)+
002604  013700  000000G                46$:    MOV     RP.ADDR,R0                      ;                                       6610
002610  032760  000040  000022                BIT     #40,22(R0)
002616  001407                                BEQ     47$
002620  012716  000000G                        MOV     #F.6,(SP)                       ;                                       6611
002624  012746  000001                        MOV     #1,-(SP)
002630  010600                                MOV     SP,R0                           ; SP,*
002632  104414                                TRAP    14
002634  005726                                TST     (SP)+
002636  013700  000000G                47$:    MOV     RP.ADDR,R0                      ;                                       6612
002642  032760  000020  000022                BIT     #20,22(R0)
002650  001407                                BEQ     48$
002652  012716  000000G                        MOV     #F.7,(SP)                       ;                                       6613
002656  012746  000001                        MOV     #1,-(SP)
002662  010600                                MOV     SP,R0                           ; SP,*
002664  104414                                TRAP    14
002666  005726                                TST     (SP)+
```

```
002670  013700  000000G                 48$:    MOV     RP.ADDR,R0                      ;                                       6614
002674  016000  000022                             MOV     22(R0),R0
002700  042700  077777                             BIC     #77777,R0
002704  020027  100000                             CMP     R0,#-100000
002710  001007                                     BNE     49$
002712  012716  000000G                           MOV     #F.8,(SP)                       ;                                       6615
002716  012746  000001                             MOV     #1,-(SP)
002722  010600                                     MOV     SP,R0                           ; SP,*
002724  104414                                     TRAP    14
002726  005726                                     TST     (SP)+
002730  013700  000000G                 49$:    MOV     RP.ADDR,R0                      ;                                       6616
002734  032760  000002  000022                     BIT     #2,22(R0)
002742  001407                                     BEQ     50$
002744  012716  000000G                           MOV     #F.9,(SP)                       ;                                       6617
002750  012746  000001                             MOV     #1,-(SP)
002754  010600                                     MOV     SP,R0                           ; SP,*
002756  104414                                     TRAP    14
002760  005726                                     TST     (SP)+
002762  013700  000000G                 50$:    MOV     RP.ADDR,R0                      ;                                       6618
002766  032760  000001  000022                     BIT     #1,22(R0)
002774  001407                                     BEQ     51$
002776  012716  000C00G                           MOV     #F.10,(SP)                      ;                                       6619
003002  012746  000001                             MOV     #1,-(SP)
003006  010600                                     MOV     SP,R0                           ; SP,*
003010  104414                                     TRAP    14
003012  005726                                     TST     (SP)+
003014  005726                          51$:    TST     (SP)+                           ;                                       6604
003016  013700  000000G                 52$:    MOV     RP.ADDR,R0                      ;                                       6621
003022  126027  000014  000211                     CMPB    14(R0),#211
003030  001402                                     BEQ     53$
003032  000137  013736'                           JMP     65$
003036  012716  000000G                 53$:    MOV     #XX40,(SP)                      ;                                       6624
003042  012746  000001                             MOV     #1,-(SP)
003046  010600                                     MOV     SP,R0                           ; SP,*
003050  104414                                     TRAP    14
003052  013700  000000G                           MOV     RP.ADDR,R0                      ;                                       6625
003056  032760  000001  000022                     BIT     #1,22(R0)
003064  001407                                     BEQ     54$
003066  012716  000000G                           MOV     #F.11,(SP)                      ;                                       6626
003072  012746  000001                             MOV     #1,-(SP)
003076  010600                                     MOV     SP,R0                           ; SP,*
003100  104414                                     TRAP    14
003102  005726                                     TST     (SP)+
003104  013700  000000G                 54$:    MOV     RP.ADDR,R0                      ;                                       6627
003110  032760  000002  000022                     BIT     #2,22(R0)
003116  001407                                     BEQ     55$
003120  012716  000000G                           MOV     #F.12,(SP)                      ;                                       6628
003124  012746  000001                             MOV     #1,-(SP)
003130  010600                                     MOV     SP,R0                           ; SP,*
003132  104414                                     TRAP    14
003134  005726                                     TST     (SP)+
003136  013700  000000G                 55$:    MOV     RP.ADDR,R0                      :                                       6629
003142  016000  000022                             MOV     22(R0),R0
```

```
003146  042700  077777                      BIC     #77777,R0
003152  020027  100000                      CMP     R0,#-100000
003156  001007                              BNE     56$
003160  012716  000000G                     MOV     #F.13,(SP)              ;                                       6630
003164  012746  000001                      MOV     #1,-(SP)
003170  010600                              MOV     SP,R0                   ; SP,*
003172  104414                              TRAP    14
003174  005726                              TST     (SP)+
003176  013700  000000G               56$:  MOV     RP.ADDR,R0              ;                                       6631
003202  032760  040000  000022             BIT     #40000,22(R0)
003210  001407                              BEQ     57$
003212  012716  000000G                     MOV     #F.14,(SP)              ;                                       6632
003216  012746  000001                      MOV     #1,-(SP)
003222  010600                              MOV     SP,R0                   ; SP,*
003224  104414                              TRAP    14
003226  005726                              TST     (SP)+
003230  013700  000000G               57$:  MOV     RP.ADDR,R0              ;                                       6633
003234  105760  000022                      TSTB    22(R0)
003240  100007                              BPL     58$
003242  012716  000000G                     MOV     #F.15,(SP)              ;                                       6634
003246  012746  000001                      MOV     #1,-(SP)
003252  010600                              MOV     SP,R0                   ; SP,*
003254  104414                              TRAP    14
003256  005726                              TST     (SP)+
003260  013700  000000G               58$:  MOV     RP.ADDR,R0              ;                                       6635
003264  032760  004000  000022             BIT     #4000,22(R0)
003272  001407                              BEQ     59$
003274  012716  000000G                     MOV     #F.16,(SP)              ;                                       6636
003300  012746  000001                      MOV     #1,-(SP)
003304  010600                              MOV     SP,R0                   ; SP,*
003306  104414                              TRAP    14
003310  005726                              TST     (SP)+
003312  013700  000000G               59$:  MOV     RP.ADDR,R0              ;                                       6637
003316  032760  002000  000022             BIT     #2000,22(R0)
003324  001407                              BEQ     60$
003326  012716  000000G                     MOV     #F.17,(SP)              ;                                       6638
003332  012746  000001                      MOV     #1,-(SP)
003336  010600                              MOV     SP,R0                   ; SP,*
003340  104414                              TRAP    14
003342  005726                              TST     (SP)+
003344  013700  000000G               60$:  MOV     RP.ADDR,R0              ;                                       6639
003350  032760  000100  000022             BIT     #100,22(R0)
003356  001407                              BEQ     61$
003360  012716  000000G                     MOV     #F.18,(SP)              ;                                       6640
003364  012746  000001                      MOV     #1,-(SP)
003370  010600                              MOV     SP,R0                   ; SP,*
003372  104414                              TRAP    14
003374  005726                              TST     (SP)+
003376  013700  000000G               61$:  MOV     RP.ADDR,R0              ;                                       6641
003402  032760  020000  000022             BIT     #20000,22(R0)
003410  001407                              BEQ     62$
003412  012716  000000G                     MOV     #F.19,(SP)              :                                       6642
003416  012746  000001                      MOV     #1,-(SP)
```

```
003422  010600                          MOV     SP,R0                   ; SP,*
003424  104414                          TRAP    14
003426  005726                          TST     (SP)+
003430  013700  000000G          62$:   MOV     RP.ADDR,R0              ;                       6643
003434  032760  010000  000022         BIT     #10000,22(R0)
003442  001407                          BEQ     63$
003444  012716  000000G                 MOV     #F.20,(SP)              ;                       6644
003450  012746  000001                  MOV     #1,-(SP)
003454  010600                          MOV     SP,R0                   ; SP,*
003456  104414                          TRAP    14
003460  005726                          TST     (SP)+
003462  013700  000000G          63$:   MOV     RP.ADDR,R0              ;                       6645
003466  032760  000004  000022         BIT     #4,22(R0)
003474  001407                          BEQ     64$
003476  012716  000000G                 MOV     #F.21,(SP)              ;                       6646
003502  012746  000001                  MOV     #1,-(SP)
003506  010600                          MOV     SP,R0                   ; SP,*
003510  104414                          TRAP    14
003512  005726                          TST     (SP)+
003514  005726                  64$:    TST     (SP)+                   ;                       6623
003516  062706  000006          65$:    ADD     #6,SP                   ;                       6553
003522  005703                  66$:    TST     R3                      ;                       6497
003524  001002                          BNE     67$
003526  000137  015136'                 JMP     85$
003532  013700  000000G          67$:   MOV     RP.ADDR,R0              ;                       6651
003536  005016                          CLR     (SP)
003540  116016  000003                  MOVB    3(R0),(SP)
003544  012746  000000G                 MOV     #XX38,-(SP)
003550  012746  000002                  MOV     #2,-(SP)
003554  010600                          MOV     SP,R0                   ; SP,*
003556  104414                          TRAP    14
003560  013700  000000G                 MOV     RP.ADDR,R0              ;                       6652
003564  005002                          CLR     R2
003566  156002  000014                  BISB    14(R0),R2
003572  012701  177777                  MOV     #-1,R1
003576  020227  000204                  CMP     R2,#204
003602  001010                          BNE     68$
003604  005001                          CLR     R1
003606  012716  000000G                 MOV     #EX.SCC,(SP)            ;                       6654
003612  012746  000001                  MOV     #1,-(SP)
003616  010600                          MOV     SP,R0                   ; SP,*
003620  104414                          TRAP    14
003622  005726                          TST     (SP)+
003624  020227  000211          68$:    CMP     R2,#211                 ;                       6652
003630  001010                          BNE     69$
003632  005001                          CLR     R1
003634  012716  000000G                 MOV     #EX.ONL,(SP)            ;                       6655
003640  012746  000001                  MOV     #1,-(SP)
003644  010600                          MOV     SP,R0                   ; SP,*
003646  104414                          TRAP    14
003650  005726                          TST     (SP)+
003652  020227  000220          69$:    CMP     R2,#220                 ;                       6652
003656  001010                          BNE     70$
```

```
003660  005001                          CLR     R1
003662  012716  000000G                 MOV     #EX.ACC,(SP)                    ;                               6656
003666  012746  000001                  MOV     #1,-(SP)
003672  010600                          MOV     SP,R0                           ; SP,*
003674  104414                          TRAP    14
003676  005726                          TST     (SP)+
003700  020227  000241          70$:    CMP     R2,#241                         ;                               6652
003704  001010                          BNE     71$
003706  005001                          CLR     R1
003710  012716  000000G                 MOV     #EX.RD,(SP)                     ;                               6657
003714  012746  000001                  MOV     #1,-(SP)
003720  010600                          MOV     SP,R0                           ; SP,*
003722  104414                          TRAP    14
003724  005726                          TST     (SP)+
003726  020227  000242          71$:    CMP     R2,#242                         ;                               6652
003732  001010                          BNE     72$
003734  005001                          CLR     R1
003736  012716  000000G                 MOV     #EX.WRT,(SP)                    ;                               6658
003742  012746  000001                  MOV     #1,-(SP)
003746  010600                          MOV     SP,R0                           ; SP,*
003750  104414                          TRAP    14
003752  005726                          TST     (SP)+
003754  020227  000211          72$:    CMP     R2,#211                         ;                               6652
003760  001010                          BNE     73$
003762  005001                          CLR     R1
003764  012716  000000G                 MOV     #EX.ONL,(SP)                    ;                               6659
003770  012746  000001                  MOV     #1,-(SP)
003774  010600                          MOV     SP,R0                           ; SP,*
003776  104414                          TRAP    14
004000  005726                          TST     (SP)+
004002  020227  000220          73$:    CMP     R2,#220                         ;                               6652
004006  001010                          BNE     74$
004010  005001                          CLR     R1
004012  012716  000000G                 MOV     #EX.ACC,(SP)                    ;                               6660
004016  012746  000001                  MOV     #1,-(SP)
004022  010600                          MOV     SP,R0                           ; SP,*
004024  104414                          TRAP    14
004026  005726                          TST     (SP)+
004030  020227  000241          74$:    CMP     R2,#241                         ;                               6652
004034  001010                          BNE     75$
004036  005001                          CLR     R1
004040  012716  000000G                 MOV     #EX.RD,(SP)                     ;                               6661
004044  012746  000001                  MOV     #1,-(SP)
004050  010600                          MOV     SP,R0                           ; SP,*
004052  104414                          TRAP    14
004054  005726                          TST     (SP)+
004056  020227  000242          75$:    CMP     R2,#242                         ;                               6652
004062  001010                          BNE     76$
004064  005001                          CLR     R1
004066  012716  000000G                 MOV     #EX.WRT,(SP)                    ;                               6662
004072  012746  000001                  MOV     #1,-(SP)
004076  010600                          MOV     SP,R0                           ; SP,*
004100  104414                          TRAP    14
```

```
004102  005726                          TST     (SP)+
004104  005701                  76$:    TST     R1                              ;                               6652
004106  001414                          BEQ     77$
004110  013700  000000G                 MOV     RP.ADDR,R0                      ;                               6663
004114  005016                          CLR     (SP)
004116  116016  000014                  MOVB    14(R0),(SP)
004122  012746  000000G                 MOV     #EX.OP,-(SP)
004126  012746  000002                  MOV     #2,-(SP)
004132  010600                          MOV     SP,R0                           ; SP,*
004134  104414                          TRAP    14
004136  022626                          CMP     (SP)+,(SP)+
004140  013700  000000G                 MOV     RP.ADDR,R0                      ;                               6665
004144  026027  000012  040000          CMP     12(R0),#40000
004152  001007                          BNE     78$
004154  012716  000000G                 MOV     #XX20,(SP)
004160  012746  000001                  MOV     #1,-(SP)
004164  010600                          MOV     SP,R0                           ; SP,*
004166  104414                          TRAP    14
004170  005726                          TST     (SP)+
004172  012716  000000G                 MOV     #XX15,(SP)                      ;                               6666
004176  012746  000001                  MOV     #1,-(SP)
004202  010600                          MOV     SP,R0                           ; SP,*
004204  104414                          TRAP    14
004206  013700  000000G                 MOV     RP.ADDR,R0                      ;                               6667
004212  116016  000016                  MOVB    16(R0),(SP)
004216  042716  177740                  BIC     #177740,(SP)
004222  012746  000000G                 MOV     #EX.OP,-(SP)
004226  012746  000002                  MOV     #2,-(SP)
004232  010600                          MOV     SP,R0                           ; SP,*
004234  104414                          TRAP    14
004236  012716  000000G                 MOV     #XX16,(SP)                      ;                               6668
004242  012746  000001                  MOV     #1,-(SP)
004246  010600                          MOV     SP,R0                           ; SP,*
004250  104414                          TRAP    14
004252  013700  000000G                 MOV     RP.ADDR,R0                      ;                               6669
004256  016001  000016                  MOV     16(R0),R1
004262  006201                          ASR     R1
004264  006201                          ASR     R1
004266  006201                          ASR     R1
004270  006201                          ASR     R1
004272  006201                          ASR     R1
004274  042701  174000                  BIC     #174000,R1
004300  010116                          MOV     R1,(SP)
004302  012746  000000G                 MOV     #EX.OP,-(SP)
004306  012746  000002                  MOV     #2,-(SP)
004312  010600                          MOV     SP,R0                           ; SP,*
004314  104414                          TRAP    14
004316  013700  000000G                 MOV     RP.ADDR,R0                      ;                               6678
004322  116000  000014                  MOVB    14(R0),R0
004326  042700  177400                  BIC     #177400,R0
004332  020027  000220                  CMP     R0,#220
004336  001406                          BEQ     79$
004340  020027  000241                  CMP     R0,#241                         ;                               6679
```

```
004344  001403                          BEQ     79$
004346  020027  000242                  CMP     R0,#242                 ;                               6680
004352  001072                          BNE     82$
004354  013700  000000G         79$:    MOV     RP.ADDR,R0              ;                               6683
004360  016016  000044                  MOV     44(R0),(SP)
004364  012746  000000G                 MOV     #XX24,-(SP)
004370  012746  000002                  MOV     #2,-(SP)
004374  010600                          MOV     SP,R0                   ; SP,*
004376  104415                          TRAP    15
004400  013700  000000G                 MOV     RP.ADDR,R0              ;                               6684
004404  016016  000020                  MOV     20(R0),(SP)
004410  012746  000000G                 MOV     #XX25,-(SP)
004414  012746  000002                  MOV     #2,-(SP)
004420  010600                          MOV     SP,R0                   ; SP,*
004422  104415                          TRAP    15
004424  013700  000000G                 MOV     RP.ADDR,R0              ;                               6685
004430  016016  000024                  MOV     24(R0),(SP)
004434  016046  000026                  MOV     26(R0),-(SP)
004440  012746  000000G                 MOV     #XX26,-(SP)
004444  012746  000003                  MOV     #3,-(SP)
004450  010600                          MOV     SP,R0                   ; SP,*
004452  104415                          TRAP    15
004454  013700  000000G                 MOV     RP.ADDR,R0              ;                               6686
004460  005760  000014                  TST     14(R0)
004464  100011                          BPL     80$
004466  016016  000040                  MOV     40(R0),(SP)             ;                               6688
004472  012746  000000G                 MOV     #XX21,-(SP)
004476  012746  000002                  MOV     #2,-(SP)
004502  010600                          MOV     SP,R0                   ; SP,*
004504  104414                          TRAP    14
004506  000412                          BR      81$                     ;                               6686
004510  016016  000050         80$:    MOV     50(R0),(SP)             ;                               6690
004514  011646                          MOV     (SP),-(SP)
004516  012746  000000G                 MOV     #XX22,-(SP)
004522  012746  000003                  MOV     #3,-(SP)
004526  010600                          MOV     SP,R0                   ; SP,*
004530  104415                          TRAP    15
004532  005726                          TST     (SP)+
004534  062706  000022         81$:    ADD     #22,SP                  ;                               6682
004540  013700  000000G         82$:    MOV     RP.ADDR,R0              ;                               6692
004544  126027  000014  000204         CMPB    14(R0),#204
004552  001404                          BEQ     83$
004554  126027  000014  000205         CMPB    14(R0),#205             ;                               6693
004562  001053                          BNE     84$
004564  013700  000000G         83$:    MOV     RP.ADDR,R0              ;                               6696
004570  016016  000020                  MOV     20(R0),(SP)
004574  012746  000000G                 MOV     #XX25,-(SP)
004600  012746  000002                  MOV     #2,-(SP)
004604  010600                          MOV     SP,R0                   ; SP,*
004606  104415                          TRAP    15
004610  013700  000000G                 MOV     RP.ADDR,R0              ;                               6697
004614  016016  000024                  MOV     24(R0),(SP)
004620  016046  000026                  MOV     26(R0),-(SP)
```

```
004624  012746  000000G                         MOV     #XX26,-(SP)
004630  012746  000003                         MOV     #3,-(SP)
004634  010600                                 MOV     SP,R0                   ; SP,*
004636  104415                                 TRAP    15
004640  013700  000000G                         MOV     CUOFF,R0                ;                                       6698
004644  006300                                 ASL     R0
004646  063700  000000G                         ADD     CST.ADDR,R0
004652  005016                                 CLR     (SP)
004654  116016  000006                         MOVB    6(R0),(SP)
004660  005046                                 CLR     -(SP)
004662  116016  000006                         MOVB    6(R0),(SP)
004666  012746  000000G                         MOV     #XX23,-(SP)
004672  012746  000003                         MOV     #3,-(SP)
004676  010600                                 MOV     SP,R0                   ; SP,*
004700  104415                                 TRAP    15
004702  004737  000000V                         JSR     PC,EMS.DUP              ;                                       6699
004706  062706  000020                         ADD     #20,SP                  ;                                       6695
004712  062706  000020                 84$:    ADD     #20,SP                  ;                                       6650
004716  062706  000016                 85$:    ADD     #16,SP                  ;                                       6474
004722  000207                                 RTS     PC                      ;                                       6463

; Routine Size:  1258 words,    Routine Base:  $CODE$ + 10220
; Maximum stack depth per invocation:  32 words
```

```
;         6711  !^
;         6712  GLOBAL ROUTINE EMS_DBN : NOVALUE =
;         6713  !+
;         6714  !       THIS ROUTINE PRINTS THE PRESENT DBN
;         6715  !
;         6716  !       IMPLICIT INPUTS:
;         6717  !               CST_ADDR - ADDRESS OF CONTROLLER STATUS TABLE
;         6718  !-
;         6719  BEGIN
;         6720  PRINTB (XX13, .CDISK);                                  ! "DISK XXX"
;         6721  PRINTB (XX23, .CST_ADDR [.CUOFF + 3, D_DBN], .CST_ADDR [.CUOFF + 3, D_DBN]);          ! "DBN: XXXXXX."
;         6722  PRINTB (XX32, .S_DUPPKT - 2);                          ! PRINTS THE BYTE COUNT
;         6723  PRINTB (XX33, .S_PATTERN);                             ! PRINTS THE PATTERN WRITTEN
;         6724  PRINTB (XX34, .(DUPPKT + .S_DUPPKT), .(DUPPKT + .S_DUPPKT));    ! PRINTS THE WORD READ
;         6725  EMS_BLK (DUPPKT +2, 256);                             ! PRINTS THE WHOLE BLOCK READ IN OCTAL
;         6726  END;                                                  ! ROUTINE EMS-DBN

                                .SBTTL  EMS.DBN ERROR MESSAGE SUBROUTINES
000000  013746  000000G   EMS.DBN::
                          MOV     CDISK,-(SP)                   ;                       6720
000004  012746  000000G           MOV     #XX13,-(SP)
000010  012746  000002            MOV     #2,-(SP)
000014  010600                    MOV     SP,R0                 ; SP,*
000016  104414                    TRAP    14
000020  013700  000000G           MOV     CUOFF,R0              ;                       6721
000024  006300                    ASL     R0
000026  063700  000000G           ADD     CST.ADDR,R0
000032  005016                    CLR     (SP)
000034  116016  000006            MOVB    6(R0),(SP)
000040  005046                    CLR     -(SP)
000042  116016  000006            MOVB    6(R0),(SP)
000046  012746  000000G           MOV     #XX23,-(SP)
000052  012746  000003            MOV     #3,-(SP)
000056  010600                    MOV     SP,R0                 ; SP,*
000060  104414                    TRAP    14
000062  013716  000000G           MOV     S.DUPPKT,(SP)         ;                       6722
000066  162716  000002            SUB     #2,(SP)
000072  012746  000000G           MOV     #XX32,-(SP)
000076  012746  000002            MOV     #2,-(SP)
000102  010600                    MOV     SP,R0                 ; SP,*
000104  104414                    TRAP    14
000106  013716  000000G           MOV     S.PATTERN,(SP)        ;                       6723
000112  012746  000000G           MOV     #XX33,-(SP)
000116  012746  000002            MOV     #2,-(SP)
000122  010600                    MOV     SP,R0                 ; SP,*
000124  104414                    TRAP    14
000126  013700  000000G           MOV     S.DUPPKT,R0           ;                       6724
000132  016016  000000G           MOV     DUPPKT(R0),(SP)
000136  011646                    MOV     (SP),-(SP)
000140  012746  000000G           MOV     #XX34,-(SP)
000144  012746  000003            MOV     #3,-(SP)
000150  010600                    MOV     SP,R0                 ; SP,*
```

```
000152  104414                          TRAP    14
000154  012716  000002G                 MOV     #DUPPKT+2,(SP)          ;                       6725
000160  012746  000400                  MOV     #400,-(SP)
000164  004737  000000V                 JSR     PC,EMS.BLK
000170  062706  000034                  ADD     #34,SP                  ;                       6719
000174  000207                          RTS     PC                      ;                       6712

; Routine Size:  63 words,      Routine Base:  $CODE$ + 15144
; Maximum stack depth per invocation:  15 words
```

```
;          6727  ROUTINE EMS_DUP : NOVALUE =
;          6728  !+
;          6729  !       THIS ROUTINE PRINTS OUT THE LISTING OF THE DUP PACKETS ONLY ASSOCIATED WITH DUP RECEIVED DATA COMMAND
;          6730  !       AND TE SEND DATA COMMAND.
;          6731  !-
;          6732  begin
;          6733
;          6734  OWN
;          6735          MESSAGETYPE : VECTOR [6] INITIAL (T_QUE, T_DEF, T_INF, T_TER, T_FAT, T_SPL),
;          6736          MSGNUMBERS  : VECTOR [9] INITIAL (M_TER, M_BIN, M_COD, M_DAT, M_UR, M_URP, M_UP, M_UL, M_ASC),
;          6737          RCD_ERRORS  : VECTOR [5] INITIAL (e5h 30, E_UNT, E_BLK, E_DEV, E_ZER);
;          6738                  ! TABLE OF BASIC, HARD ERROR MESSAGE ADDRESSES, INDEXED BY STATUS CODE
;          6739  !PRINTB (XX13, .CDISK);                                  ! 'DISK XXX'
;          6740  PRINTB (XX29);
;          6741  if .duppkt [duptype] lss 6
;          6742  then
;          6743      PRINTB (.MESSAGETYPE [.DUPPKT [DUPTYPE] - 1])
;          6744  else
;          6745      printb (ex_op, .duppkt [duptype]);
;          6746
;          6747  PRINTB (XX30);
;          6748  if .duppkt [dupmesg] lss 9
;          6749  then
;          6750      begin
;          6751      PRINTB (.MSGNUMBERS [.DUPPKT [DUPMESG] - 1]);
;          6752      IF .DUPPKT [DUPMESG] EQL 3
;          6753      THEN
;          6754          BEGIN
;          6755          PRINTB (XX31);
;          6756          PRINTB (.RCD_ERRORS [.DUPPKT [DUPBF1]]);
;          6757          END;
;          6758      end
;          6759  else printb (ex_op, .duppkt [dupmesg]);
;          6760  end;


015342  000000G                         MESSAGETYPE:
                                                .WORD   T.QUE
015344  000000G                                 .WORD   T.DEF
015346  000000G                                 .WORD   T.INF
015350  000000G                                 .WORD   T.TER
015352  000000G                                 .WORD   T.FAT
015354  000000G                                 .WORD   T.SPL
015356  000000G                         MSGNUMBERS:
                                                .WORD   M.TER
015360  000000G                                 .WORD   M.BIN
015362  000000G                                 .WORD   M.COD
015364  000000G                                 .WORD   M.DAT
015366  000000G                                 .WORD   M.UR
015370  000000G                                 .WORD   M.URP
015372  000000G                                 .WORD   M.UP
015374  000000G                                 .WORD   M.UL
015376  000000G                                 .WORD   M.ASC
```

```
015400  000000G                         RCD.ERRORS:
                                                .WORD   EBH.30
015402  000000G                                 .WORD   E.UNT
015404  000000G                                 .WORD   E.BLK
015406  000000G                                 .WORD   E.DEV
015410  000000G                                 .WORD   E.ZER


                                                .SBTTL  EMS.DUP ERROR MESSAGE SUBROUTINES
000000  012746  000000G             EMS.DUP:MOV     #XX29,-(SP)                             ;                               6740
000004  012746  000001                         MOV     #1,-(SP)
000010  010600                                 MOV     SP,R0                                   ; SP,*
000012  104414                                 TRAP    14
000014  013700  000000G                        MOV     DUPPKT,R0                               ;                               6741
000020  006200                                 ASR     R0
000022  006200                                 ASR     R0
000024  006200                                 ASR     R0
000026  006200                                 ASR     R0
000030  000300                                 SWAB    R0
000032  042700  177760                        BIC     #177760,R0
000036  020027  000006                        CMP     R0,#6
000042  002010                                 BGE     1$
000044  006300                                 ASL     R0                                      ;                               6743
000046  016016  015340'                       MOV     MESSAGETYPE-2(R0),(SP)
000052  012746  000001                         MOV     #1,-(SP)
000056  010600                                 MOV     SP,R0                                   ; SP,*
000060  104414                                 TRAP    14
000062  000410                                 BR      2$                                      ;                               6741
000064  010016                      1$:        MOV     R0,(SP)                                 ;                               6745
000066  012746  000000G                        MOV     #EX.OP,-(SP)
000072  012746  000002                         MOV     #2,-(SP)
000076  010600                                 MOV     SP,R0                                   ; SP,*
000100  104414                                 TRAP    14
000102  005726                                 TST     (SP)+
000104  012716  000000G                 2$:     MOV     #XX30,(SP)                              ;                               6747
000110  012746  000001                         MOV     #1,-(SP)
000114  010600                                 MOV     SP,R0                                   ; SP,*
000116  104414                                 TRAP    14
000120  013700  000000G                        MOV     DUPPKT,R0                               ;                               6748
000124  042700  170000                        BIC     #170000,R0
000130  020027  000011                        CMP     R0,#11
000134  002037                                 BGE     3$
000136  006300                                 ASL     R0                                      ;                               6751
000140  016016  015354'                       MOV     MSGNUMBERS-2(R0),(SP)
000144  012746  000001                         MOV     #1,-(SP)
000150  010600                                 MOV     SP,R0                                   ; SP,*
000152  104414                                 TRAP    14
000154  013700  000000G                        MOV     DUPPKT,R0                               ;                               6752
000160  042700  170000                        BIC     #170000,R0
000164  020027  000003                        CMP     R0,#3
000170  001031                                 BNE     4$
000172  012716  000000G                        MOV     #XX31,(SP)                              ;                               6755
000176  012746  000001                         MOV     #1,-(SP)
```

```
000202  010600                          MOV     SP,R0                           ; SP,*
000204  104414                          TRAP    14
000206  013700  000002G                 MOV     DUPPKT+2,R0                     ;                                       6756
000212  006300                          ASL     R0
000214  016016  015400'                 MOV     RCD.ERRORS(R0),(SP)
000220  012746  000001                  MOV     #1,-(SP)
000224  010600                          MOV     SP,R0                           ; SP,*
000226  104414                          TRAP    14
000230  022626                          CMP     (SP)+,(SP)+                     ;                                       6754
000232  000410                          BR      4$                              ;                                       6748
000234  010016                  3$:     MOV     R0,(SP)                         ;                                       6759
000236  012746  000000G                 MOV     #EX.OP,-(SP)
000242  012746  000002                  MOV     #2,-(SP)
000246  010600                          MOV     SP,R0                           ; SP,*
000250  104414                          TRAP    14
000252  005726                          TST     (SP)+
000254  062706  000012          4$:     ADD     #12,SP                          ;                                       6732
000260  000207                          RTS     PC                              ;                                       6727

; Routine Size:  89 words,      Routine Base:  $CODE$ + 15412
; Maximum stack depth per invocation:  9 words
```

```
;        6761  global routine EMS_BLK (ADDR, LENGTH) : novalue =
;        6762
;        6763  !+
;        6764  !       THIS ROUTINE WILL PRINTX A BLOCK OF MEMORY WHICH IS "LENGTH" WORDS
;        6765  !       LONG STARTING AT ADDRESS "ADDR". PRINTING IS DONE IN OCTAL, 8 WORDS
;        6766  !       TO A LINE.
;        6767  !-
;        6768
;        6769      begin
;        6770
;        6771      literal
;        6772          MASK = %o'7';
;        6773  PRINTX (CRLF);
;        6774      incr COUNT from 1 to .LENGTH do             ! FOR EACH WORD TO PRINT
;        6775          begin
;        6776
;        6777          if ((.COUNT - 1) and MASK) eql 0        ! IF AT START OF A NEW LINE
;        6778          then
;        6779              PRINTX (SPACE4);                    ! PRINTX 4 SPACES
;        6780
;        6781          PRINTX (EX_WRD, ..ADDR);                ! PRINTX A WORD
;        6782          ADDR = .ADDR + 2;                       ! ADVANCE TO NEXT ADDRESS
;        6783
;        6784          if (((.COUNT and MASK) eql 0) or        ! IF AT THE END OF A LINE OR
;        6785              (.COUNT eql .LENGTH))               ! WHEN DONE
;        6786          then
;        6787              PRINTX (CRLF);                      ! PRINTX <CR><LF>
;        6788
;        6789          end;
;        6790
;        6791      end;


                                .SBTTL  EMS.BLK ERROR MESSAGE SUBROUTINES
000000  010146          EMS.BLK::
                                MOV     R1,-(SP)                        ;                       6761
000002  012746  000000G         MOV     #CRLF,-(SP)                     ;                       6773
000006  012746  000001          MOV     #1,-(SP)
000012  010600                  MOV     SP,R0                           ; SP,*
000014  104415                  TRAP    15
000016  005001                  CLR     R1                              ; COUNT                 6774
000020  000445                  BR      5$
000022  010100          1$:     MOV     R1,R0                           ; COUNT,*               6777
000024  005300                  DEC     R0
000026  032700  000007          BIT     #7,R0
000032  001007                  BNE     2$
000034  012716  000000G         MOV     #SPACE4,(SP)                    ;                       6779
000040  012746  000001          MOV     #1,-(SP)
000044  010600                  MOV     SP,R0                           ; SP,*
000046  104415                  TRAP    15
000050  005726                  TST     (SP)+
000052  017616  000012  2$:     MOV     @12(SP),(SP)                    ; ADDR,*                6781
000056  012746  000000G         MOV     #EX.WRD,-(SP)
```

```
000062  012746  000002                        MOV     #2,-(SP)
000066  010600                                MOV     SP,R0           ; SP,*
000070  104415                                TRAP    15
000072  062766  000002  000016                ADD     #2,16(SP)       ; *,ADDR                6782
000100  032701  000007                        BIT     #7,R1           ; *,COUNT               6784
000104  001403                                BEQ     3$
000106  020166  000014                        CMP     R1,14(SP)       ; COUNT,LENGTH          6785
000112  001007                                BNE     4$
000114  012716  000000G               3$:     MOV     #CRLF,(SP)      ;                       6787
000120  012746  000001                        MOV     #1,-(SP)
000124  010600                                MOV     SP,R0           ; SP,*
000126  104415                                TRAP    15
000130  005726                                TST     (SP)+
000132  022626                        4$:     CMP     (SP)+,(SP)+     ;                       6775
000134  005201                        5$:     INC     R1              ; COUNT                 6774
000136  020166  000010                        CMP     R1,10(SP)       ; COUNT,LENGTH
000142  003727                                BLE     1$
000144  022626                                CMP     (SP)+,(SP)+     ;                       6769
000146  012601                                MOV     (SP)+,R1        ;                       6761
000150  000207                                RTS     PC

; Routine Size:  53 words,      Routine Base:  $CODE$ + 15674
; Maximum stack depth per invocation:  8 words
```

```
;       6792  !ROUTINE EMS_MAL : NOVALUE =
;       6793  !+
;       6794  !       THIS ROUTINE PRINTS ALL TABLES IN AN EFFORT TO FIND THE MISSING MESSAGE OR THE PROBLEM
;       6795  !-
;       6796  !begin
;       6797
;       6798  !PRINTB (EB_DCT,dct);                                  ! PRINT " DCT TABLE CONTENTS"
;       6799  !EMS_BLK (DCT, DCT_LEN);                                       ! PRINT BLOCK OF WORDS
;       6800
;       6801  !PRINTB (EB_COMM, .DCT [0, RR_BEG]);                   ! PRINT "COMMAND RING" AND STARTING ADDR
;       6802  !EMS_BLK (.DCT [0, RR_BEG] - 4, COMM_LEN);                       ! PRINT BLOCK OF COMMAND RING
;       6803
;       6804  !PRINTB (EBNEX1, .DCT [0, RR_POLL]);                            !PRINT ADDR OF COMMAND OF NEXT RR POLL
;       6805
;       6806  !PRINTB (EB_NEX2, ..DCT [0, RR_POLL] - 8);                               !PRINT ADDR OF PACKET TO BE POLLED
;       6807
;       6808  !PRINTB (EBNEX3, .DCT [0, CR_POLL] - 8);                               !PRINT ADDR OF PACKET TO BE POLLED
;       6809  !PRINTB (EB_PKT);                                  ! PRINTS "PACKETS IN MEMORY"
;       6810  !incr COUNT from 0 to PKT_CNT - 1 do                ! FOR EACH MSCP ENVELOPE
;       6811  !       begin
;       6812  !       EMS_BLK ((MSCP_PKT [.COUNT, 0,0,16,0]), PKT_LEN);       !PRINTS CONTENTS OF PACKETS
;       6813  !       end;
;       6814  !end;
;       6815
```

```
;          6816  !ROUTINE EMS_RAL : NOVALUE =
;          6817  !+
;          6818  !       THIS ROUTINE PRINTS ALL return packets IN AN EFFORT TO FIND THE MISSING packet OR THE PROBLEM
;          6819  !-
;          6820  !begin
;          6821  !PRINTB (EB_RAL);
;          6822  !incr COUNT from 0 to RP_CNT - 1 do
;          6823  !       begin
;          6824  !       EMS_BLK ((RETPKT + .COUNT * RP_LEN) , RP_LEN);                  ! PRINT BLOCK OF WORDS
;          6825  !       end;
;          6826
;          6827  !end;
;          6828
```

```
;          6829  routine EMS_LBN : novalue =
;          6830
;          6831  !+
;          6832  !       THIS ROUTINE PRINTS (EXTENDED) ONE OF TWO BLOCK NUMBERS APPEARING IN
;          6833  !       THE CURRENT RETURN PACKET. NORMALLY, THE LBN FIELD IS PRINTED; THIS
;          6834  !       FIELD WAS COPIED INTO THE RETURN PACKET FROM THE ASSOCIATED COMMAND
;          6835  !       PACKET. HOWEVER, IF THE "FLAGS" FIELD OF THE CURRENT RETURN PACKET
;          6836  !       INDICATES "BAD BLOCK REPORTED", THEN THE "FIRST BAD BLOCK" FIELD IS
;          6837  !       PRINTED.
;          6838  !
;          6839  !       IMPLICIT INPUTS:
;          6840  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;          6841  !-
;          6842
;          6843      if BIT_TST (RP_ADDR [FLAGS], EF_BBR)                    ! IF BAD BLOCK REPORTED
;          6844      then
;          6845          PRINTX (XX21, .RP_ADDR [BBLK_LO], .RP_ADDR [BBLK_LO])   ! "BAD BLOCK REPORTED: XXXXXX."
;          6846      else
;          6847          PRINTX (XX22, .RP_ADDR [LBN_LO], .RP_ADDR [LBN_LO]);    ! "LBN: XXXXXX."


                                .SBTTL  EMS.LBN ERROR MESSAGE SUBROUTINES
000000  013700  000000G   EMS.LBN:MOV     RP.ADDR,R0                      ;                       6843
000004  005760  000014              TST     14(R0)
000010  100012                      BPL     1$
000012  016046  000040              MOV     40(R0),-(SP)                    ;                       6845
000016  011646                      MOV     (SP),-(SP)
000020  012746  000000G             MOV     #XX21,-(SP)
000024  012746  000003              MOV     #3,-(SP)
000030  010600                      MOV     SP,R0                           ; SP,*
000032  104415                      TRAP    15
000034  000411                      BR      2$                              ;                       6843
000036  016046  000050      1$:     MOV     50(R0),-(SP)                    ;                       6847
000042  011646                      MOV     (SP),-(SP)
000044  012746  000000G             MOV     #XX22,-(SP)
000050  012746  000003              MOV     #3,-(SP)
000054  010600                      MOV     SP,R0                           ; SP,*
000056  104415                      TRAP    15
000060  062706  000010      2$:     ADD     #10,SP                          ;                       6843
000064  000207                      RTS     PC                              ;                       6829

; Routine Size:  27 words,      Routine Base:  $CODE$ + 16046
; Maximum stack depth per invocation:  6 words
```

```
;          6848  global routine EMS_EL (index) : novalue =
;          6849
;          6850  !+
;          6851  !       THIS ROUTINE IS CALLED FROM 'SEQUEN' AND 'DATAGM' AND PRINTS THE CONTENTS OF THE
;          6852  !       ERROR-LOG PACKET
;          6853  !-
;          6854
;          6855      begin
;          6856
;          6857      local
;          6858          ELOG_ADDR : ref block [EP_LEN, word] field (EP_FIELDS),
;          6859          REASON : word,
;          6860          DISK_NUM : byte,
;          6861          ELOG_CODE : byte,
;          6862          ELOG_SUB : word;
;          6863
;          6864      ELOG_ADDR = ELOG_PKT + (.index * EP_LEN * 2);              ! ERROR LOG PACKET'S ADDRESS
;          6865      REASON = .ELOG_ADDR [EL_FORMAT];                           ! FORMAT
;          6866      DISK_NUM = .ELOG_ADDR [EL_DK_NUM];                         ! DISK NUMBER
;          6867      ELOG_CODE = .ELOG_ADDR [EL_CODE];                          ! CODE
;          6868      ELOG_SUB = .ELOG_ADDR [EL_SUBCODE];                        ! SUBCODE
;          6869      PRINTB (ELG_00);                                           ! ERROR-LOG MESSAGE RECEIVED
;          6870
;          6871      if (.REASON eql FORMAT_CNTR) or
;          6872         (.REASON eql FORMAT_HOST)
;          6873      then
;          6874          PRINTB (.ELG_FMT [.REASON])                            ! PRINT BASIC REASON
;          6875      else
;          6876          PRINTB (.ELG_FMT [.REASON], .DISK_NUM);                ! PRINT BASIC REASON WITH DISK NUMBER
;          6877
;          6878      if (.ELOG_CODE gtru 0) and
;          6879         (.ELOG_CODE lequ 11)
;          6880      then
;          6881          begin
;          6882          PRINTX (ASTERISK);
;          6883          PRINTX (.ERR_COD [.ELOG_CODE - 1]);                    ! CODE
;          6884          end
;          6885      else
;          6886
;          6887          if .ELOG_CODE eql ST_DIA                               ! MESSAGE FROM INTERNAL DIAGNOSTICS
;          6888          then
;          6889              begin
;          6890              PRINTX (ASTERISK);
;          6891              PRINTX (.ERR_COD [12]);
;          6892              end;
;          6893
;          6894      if (.ELOG_CODE eql ST_MFE) and
;          6895         (.ELOG_SUB lequ 10)
;          6896      then
;          6897          begin
;          6898          PRINTX (CRLF);
;          6899          PRINTX (ASTERISK);
;          6900          PRINTX (.TBL_MFE [.ELOG_SUB]);                         ! MEDIA FORMAT ERROR
```

```
;         6901          end;
;         6902
;         6903      if (.ELOG_CODE eql ST_DAT) and
;         6904         (.ELOG_SUB lequ 15)
;         6905      then
;         6906          begin
;         6907          PRINTX (CRLF);
;         6908          PRINTX (ASTERISK);
;         6909          PRINTX (.TBL_DAT [.ELOG_SUB]);                    ! DATA ERROR
;         6910          end;
;         6911
;         6912      if (.ELOG_CODE eql ST_HST) and
;         6913         (.ELOG_SUB lequ 4)
;         6914      then
;         6915          begin
;         6916          PRINTX (CRLF);
;         6917          PRINTX (ASTERISK);
;         6918          PRINTX (.TBL_HST [.ELOG_SUB]);                    ! HOST ACCESS ERROR
;         6919          end;
;         6920
;         6921      if (.ELOG_CODE eql ST_CNT) and
;         6922         (.ELOG_SUB lequ 3)
;         6923      then
;         6924          begin
;         6925          PRINTX (CRLF);
;         6926          PRINTX (ASTERISK);
;         6927          PRINTX (.TBL_CNT [.ELOG_SUB]);                    ! CONTROLLER ERROR
;         6928          end;
;         6929
;         6930      if (.ELOG_CODE eql ST_DRV) and
;         6931         (.ELOG_SUB lequ 8)
;         6932      then
;         6933          begin
;         6934          PRINTX (CRLF);
;         6935          PRINTX (ASTERISK);
;         6936          PRINTX (.TBL_DRV [.ELOG_SUB]);                    ! DRIVE ERROR
;         6937          end;
;         6938
;         6939      if .REASON eql FORMAT_XFER                            ! IF DISK XFER INVOLVED
;         6940      then
;         6941
;         6942          if .ELOG_ADDR [EL_BLOCK_TYPE] eql TYPE_LBN        ! PRINT LBN OR RBN
;         6943          then
;         6944              PRINTX (XX22, .ELOG_ADDR [EL_BLOCK], .ELOG_ADDR [EL_BLOCK])
;         6945          else
;         6946              PRINTX (EX_RBN, .ELOG_ADDR [EL_BLOCK], .ELOG_ADDR [EL_BLOCK]);
;         6947
;         6948      PRINTX (CRLF);
;         6949      EMS_BLK ((.ELOG_ADDR + 2), ((.ELOG_ADDR [EL_MSGLEN] + 1) / 2) + 2); ! PRINTX CONTENTS OF PACKET
;         6950      ELOG_ADDR [EL_CONTENTS] = EMPTY;                      ! DECLARE SAVE AREA FREE
;         6951
;         6952      end;
```

```
                                .SBTTL  EMS.EL ERROR MESSAGE SUBROUTINES
000000  004137  000000G       EMS.EL::JSR     R1,$SAVES                       ;                               6848
000004  005746                        TST     -(SP)
000006  016646  000020                MOV     20(SP),-(SP)                    ; INDEX,*                       6864
000012  012746  000102                MOV     #102,-(SP)
000016  004737  000000G               JSR     PC,BL$MUL
000022  062700  000000G               ADD     #ELOG.PKT,R0
000026  010003                        MOV     R0,R3                           ; *,ELOG.ADDR
000030  005004                        CLR     R4                              ; REASON                        6865
000032  156304  000016                BISB    16(R3),R4                       ; *(ELOG.ADDR),REASON
000036  116366  000012  000004        MOVB    12(R3),4(SP)                    ; *(ELOG.ADDR),DISK.NUM         6866
000044  116300  000020                MOVB    20(R3),R0                       ; *(ELOG.ADDR),*                6867
000050  042700  177740                BIC     #177740,R0
000054  105002                        CLRB    R2                              ; ELOG.CODE
000056  050002                        BIS     R0,R2                           ; *,ELOG.CODE
000060  016301  000020                MOV     20(R3),R1                       ; *(ELOG.ADDR),ELOG.SUB         6868
000064  006201                        ASR     R1                              ; ELOG.SUB
000066  006201                        ASR     R1                              ; ELOG.SUB
000070  006201                        ASR     R1                              ; ELOG.SUB
000072  006201                        ASR     R1                              ; ELOG.SUB
000074  006201                        ASR     R1                              ; ELOG.SUB
000076  042701  174000                BIC     #174000,R1                      ; *,ELOG.SUB
000102  012716  000000G               MOV     #ELG.00,(SP)                    ;                               6869
000106  012746  000001                MOV     #1,-(SP)
000112  010600                        MOV     SP,R0                           ; SP,*
000114  104414                        TRAP    14
000116  010405                        MOV     R4,R5                           ; REASON,*                      6874
000120  006305                        ASL     R5
000122  005704                        TST     R4                              ; REASON                        6871
000124  001403                        BEQ     1$
000126  020427  000001                CMP     R4,#1                           ; REASON,*                      6872
000132  001007                        BNE     2$
000134  016516  000000G       1$:     MOV     ELG.FMT(R5),(SP)                ;                               6874
000140  012746  000001                MOV     #1,-(SP)
000144  010600                        MOV     SP,R0                           ; SP,*
000146  104414                        TRAP    14
000150  000412                        BR      3$                              ;                               6871
000152  005016                2$:     CLR     (SP)                            ;                               6876
000154  116616  000006                MOVB    6(SP),(SP)                      ; DISK.NUM,*
000160  016546  000000G               MOV     ELG.FMT(R5),-(SP)
000164  012746  000002                MOV     #2,-(SP)
000170  010600                        MOV     SP,R0                           ; SP,*
000172  104414                        TRAP    14
000174  005726                        TST     (SP)+
000176  105702                3$:     TSTB    R2                              ; ELOG.CODE                     6878
000200  001423                        BEQ     4$
000202  120227  000013                CMPB    R2,#13                          ; ELOG.CODE,*                   6879
000206  101020                        BHI     4$
000210  012716  000000G               MOV     #ASTERISK,(SP)                  ;                               6882
000214  012746  000001                MOV     #1,-(SP)
000220  010600                        MOV     SP,R0                           ; SP,*
000222  104415                        TRAP    15
```

```
000224  005000                          CLR     R0                              ;                               6883
000226  150200                          BISB    R2,R0                           ; ELOG.CODE,*
000230  006300                          ASL     R0
000232  016016  177776G                 MOV     ERR.COD-2(R0),(SP)
000236  012746  000001                  MOV     #1,-(SP)
000242  010600                          MOV     SP,R0                           ; SP,*
000244  104415                          TRAP    15
000246  000417                          BR      5$                              ;                               6881
000250  120227  000037          4$:     CMPB    R2,#37                          ; ELOG.CODE,*                   6887
000254  001015                          BNE     6$
000256  012716  000000G                 MOV     #ASTERISK,(SP)                  ;                               6890
000262  012746  000001                  MOV     #1,-(SP)
000266  010600                          MOV     SP,R0                           ; SP,*
000270  104415                          TRAP    15
000272  013716  000030G                 MOV     ERR.COD+30,(SP)                 ;                               6891
000276  012746  000001                  MOV     #1,-(SP)
000302  010600                          MOV     SP,R0                           ; SP,*
000304  104415                          TRAP    15
000306  022626                  5$:     CMP     (SP)+,(SP)+                     ;                               6889
000310  120227  000005          6$:     CMPB    R2,#5                           ; ELOG.CODE,*                   6894
000314  001031                          BNE     7$
000316  020127  000012                  CMP     R1,#12                          ; ELOG.SUB,*                    6895
000322  101026                          BHI     7$
000324  012716  000000G                 MOV     #CRLF,(SP)                      ;                               6898
000330  012746  000001                  MOV     #1,-(SP)
000334  010600                          MOV     SP,R0                           ; SP,*
000336  104415                          TRAP    15
000340  012716  000000G                 MOV     #ASTERISK,(SP)                  ;                               6899
000344  012746  000001                  MOV     #1,-(SP)
000350  010600                          MOV     SP,R0                           ; SP,*
000352  104415                          TRAP    15
000354  010100                          MOV     R1,R0                           ; ELOG.SUB,*                    6900
000356  006300                          ASL     R0
000360  016016  000064'                 MOV     TBL.MFE(R0),(SP)
000364  012746  000001                  MOV     #1,-(SP)
000370  010600                          MOV     SP,R0                           ; SP,*
000372  104415                          TRAP    15
000374  062706  000006                  ADD     #6,SP                           ;                               6897
000400  120227  000010          7$:     CMPB    R2,#10                          ; ELOG.CODE,*                   6903
000404  001031                          BNE     8$
000406  020127  000017                  CMP     R1,#17                          ; ELOG.SUB,*                    6904
000412  101026                          BHI     8$
000414  012716  000000G                 MOV     #CRLF,(SP)                      ;                               6907
000420  012746  000001                  MOV     #1,-(SP)
000424  010600                          MOV     SP,R0                           ; SP,*
000426  104415                          TRAP    15
000430  012716  000000G                 MOV     #ASTERISK,(SP)                  ;                               6908
000434  012746  000001                  MOV     #1,-(SP)
000440  010600                          MOV     SP,R0                           ; SP,*
000442  104415                          TRAP    15
000444  010100                          MOV     R1,R0                           ; ELOG.SUB,*                    6909
000446  006300                          ASL     R0
000450  016016  000120'                 MOV     TBL.DAT(R0),(SP)
```

```
000454  012746  000001                    MOV     #1,-(SP)
000460  010600                            MOV     SP,R0                   ; SP,*
000462  104415                            TRAP    15
000464  062706  000006                    ADD     #6,SP                   ;                       6906
000470  120227  000011          8$:       CMPB    R2,#11                  ; ELOG.CODE,*           6912
000474  001031                            BNE     9$
000476  020127  000004                    CMP     R1,#4                   ; ELOG.SUB,*            6913
000502  101026                            BHI     9$
000504  012716  000000G                   MOV     #CRLF,(SP)              ;                       6916
000510  012746  000001                    MOV     #1,-(SP)
000514  010600                            MOV     SP,R0                   ; SP,*
000516  104415                            TRAP    15
000520  012716  000000G                   MOV     #ASTERISK,(SP)          ;                       6917
000524  012746  000001                    MOV     #1,-(SP)
000530  010600                            MOV     SP,R0                   ; SP,*
000532  104415                            TRAP    15
000534  010100                            MOV     R1,R0                   ; ELOG.SUB,*            6918
000536  006300                            ASL     R0
000540  016016  000160'                   MOV     TBL.HST(R0),(SP)
000544  012746  000001                    MOV     #1,-(SP)
000550  010600                            MOV     SP,R0                   ; SP,*
000552  104415                            TRAP    15
000554  062706  000006                    ADD     #6,SP                   ;                       6915
000560  120227  000012          9$:       CMPB    R2,#12                  ; ELOG.CODE,*           6921
000564  001031                            BNE     10$
000566  020127  000003                    CMP     R1,#3                   ; ELOG.SUB,*            6922
000572  101026                            BHI     10$
000574  012716  000000G                   MOV     #CRLF,(SP)              ;                       6925
000600  012746  000001                    MOV     #1,-(SP)
000604  010600                            MOV     SP,R0                   ; SP,*
000606  104415                            TRAP    15
000610  012716  000000G                   MOV     #ASTERISK,(SP)          ;                       6926
000614  012746  000001                    MOV     #1,-(SP)
000620  010600                            MOV     SP,R0                   ; SP,*
000622  104415                            TRAP    15
000624  010100                            MOV     R1,R0                   ; ELOG.SUB,*            6927
000626  006300                            ASL     R0
000630  016016  000172'                   MOV     TBL.CNT(R0),(SP)
000634  012746  000001                    MOV     #1,-(SP)
000640  010600                            MOV     SP,R0                   ; SP,*
000642  104415                            TRAP    15
000644  062706  000006                    ADD     #6,SP                   ;                       6924
000650  120227  000013          10$:      CMPB    R2,#13                  ; ELOG.CODE,*           6930
000654  001030                            BNE     11$
000656  020127  000010                    CMP     R1,#10                  ; ELOG.SUB,*            6931
000662  101025                            BHI     11$
000664  012716  000000G                   MOV     #CRLF,(SP)              ;                       6934
000670  012746  000001                    MOV     #1,-(SP)
000674  010600                            MOV     SP,R0                   ; SP,*
000676  104415                            TRAP    15
000700  012716  000000G                   MOV     #ASTERISK,(SP)          ;                       6935
000704  012746  000001                    MOV     #1,-(SP)
000710  010600                            MOV     SP,R0                   ; SP,*
```

```
000712  104415                          TRAP    15
000714  006301                          ASL     R1                      ;                               6936
000716  016116  000202'                 MOV     TBL.DRV(R1),(SP)
000722  012746  000001                  MOV     #1,-(SP)
000726  010600                          MOV     SP,R0                   ; SP,*
000730  104415                          TRAP    15
000732  062706  000006                  ADD     #6,SP                   ;                               6933
000736  020427  000002          11$:    CMP     R4,#2                   ; REASON,*                      6939
000742  001031                          BNE     14$
000744  032763  170000  000060          BIT     #170000,60(R3)          ; *,*(ELOG.ADDR)                6942
000752  001012                          BNE     12$
000754  016316  000056                  MOV     56(R3),(SP)             ; *(ELOG.ADDR),*                6944
000760  011646                          MOV     (SP),-(SP)
000762  012746  000000G                 MOV     #XX22,-(SP)
000766  012746  000003                  MOV     #3,-(SP)
000772  010600                          MOV     SP,R0                   ; SP,*
000774  104415                          TRAP    15
000776  000411                          BR      13$                     ;                               6942
001000  016316  000056          12$:    MOV     56(R3),(SP)             ; *(ELOG.ADDR),*                6946
001004  011646                          MOV     (SP),-(SP)
001006  012746  000000G                 MOV     #EX.RBN,-(SP)
001012  012746  000003                  MOV     #3,-(SP)
001016  010600                          MOV     SP,R0                   ; SP,*
001020  104415                          TRAP    15
001022  062706  000006          13$:    ADD     #6,SP                   ;                               6942
001026  012716  000000G         14$:    MOV     #CRLF,(SP)              ;                               6948
001032  012746  000001                  MOV     #1,-(SP)
001036  010600                          MOV     SP,R0                   ; SP,*
001040  104415                          TRAP    15
001042  012716  000002                  MOV     #2,(SP)                 ;                               6949
001046  060316                          ADD     R3,(SP)                 ; ELOG.ADDR,*
001050  016346  000002                  MOV     2(R3),-(SP)             ; *(ELOG.ADDR),*
001054  005216                          INC     (SP)
001056  012746  000002                  MOV     #2,-(SP)
001062  004737  000000G                 JSR     PC,BL$DIV
001066  010066  000002                  MOV     R0,2(SP)
001072  062766  000002  000002          ADD     #2,2(SP)
001100  005726                          TST     (SP)+
001102  004737  015674'                 JSR     PC,EMS.BLK
001106  105063  000001                  CLRB    1(R3)                   ; *(ELOG.ADDR)                  6950
001112  062706  000016                  ADD     #16,SP                  ;                               6848
001116  000207                          RTS     PC

; Routine Size:  296 words,    Routine Base:  $CODE$ + 16134
; Maximum stack depth per invocation:  16 words
```

```
;        6953  global routine EMS_CMP (ADDR) : novalue =
;        6954
;        6955  !+
;        6956  ! THIS ROUTINE IS CALLED FROM 'HOST_WRT_CHK' AND PRINTS RELEVENT DATA ON A HOST
;        6957  ! COMPARE ERROR
;        6958  !-
;        6959
;        6960      begin
;        6961
;        6962      local
;        6963          ORIG_ADDR : ref block [RP_LEN, word] field (RP_FIELDS);
;        6964
;        6965      ORIG_ADDR = .ADDR;                                          ! ADDRESS OF THE WRITE RETPKT
;        6966      PRINTB (XX13, .CDISK);                                      ! 'DISK XXX"
;        6967      PRINTB (DASH);                                              !
;        6968      PRINTB (.ERR_COD [12]);                                     ! " - HOST COMPARE ERROR"
;        6969      PRINTX (EX_LBW, .ORIG_ADDR [LBN_LO], .ORIG_ADDR [LBN_LO]);  ! LBN (WRITE)
;        6970      PRINTX (EX_LBR, .RP_ADDR [LBN_LO], .RP_ADDR [LBN_LO]);      ! LBN (READ)
;        6971      PRINTX (EX_CBW, .ORIG_ADDR [CBCNT_LO]);                     ! BYTE COUNT (WRITE)
;        6972      PRINTX (XX25, .ORIG_ADDR [BCNT_LO]);                        ! BYTE COUNT XMITTED (WRITE)
;        6973      PRINTX (EX_CBR, .RP_ADDR [CBCNT_LO]);                       ! BYTE COUNT (READ);
;        6974      PRINTX (XX25, .RP_ADDR [BCNT_LO]);                          ! BYTE COUNT XMITTED (READ)
;        6975      PRINTX (EX_BDW, .ORIG_ADDR [BUFF_1], .ORIG_ADDR [BUFF_0]);  ! BUFFER ADDRESS (WRITE)
;        6976      PRINTX (EX_BDR, .RP_ADDR [BUFF_1], .RP_ADDR [BUFF_0]);      ! BUFFER ADDRESS (READ)
;        6977      end;


                                .SBTTL  EMS.CMP ERROR MESSAGE SUBROUTINES
000000  010146          EMS.CMP::
                                MOV     R1,-(SP)                            ;                       6953
000002  016601  000004          MOV     4(SP),R1                            ; ADDR,ORIG.ADDR        6965
000006  013746  000000G         MOV     CDISK,-(SP)                         ;                       6966
000012  012746  000000G         MOV     #XX13,-(SP)
000016  012746  000002          MOV     #2,-(SP)
000022  010600                  MOV     SP,R0                               ; SP,*
000024  104414                  TRAP    14
000026  012716  000000G         MOV     #DASH,(SP)                          ;                       6967
000032  012746  000001          MOV     #1,-(SP)
000036  010600                  MOV     SP,R0                               ; SP,*
000040  104414                  TRAP    14
000042  013716  000030G         MOV     ERR.COD+30,(SP)                     ;                       6968
000046  012746  000001          MOV     #1,-(SP)
000052  010600                  MOV     SP,R0                               ; SP,*
000054  104414                  TRAP    14
000056  016116  000050          MOV     50(R1),(SP)                         ; *(ORIG.ADDR),*        6969
000062  011646                  MOV     (SP),-(SP)
000064  012746  000000G         MOV     #EX.LBW,-(SP)
000070  012746  000003          MOV     #3,-(SP)
000074  010600                  MOV     SP,R0                               ; SP,*
000076  104415                  TRAP    15
000100  013700  000000G         MOV     RP.ADDR,R0                          ;                       6970
000104  016016  000050          MOV     50(R0),(SP)
000110  011646                  MOV     (SP),-(SP)
```

```
000112  012746  000000G        MOV     #EX.LBR,-(SP)
000116  012746  000003         MOV     #3,-(SP)
000122  010600                 MOV     SP,R0                   ; SP,*
000124  104415                 TRAP    15
000126  016116  000044         MOV     44(R1),(SP)             ; *(ORIG.ADDR),*                6971
000132  012746  000000G        MOV     #EX.CBW,-(SP)
000136  012746  000002         MOV     #2,-(SP)
000142  010600                 MOV     SP,R0                   ; SP,*
000144  104415                 TRAP    15
000146  016116  000020         MOV     20(R1),(SP)             ; *(ORIG.ADDR),*                6972
000152  012746  000000G        MOV     #XX25,-(SP)
000156  012746  000002         MOV     #2,-(SP)
000162  010600                 MOV     SP,R0                   ; SP,*
000164  104415                 TRAP    15
000166  013700  000000G        MOV     RP.ADDR,R0              ;                               6973
000172  016016  000044         MOV     44(R0),(SP)
000176  012746  000000G        MOV     #EX.CBR,-(SP)
000202  012746  000002         MOV     #2,-(SP)
000206  010600                 MOV     SP,R0                   ; SP,*
000210  104415                 TRAP    15
000212  013700  000000G        MOV     RP.ADDR,R0              ;                               6974
000216  016016  000020         MOV     20(R0),(SP)
000222  012746  000000G        MOV     #XX25,-(SP)
000226  012746  000002         MOV     #2,-(SP)
000232  010600                 MOV     SP,R0                   ; SP,*
000234  104415                 TRAP    15
000236  016116  000024         MOV     24(R1),(SP)             ; *(ORIG.ADDR),*                6975
000242  016146  000026         MOV     26(R1),-(SP)            ; *(ORIG.ADDR),*
000246  012746  000000G        MOV     #EX.BDW,-(SP)
000252  012746  000003         MOV     #3,-(SP)
000256  010600                 MOV     SP,R0                   ; SP,*
000260  104415                 TRAP    15
000262  013700  000000G        MOV     RP.ADDR,R0              ;                               6976
000266  016016  000024         MOV     24(R0),(SP)
000272  016046  000026         MOV     26(R0),-(SP)
000276  012746  000000G        MOV     #EX.BDR,-(SP)
000302  012746  000003         MOV     #3,-(SP)
000306  010600                 MOV     SP,R0                   ; SP,*
000310  104415                 TRAP    15
000312  062706  000062         ADD     #62,SP                  ;                               6960
000316  012601                 MOV     (SP)+,R1                ;                               6953
000320  000207                 RTS     PC

; Routine Size:  105 words,     Routine Base:  $CODE$ + 17254
; Maximum stack depth per invocation:  28 words
```

```
;         6978  BGNMSG (EMS_01);


                                        .SBTTL  EMS.01 ERROR MESSAGE SUBROUTINES
000000  004737  000000V          EMS.01::JSR     PC,M$EMS.01                     ;                6978
000004  104423                           TRAP    23
000006  000207                           RTS     PC

; Routine Size:  4 words,      Routine Base:  $CODE$ + 17576
; Maximum stack depth per invocation:  2 words


;         6979  PRINTB (EBS_01, MAX_UNITS);                  ! 'MORE THAN XX UNITS SPECIFIED"
;         6980  ENDMSG;


                                        .SBTTL  M$EMS.01 ERROR MESSAGE SUBROUTINES
000000  012746  000004          M$EMS.01:
                                        MOV     #4,-(SP)                        ;                6979
000004  012746  000000G                 MOV     #EBS.01,-(SP)
000010  012746  000002                  MOV     #2,-(SP)
000014  010600                          MOV     SP,R0                           ; SP,*
000016  104414                          TRAP    14
000020  062706  000006                  ADD     #6,SP                           ;                6978
000024  000207                          RTS     PC

; Routine Size:  11 words,     Routine Base:  $CODE$ + 17606
; Maximum stack depth per invocation:  5 words
```

```
;         6981  BGNMSG (EMS_10);


                                        .SBTTL  EMS.10 ERROR MESSAGE SUBROUTINES
000000  004737  000000V           EMS.10::JSR     PC,M$EMS.10                     ;                       6981
000004  104423                            TRAP    23
000006  000207                            RTS     PC

; Routine Size:  4 words,      Routine Base:  $CODE$ + 17634
; Maximum stack depth per invocation:  2 words


;         6982  PRINTB (EBD_10, .RDRX_ADDR + .OF_RC);          ! 'NO RESPONSE AT ADDRESS XXXXXX'
;         6983  ENDMSG;


                                        .SBTTL  M$EMS.10 ERROR MESSAGE SUBROUTINES
000000  013746  000000G           M$EMS.10:
                                        MOV     RDRX.ADDR,-(SP)                 ;                       6982
000004  063716  000000G                   ADD     OF.RC,(SP)
000010  012746  000000G                   MOV     #EBD.10,-(SP)
000014  012746  000002                    MOV     #2,-(SP)
000020  010600                            MOV     SP,R0                           ; SP,*
000022  104414                            TRAP    14
000024  062706  000006                    ADD     #6,SP                           ;                       6981
000030  000207                            RTS     PC

; Routine Size:  13 words,      Routine Base:  $CODE$ + 17644
; Maximum stack depth per invocation:  5 words
```

```
;         6984  BGNMSG (EMS_12);


                                        .SBTTL  EMS.12 ERROR MESSAGE SUBROUTINES
000000  004737  000000V             EMS.12::JSR     PC,M$EMS.12                     ;                       6984
000004  104423                             TRAP    23
000006  000207                             RTS     PC

; Routine Size:  4 words,      Routine Base:  $CODE$ + 17676
; Maximum stack depth per invocation:  2 words


;         6985  PRINTB (EBD_12, .RDRX_ADDR);              ! "INCORRECT BR LEVEL GIVEN FOR DEVICE XXXXXX"
;         6986  ENDMSG;


                                        .SBTTL  M$EMS.12 ERROR MESSAGE SUBROUTINES
000000  013746  000000G           M$EMS.12:
                                        MOV     RDRX.ADDR,-(SP)                 ;                       6985
000004  012746  000000G                 MOV     #EBD.12,-(SP)
000010  012746  000002                  MOV     #2,-(SP)
000014  010600                          MOV     SP,R0                           ; SP,*
000016  104414                          TRAP    14
000020  062706  000006                  ADD     #6,SP                           ;                       6984
000024  000207                          RTS     PC

; Routine Size:  11 words,      Routine Base:  $CODE$ + 17706
; Maximum stack depth per invocation:  5 words
```

```
;         6987  BGNMSG (EMS_13);


                                        .SBTTL  EMS.13 ERROR MESSAGE SUBROUTINES
000000  004737  000000V              EMS.13::JSR     PC,MSEMS.13                     ;                       6987
000004  104423                              TRAP    23
000006  000207                              RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 17734
; Maximum stack depth per invocation:  2 words


;         6988  PRINTB (EBD_13, .STEP);                         ! "STEP X READ ERROR"
;         6989  EMS_SA ();                                      ! PRINTX SA CONTENTS
;         6990  ENDMSG;


                                        .SBTTL  MSEMS.13 ERROR MESSAGE SUBROUTINES
000000  013746  000000G              MSEMS.13:
                                        MOV     STEP,-(SP)                      ;                       6988
000004  012746  000000G                 MOV     #EBD.13,-(SP)
000010  012746  000002                  MOV     #2,-(SP)
000014  010600                          MOV     SP,R0                           ; SP,*
000016  104414                          TRAP    14
000020  004737  007164'                 JSR     PC,EMS.SA                       ;                       6989
000024  062706  000006                  ADD     #6,SP                           ;                       6987
000030  000207                          RTS     PC

; Routine Size:  13 words,      Routine Base:  $CODE$ + 17744
; Maximum stack depth per invocation:  5 words
```

```
;         6991  BGNMSG (EMS_14);


                                        .SBTTL  EMS.14 ERROR MESSAGE SUBROUTINES
000000  004737  000000V              EMS.14::JSR     PC,M$EMS.14                     ;                                       6991
000004  104423                              TRAP    23
000006  000207                              RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 17776
; Maximum stack depth per invocation:  2 words


;         6992  PRINTB (EBD_14, .IRDRX_ADDR);                   ! 'BAD SA CODE FROM DEVICE XXXXXX"
;         6993  EMS_SA ();                                      ! PRINTX SA REGISTER CONTENTS
;         6994  ENDMSG;


                                        .SBTTL  M$EMS.14 ERROR MESSAGE SUBROUTINES
000000  013746  000000G              M$EMS.14:
                                        MOV     IRDRX.ADDR,-(SP)                ;                                       6992
000004  012746  000000G                       MOV     #EBD.14,-(SP)
000010  012746  000002                        MOV     #2,-(SP)
000014  010600                                MOV     SP,R0                           ; SP,*
000016  104414                                TRAP    14
000020  004737  007164'                       JSR     PC,EMS.SA                       ;                                       6993
000024  062706  000006                        ADD     #6,SP                           ;                                       6991
000030  000207                              RTS     PC

; Routine Size:  13 words,      Routine Base:  $CODE$ + 20006
; Maximum stack depth per invocation:  5 words
```

```
;         6995  BGNMSG (EMS_18);


                                        .SBTTL  EMS.18 ERROR MESSAGE SUBROUTINES
000000  004737  000000V              EMS.18::JSR     PC,MSEMS.18                     ;                                   6995
000004  104423                              TRAP    23
000006  000207                              RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 20040
; Maximum stack depth per invocation:  2 words


;         6996  PRINTB (EBD_18, .CDISK);                        ! "DISK XXX WENT OFFLINE"
;         6997  EMSCMD ();                                      ! PRINTX RELEVANT RETPKT FIELDS
;         6998  ENDMSG;


                                        .SBTTL  MSEMS.18 ERROR MESSAGE SUBROUTINES
000000  013746  000000G              MSEMS.18:
                                        MOV     CDISK,-(SP)                     ;                                   6996
000004  012746  000000G                 MOV     #EBD.18,-(SP)
000010  012746  000002                  MOV     #2,-(SP)
000014  010600                          MOV     SP,R0                           ; SP,*
000016  104414                          TRAP    14
000020  004737  010220'                 JSR     PC,EMSCMD                       ;                                   6997
000024  062706  000006                  ADD     #6,SP                           ;                                   6995
000030  000207                          RTS     PC

; Routine Size:  13 words,      Routine Base:  $CODE$ + 20050
; Maximum stack depth per invocation:  5 words
```

```
;         6999  BGNMSG (EMS_21);


                                        .SBTTL  EMS.21 ERROR MESSAGE SUBROUTINES
000000  004737  000000V            EMS.21::JSR     PC,M$EMS.21                     ;                       6999
000004  104423                            TRAP    23
000006  000207                            RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 20102
; Maximum stack depth per invocation:  2 words


;         7000  EMSCMD ();                                        ! CONTENTS OF RETURN PACKET
;         7001  !EMS_RAL ();                                      ! all return packets
;         7002  !EMS_MAL ();                                      ! all message packets
;         7003  ENDMSG;


                                        .SBTTL  M$EMS.21 ERROR MESSAGE SUBROUTINES
000000  004737  010220'             M$EMS.21:
                                        JSR     PC,EMSCMD                       ;                       7000
000004  000207                            RTS     PC                              ;                       6999

; Routine Size:  3 words,       Routine Base:  $CODE$ + 20112
; Maximum stack depth per invocation:  1 word
```

```
;          7004  BGNMSG (EMS_22)


                                        .SBTTL  EMS.22 ERROR MESSAGE SUBROUTINES
000000  004737  000000V                EMS.22::JSR     PC,M$EMS.22                     ;                       7004
000004  104423                                TRAP    23
000006  000207                                RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 20120
; Maximum stack depth per invocation:  2 words


;          7005  EMS_DBN ();                                  ! contents of dup buffer
;          7006  ENDMSG;


                                        .SBTTL  M$EMS.22 ERROR MESSAGE SUBROUTINES
000000  004737  015144'                M$EMS.22:
                                        JSR     PC,EMS.DBN                      ;                       7005
000004  000207                                RTS     PC                              ;                       7004

; Routine Size:  3 words,       Routine Base:  $CODE$ + 20130
; Maximum stack depth per invocation:  1 word
```

```
;        7007  BGNMSG (EMS_30);

                                        .SBTTL  EMS.30 ERROR MESSAGE SUBROUTINES
000000  004737  000000V              EMS.30::JSR     PC,M$EMS.30                     ;                       7007
000004  104423                              TRAP    23
000006  000207                              RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 20136
; Maximum stack depth per invocation:  2 words

;        7008  EMSCMD ();                                       ! contents of return packet
;        7009  ENDMSG;

                                        .SBTTL  M$EMS.30 ERROR MESSAGE SUBROUTINES
000000  004737  010220'               M$EMS.30:
                                        JSR     PC,EMSCMD                       ;                       7008
000004  000207                              RTS     PC                              ;                       7007
; Routine Size:  3 words,       Routine Base:  $CODE$ + 20146
; Maximum stack depth per invocation:  1 word
;        7010
;        7011  end
;        7012
;        7013  eludom
;                                    OTS external references
                                            .GLOBL  $SAVE5, $SAVE4, $SAVE3, $SAVE2
                                            .GLOBL  BL$DIV, BL$MUL
;                                   PSECT SUMMARY
;
;
;       Psect Name                          Words     Attributes
;         $OWN$                               74       RW  ,  D  ,  LCL,  REL,  CON
;         $CODE$                            4150       RO  ,  I  ,  LCL,  REL,  CON
;         $PLIT$                              12       RO  ,  D  ,  LCL,  REL,  CON
;                                   LIBRARY STATISTICS
;                                            -------- Symbols --------     Blocks
;       File                                  Total    Loaded   Percent      Read
;
;                                               455       350        76        90
;
; Information:  1
; Warnings:     0
; Errors:       0
;                                   COMMAND QUALIFIERS
; Size:         3970 code + 7027 data words
; Run Time:       03:47.6
; Elapsed Time:   07:35.4
; Memory Used:  825 pages
; Compilation Complete
```

```
0001
0002  module ZRQAM3 (
0003
0004  %title 'RD/RX EXERCISER'
0005                  ident = 'V01.2',
0006                  addressing_mode (absolute),
0007                  environment (noeis)
0008                  ) =
0009
0010  begin
0011
0012  %sbttl 'DECLARATIONS'
0013
0014  library 'ZRQAB0.L16';                    ! RDRX EXERCISER GLOBAL LIBRARY
0015
0016  require 'BLSMAC.REQ';                    ! DIAGNOSTIC SUPERVISOR LIBRARY
1507
1508  EQUALS;
1509
1510  forward routine                          ! ROUTINES APPEAR IN THIS ORDER
1511      INIT_TEST : novalue,                 !    INDENTATION IMPLIES CALLED SUBROUTINE
1512          DRIVER_INIT : novalue,
1513              CTLR_INIT : novalue,
1514                  INI_CTLR_DAT : novalue,
1515                  REG_EXIST,
1516                  VEC_BR_TEST,
1517                  INT_GEN,
1518                  HARD_INIT,
1519                  INI_RRING : novalue,
1520                  SET_CTLR_CHAR,
1521              UNIT_INIT : novalue,
1522                  DR_ERR : novalue,
1523          ACCESS : novalue,
1524      MULTI_DRIVE : novalue,
1525          MD_INIT : novalue,
1526              INIT_IO_BUFF : novalue,
1527          FATAL_ERROR : novalue,
1528          QIO_OK,
1529              QIO_OUT,
1530              QIO_GEN : novalue,
1531                  GET_RANDOM : novalue,
1532                  QIO_UNIT : novalue,
1533                  QIO_FUNC : novalue,
1534                    DUP : novalue,
1535                          DUPWRTDBN : novalue,
1536                          DUPREDDBN : novalue,
1537                              DUPCOMMAND : novalue,
1538                  QIO_LBN : novalue,
1539                  QIO_SIZE : novalue,
1540                  FILL_BUFF : novalue,
1541              PROC_RETPKT : novalue,
1542                  DIO_RETPKT : novalue,
1543                      DUP_COMPARE : novalue,
```

```
;       1544                    IO_RETPKT : novalue,
;       1545                        FSET_UPAR : novalue,
;       1546                        HARD_ERROR : novalue,
;       1547                        UPD_IO_TALLY : novalue,
;       1548                            OVF_CHK : novalue,
;       1549                        HOST_WRT_CHK,
;       1550                        SWEEP : novalue,
;       1551                            RPS_REM,
;       1552                    DR_RETPKT : novalue,
;       1553                AZINTO : L$ISR novalue,
;       1554                    AZINT : novalue,
;       1555            !           FATAL_ERROR : novalue,
;       1556                        POLL_CRING : novalue,
;       1557                        POLL_RRING : novalue,
;       1558                        DUP_RSP : novalue,
;       1559                        MSCP_RSP : novalue,
;       1560                            SEQUEN : novalue,
;       1561                                SOFT_ERROR : novalue,
;       1562                            DATAGM : novalue;
;       1563            !                       SOFT_ERROR : novalue;
;       1564
;       1565        external
;       1566            CST : blockvector [MAX_CTLR, CST_LEN, word] field (CST_FIELDS),
;       1567                                ! RUN-TIME CONTROLLER STATUS TABLES
;       1568            CST_ADDR : ref block [CST_LEN, word] field (CST_FIELDS),
;       1569                                ! CONTROLLER STATUS TABLE ADDRESS OF "CURRENT" CONTROLLER
;       1570            DCT : blockvector [MAX_CTLR, DCT_LEN, word] field (DCT_FIELDS),
;       1571                                ! DRIVER CONTROLLER TABLES
;       1572            DCT_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS),
;       1573                                ! ADDRESS OF "CURRENT" DRIVER CONTROLLER TABLE
;       1574            RDRX_ADDR : ref rdrx field (RC_REG),
;       1575                                ! DEVICE ADDRESS OF "CURRENT" CONTROLLER
;       1576            IRDRX_ADDR : ref rdrx field (RC_REG),
;       1577                                ! DEVICE ADDRESS OF INTERRUPTING CONTROLLER
;       1578            DUPPKT : BLOCK [257, WORD] field (DP_FIELDS),
;       1579                                ! BUFFER CONTAINING DUP INFORMATION FROM RECEIVE AND SEND COMMANDS
;       1580            TALLY : vector [MAX_UNITS * TALLY_LEN, word] field (T_FIELDS),
;       1581                                ! STATISTICS TABLES
;       1582            T_ADDR : ref block [TALLY_LEN, word] field (T_FIELDS),
;       1583                                ! ADDRESS OF STATISTICS TABLE (TALLY) FOR CURRENT UNIT
;       1584            C_ERR_TBL : blockvector [MAX_CTLR, C_ERR_LEN, word] field (C_ERR_FIELDS),
;       1585                                ! STATISTICS TABLE FOR CONTROLLER ERRORS
;       1586            MSCP_PKT : blockvector [PKT_CNT, PKT_LEN, word] field (PKT_FIELDS),
;       1587                                ! MSCP PACKET POOL
;       1588            IPKT_ADDR : ref block [PKT_LEN, word] field (PKT_FIELDS),
;       1589                                ! ADDRESS OF AN MSCP PACKET (INTERRUPT PROCESSING)
;       1590            PKT_USE : vector [PKT_CNT, byte, signed],
;       1591                                ! MSCP PACKET POOL ALLOCATION TABLE
;       1592            RETPKT : blockvector [RP_CNT, RP_LEN, word] field (RP_FIELDS),
;       1593                                ! RETURN PACKET POOL
;       1594            RP_USE : vector [RP_CNT, byte, signed],
;       1595                                ! RETURN PACKET POOL ALLOCATION TABLE
;       1596            RP_INDX : word,     ! CURRENT RETURN PACKET INDEX
```

```
;          1597      RP_ADDR : ref block [RP_LEN, word] field (RP_FIELDS),
;          1598                          ! CURRENT RETURN PACKET ADDRESS
;          1599      ELOG_PKT : blockvector [EP_CNT, EP_LEN, word] field (EP_FIELDS),
;          1600                          ! ERROR-LOG PACKET SAVE AREA
;          1601      BUFF_ADDR : vector [MAX_BUF_CNT],    ! TABLE OF I/O BUFFER DESCRIPTORS
;          1602      BUFF_OWN : vector [MAX_BUF_CNT, byte, signed],      ! I/O BUFFER OWNERSHIP (CONTROLLER NUMBER)
;          1603      IODQ : vector [IODQ_LEN, byte],
;          1604                                          ! I/O DONE QUEUE - CIRCULAR QUEUE OF RETPKT INDECES
;          1605      IODQ_IN : word,                     ! I/O DONE QUEUE IN POINTER
;          1606      IODQ_OUT : word,                    ! I/O DONE QUEUE OUT POINTER
;          1607      ENTRY_REASON : byte,                ! CURRENT OPERATOR COMMAND
;          1608      EOP_FLAG : byte,                    ! END-OF-PASS FLAG
;          1609      DUP_FLAGS : WORD,                   ! DUP FLAGS
;          1610      CCTLR : word,                       ! NUMBER OF "CURRENT" CONTROLLER
;          1611      CDISK : word,                       ! CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;          1612      CUOFF : word,                       ! CURRENT UNIT CST OFFSET
;          1613      CTLR_CNT : word,                    ! TOTAL NUMBER OF CONFIGURED CONTROLLERS
;          1614      DUR : vector [MAX_UNITS, byte],     ! DROP UNIT REASON
;          1615      QIO : vector [MAX_CTLR, byte],      ! NUMBER OF OUTSTANDING QIOS PER CONTROLLER
;          1616      FREE_MEM_ADDR,                      ! START OF FREE MEMORY
;          1617      BYTS_PER_QIO : word,                ! SIZE (BYTES) OF AN I/O BUFFER
;          1618      ST_CODE : word,                     ! CURRENT STATUS CODE
;          1619      SB_CODE : word,                     ! CURRENT SUB-CODE
;          1620      STEP : word,                        ! CURRENT STEP IN HARD_INIT
;          1621      OF_RC : signed word,                ! OFFSET (0 OR 2) TO READ IP OR SA
;          1622      SA_REG : word,                      ! STORAGE FOR SA REGISTER READS AND WRITES
;          1623      CMD_TIME : word,                    ! COMMAND TIMEOUT VALUE (IN SECONDS)
;          1624      NEX : word,                         ! NON-EXISTENT MEMORY TRAP INDICATOR
;          1625      CRN_LOW : word,                             ! COMMAND REF NUMBER OF LAST COMMAND SENT
;          1626      CRN_HIGH : word,                            ! COMMAND REF NUMBER OF LAST COMMAND SENT
;          1627      P_INDEX : signed word,        ! CURRENT message PACKET INDEX
;          1628      S_DUPPKT : WORD,              ! DBN BYTE COUNTER
;          1629      S_PATTERN : WORD,             ! THE PATTERN WRITTEN TO DBN'S
;          1630      CREDIT_BAL : word,                  ! CREDIT BALANCE
;          1631      NEXT_PKT_USE : byte,                ! POINTER TO NEXT ENTRY IN PKT_USE TABLE
;          1632      DBM12,
;          1633      DBM18,
;          1634      DBM19,
;          1635      DBM20,
;          1636      DBM21,
;          1637      !DBM22,
;          1638      DBM23,
;          1639      DBM25,
;          1640      DBM26,
;          1641      DBM29,
;          1642      DBM108,
;          1643      DBM109,
;          1644      DBM110,
;          1645      DBM112,
;          1646      EH_0,
;          1647      EH_1,
;          1648      EH_2,
;          1649      EH_3,
```

```
;          1650      EH_4,
;          1651      EH_5,
;          1652      EH_6,
;          1653      EH_7,
;          1654      EH_8,
;          1655      EH_9,
;          1656      EH_10,
;          1657      EH_12,
;          1658      EH_13,
;          1659      MSG_02,
;          1660      MSG_03,
;          1661      EGS_02,
;          1662      EGD_10,
;          1663      EGD_11,
;          1664      EGD_12,
;          1665      EGD_13,
;          1666      EGD_14,
;          1667      EGD_15,
;          1668      EGD_16,
;          1669      EGD_17,
;          1670      EGD_18,
;          1671      EGD_20,
;          1672      EGD_21,
;          1673      EGD_22,
;          1674      EGD_23,
;          1675      EGH_30,
;          1676      CRLF,
;          1677      SWP_ERROR : word,                          ! HARD ERROR LIMIT FOR DROPPING UNIT
;          1678      SWP_XFER : word,                           ! TRANSFER LIMIT FOR DROPPING UNIT
;          1679      SWP_FLAGS : word,                          ! FLAGS (SEE DOCUMENTATION)
;          1680      dupround : word,                           ! ratio constant for dup exerciser
;          1681      SWP_RAT : word,                            ! RD51 OPERATION RATIO
;          1682      SWP_DPAT : word,                           ! DATA PATTERN NUMBER
;          1683      SWP_UCNT : word,                           ! USER DATA PATTERN COUNT
;          1684      SWP_UDPAT : vector [MAX_UDP_CNT, word],    ! USER DATA PATTERN
;          1685      L$LUN,
;          1686      L$UNIT;
;          1687
;          1688  psect
;          1689      own = $GGG$(read, nowrite, execute, local, concatenate);
;          1690
;          1691  own
;          1692      COMM_AREA : blockvector [MAX_CTLR, COMM_LEN, word] field (COM_FIELDS),
;          1693                              ! COMMUNICATIONS AREA BETWEEN HOST AND AZTEC CONTROLLERS
;          1694      BST : vector [MAX_UNITS, word, signed],
;          1695                              ! BLOCK SEQUENCE TABLE FOR SEQUENTIAL LBN (VS. RANDOM SEEK) MODE
;          1696      DPST : vector [MAX_UNITS, byte],           ! DATA PATTERN SEQUENCE TABLE
;          1697      MAX_LBN : vector [MAX_UNITS, word],        ! LARGEST LBN ALLOWED
;          1698      STORAGE : vector [MAX_UNITS, word],        ! DUMMY STORAGE
;          1699      ICOM_ADDR : ref block [COMM_LEN, word] field (COM_FIELDS),
;          1700                              ! ADDRESS OF INTERRUPTING CONTROLLER'S COMMUNICATION AREA
;          1701      ICST_ADDR : ref block [CST_LEN, word] field (CST_FIELDS),
;          1702                              ! ADDRESS OF INTERRUPTING CONTROLLER'S CST
```

```
1703        IDCT_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS),
1704                    ! ADDRESS OF INTERRUPTING CONTROLLER'S DCT
1705        INT_ADDR : vector [MAX_CTLR] initial (AZINT0 %(, AZINT1, AZINT2, AZINT3)%),
1706                    ! INTERRUPT SERVICE ROUTINE ADDRESS TABLE
1707        RDM_CNT : word initial (RDM_LEN),           ! NUMBER OF RANDOM NUMBERS  \ KEEP
1708        RANDOM : vector [RDM_LEN, word],            ! RANDOM NUMBER TABLE       / TOGETHER
1709        ICTLR : word,                               ! INTERRUPTING CONTROLLING NUMBER
1710        MX1 : signed word,                          ! MSCP PKT INDEX FOR FIRST QIO
1711        MX2 : signed word,                          ! MSCP PKT INDEX FOR SECOND QIO
1712        MAD1 : ref block [PKT_LEN, word] field (PKT_FIELDS),
1713                    ! ADDRESS OF MSCP PACKET FOR FIRST QIO
1714        MAD2 : ref block [PKT_LEN, word] field (PKT_FIELDS),
1715                    ! ADDRESS OF MSCP PACKET FOR SECOND QIO
1716        LAST_PKT : blockvector [MAX_CTLR, LAST_PKT_LEN, word] field (LAST_PKT_FIELDS),
1717                    ! SAVE AREA FOR INFO ABOUT LAST RESPONSE PACKET
1718        PAT02 : vector [2] initial (1,                                  ! PATTERN 2
1719                %o'000000'),
1720        PAT03 : vector [2] initial (1,                                  ! PATTERN 3
1721                %o'177777'),
1722        PAT04 : vector [2] initial (1,                                  ! PATTERN 4
1723                %o'105613'),
1724        PAT05 : vector [2] initial (1,                                  ! PATTERN 5
1725                %o'031463'),
1726        PAT06 : vector [2] initial (1,                                  ! PATTERN 6
1727                %o'030221'),
1728        PAT07 : vector [17] initial (16,                                ! PATTERN 7
1729                %o'000001', %o'000003', %o'000007', %o'000017',
1730                %o'000037', %o'000077', %o'000177', %o'000377',
1731                %o'000777', %o'001777', %o'003777', %o'007777',
1732                %o'017777', %o'037777', %o'077777', %o'177777'),
1733        PAT08 : vector [17] initial (16,                                ! PATTERN 8
1734                %o'177776', %o'177774', %o'177770', %o'177760',
1735                %o'177740', %o'177700', %o'177600', %o'177400',
1736                %o'177000', %o'176000', %o'174000', %o'170000',
1737                %o'160000', %o'140000', %o'100000', %o'000000'),
1738        PAT09 : vector [17] initial (16,                                ! PATTERN 9
1739                rep 3 of (%o'000000'), rep 3 of (%o'177777'),
1740                rep 2 of (%o'000000'), rep 2 of (%o'177777'),
1741                %o'000000', %o'177777', %o'000000', %o'177777',
1742                %o'000000', %o'177777'),
1743        PAT10 : vector [2] initial (1,                                  ! PATTERN 10
1744                %o'133331'),
1745        PAT11 : vector [17] initial (16,                                ! PATTERN 11
1746                rep 3 of (%o'052525'), rep 3 of (%o'125252'),
1747                rep 2 of (%o'052525'), rep 2 of (%o'125252'),
1748                %o'052525', %o'125252', %o'052525', %o'125252',
1749                %o'052525', %o'125252'),
1750        PAT12 : vector [21] initial (20,                                ! PATTERN 12
1751                rep 3 of (%o'026455'), rep 3 of (%o'151322'),
1752                rep 2 of (%o'026455'), rep 2 of (%o'151322'),
1753                rep 2 of (%o'026455'),
1754                %o'151322', %o'026455', %o'151322', %o'026455',
1755                %o'151322', %o'026455', %o'151322', %o'026455'),
```

```
1756      PAT13 : vector [2] initial (1,                                  ! PATTERN 13
1757              %o'066666'),
1758      PAT14 : vector [17] initial (16,                                ! PATTERN 14
1759              %o'000001', %o'000002', %o'000004', %o'000010',
1760              %o'000020', %o'000040', %o'000100', %o'000200',
1761              %o'000400', %o'001000', %o'002000', %o'004000',
1762              %o'010000', %o'020000', %o'040000', %o'100000'),
1763      PAT15 : vector [17] initial (16,                                ! PATTERN 15
1764              %o'177776', %o'177775', %o'177773', %o'177767',
1765              %o'177757', %o'177737', %o'177677', %o'177577',
1766              %o'177377', %o'176777', %o'175777', %o'173777',
1767              %o'167777', %o'157777', %o'137777', %o'077777'),
1768      PAT16 : vector [17] initial (16,                                ! PATTERN 16
1769              rep 3 of (%o'133331'), rep 3 of (%o'155554'),
1770              rep 2 of (%o'133331'), rep 2 of (%o'155554'),
1771              %o'133331', %o'155554', %o'133331', %o'155554',
1772              %o'133331', %o'155554'),
1773      PAT17 : vector [22] initial (21,                                ! PATTERN 17
1774              %o'000000', rep 2 of (%o'106466'),
1775              rep 3 of (%o'071311'), rep 4 of (%o'106466'),
1776              rep 5 of (%o'071311'), rep 6 of (%o'106466')),
1777      PAT18 : vector [22] initial (21,                                ! PATTERN 18
1778              %o'106466', %o'000000', %o'071311',
1779              rep 3 of (%o'106466'), rep 4 of (%o'071311'),
1780              rep 5 of (%o'106466'), rep 6 of (%o'071311')),
1781      PAT19 : vector [22] initial (21,                                ! PATTERN 19
1782              %o'000000', rep 2 of (%o'134631'),
1783              rep 3 of (%o'043146'), rep 4 of (%o'134631'),
1784              rep 5 of (%o'043146'), rep 6 of (%o'134631')),
1785      PAT20 : vector [22] initial (21,                                ! PATTERN 20
1786              %o'134631', %o'000000', %o'043146',
1787              rep 3 of (%o'134631'), rep 4 of (%o'043146'),
1788              rep 5 of (%o'134631'), rep 6 of (%o'043146')),
1789      PAT21 : vector [2] initial (1,                                  ! PATTERN 21
1790              %o'000000'),                                            !    (LBN)
1791      DPA_TBL : vector [DP_CNT] initial                               ! DATA PATTERN ADDRESS TABLE
1792                (RDM_CNT, PAT02, PAT03, PAT04, PAT05,
1793                 PAT06, PAT07, PAT08, PAT09, PAT10, PAT11,
1794                 PAT12, PAT13, PAT14, PAT15, PAT16, PAT17,
1795                 PAT18, PAT19, PAT20, PAT21),
1796      TRK_SGN : vector [MAX_UNITS, word, signed] initial (rep MAX_UNITS of (1)),      ! CURRENT TRACK DIRECTION
1797      BST_CNT : word initial (0),                                     ! CURRENT SEQUENTIAL BLOCK COUNT
1798      BST_DEV : word initial (0),                                     ! CURRENT SEQUENTIAL BLOCK DEVICE
1799      CURRENT_VECTOR : word,                                          ! CURRENT DEVICE'S VECTOR ADDRESS
1800      BRLEVEL : word,                                                 ! CURRENT DEVICE'S BR LEVEL
1801      COMPARE_DATA : byte;                                            ! FLAGGED CLEARED TO BYPASS HOST COMPARES
1802
1803  external routine
1804      NEX_TRAP : L$ISR novalue,
1805      SET_CPAR : novalue,
1806      SET_UPAR : novalue,
1807      OUT_IODQ,
1808      IN_IODQ : novalue,
```

```
;          1809      GET_PKT,
;          1810      PUT_PKT : novalue,
;          1811      GET_RETPKT,
;          1812      PUT_RETPKT : novalue,
;          1813      GET_IO_BUFF : novalue,
;          1814      PUT_IO_BUFF : novalue,
;          1815      PUTA_BUFF : novalue,
;          1816      SEND,
;          1817      WAIT : novalue,
;          1818      DROP_CTLR : novalue,
;          1819      DRV_CTLERR : novalue,
;          1820      EMSCMD : novalue,
;          1821      EMS_22 : novalue,
;          1822      EMS_EL : novalue,
;          1823      EMS_CMP : novalue,
;          1824      EMS_10 : novalue,
;          1825      EMS_12 : novalue,
;          1826      EMS_13 : novalue,
;          1827      EMS_14 : novalue,
;          1828      EMS_18 : novalue,
;          1829      EMS_21 : novalue,
;          1830      EMS_30 : novalue;
```

```
;       1831  %sbttl 'TEST SECTION'
;       1832
;       1833  !+
;       1834  !       THIS SECTION CONTAINS THE TOP-LEVEL TEST CODE FOR THE RDRX EXERCISER.
;       1835  !       THE EXERCISER CONSISTS OF ONE TEST WHICH IS SUBDIVIDED INTO A NUMBER OF
;       1836  !       SUBTESTS. ALL SUBTESTS ARE DECLARED WITHIN THIS BLOCK.
;       1837  !-
;       1838
;       1839  BGNTST;
;       1840
;       1841  EOP_FLAG = TRUE;                                      ! ASSUME NO UNIT AVAILABLE
;       1842  COMPARE_DATA = TRUE;                                  ! ALLOW HOST COMPARES IF ASKED FOR
;       1843  INIT_TEST ();                                         ! INITIALIZE TEST ENVIRONMENT
;       1844
;       1845  incr CTLR from 0 to (MAX_CTLR - 1) do                 ! FOR EVERY CONTROLLER
;       1846
;       1847      if (.CST [.CTLR, STATE] eql ONLINE) and           ! IF CONTROLLER ONLINE
;       1848         (.DCT [.CTLR, STAT] eql ONLINE) and
;       1849         (.CST [.CTLR, U_CNT] gequ 0)
;       1850      then
;       1851
;       1852          incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + 4) by UNIT_SIZE do
;       1853
;       1854              if .CST [.CTLR, .OFFSET, D_STAT] eql ONLINE ! IF AT LEAST ONE UNIT ALIVE
;       1855              then
;       1856                  begin
;       1857                  EOP_FLAG = FALSE;                     ! NOT END OF PASS
;       1858                  exitloop;
;       1859                  end;
;       1860
;       1861  if not .EOP_FLAG
;       1862  then
;       1863      MULTI_DRIVE ();                                   ! RUN MULTI-DRIVE TEST
;       1864
;       1865  ENDTST;


                                .TITLE  ZRQAM3 RD/RX EXERCISER
                                .IDENT  /V01.2/
                                .ENABL  AMA


000000                          .PSECT  $GGG$,  RO
000000                  COMM.AREA:
                                .BLKW   24
000050                  BST:    .BLKW   4
000060                  DPST:   .BLKW   2
000064                  MAX.LBN:.BLKW   4
000074                  STORAGE:.BLKW   4
000104                  ICOM.ADDR:
                                .BLKW   1
000106                  ICST.ADDR:
                                .BLKW   1
```

```
000110                          IDCT.ADDR:
                                        .BLKW   1
000112  000000V                 INT.ADDR:
                                        .WORD   AZINT0
000114  000020                  RDM.CNT:.WORD   20
000116                          RANDOM: .BLKW   20
000156                          ICTLR:  .BLKW   1
000160                          MX1:    .BLKW   1
000162                          MX2:    .BLKW   1
000164                          MAD1:   .BLKW   1
000166                          MAD2:   .BLKW   1
000170                          LAST.PKT:
                                        .BLKW   3
000176  000001                  PAT02:  .WORD   1
000200  000000                          .WORD   0
000202  000001                  PAT03:  .WORD   1
000204  177777                          .WORD   -1
000206  000001                  PAT04:  .WORD   1
000210  105613                          .WORD   -72165
000212  000001                  PAT05:  .WORD   1
000214  031463                          .WORD   31463
000216  000001                  PAT06:  .WORD   1
000220  030221                          .WORD   30221
000222  000020                  PAT07:  .WORD   20
000224  000001                          .WORD   1
000226  000003                          .WORD   3
000230  000007                          .WORD   7
000232  000017                          .WORD   17
000234  000037                          .WORD   37
000236  000077                          .WORD   77
000240  000177                          .WORD   177
000242  000377                          .WORD   377
000244  000777                          .WORD   777
000246  001777                          .WORD   1777
000250  003777                          .WORD   3777
000252  007777                          .WORD   7777
000254  017777                          .WORD   17777
000256  037777                          .WORD   37777
000260  077777                          .WORD   77777
000262  177777                          .WORD   -1
000264  000020                  PAT08:  .WORD   20
000266  177776                          .WORD   -2
000270  177774                          .WORD   -4
000272  177770                          .WORD   -10
000274  177760                          .WORD   -20
000276  177740                          .WORD   -40
000300  177700                          .WORD   -100
000302  177600                          .WORD   -200
000304  177400                          .WORD   -400
000306  177000                          .WORD   -1000
000310  176000                          .WORD   -2000
000312  174000                          .WORD   -4000
000314  170000                          .WORD   -10000
```

```
000316  160000                          .WORD   -20000
000320  140000                          .WORD   -40000
000322  100000                          .WORD   -100000
000324  000000                          .WORD   0
000326  000020                  PAT09:  .WORD   20
000330  000000                          .WORD   0
000332  000000                          .WORD   0
000334  000000                          .WORD   0
000336  177777                          .WORD   -1
000340  177777                          .WORD   -1
000342  177777                          .WORD   -1
000344  000000                          .WORD   0
000346  000000                          .WORD   0
000350  177777                          .WORD   -1
000352  177777                          .WORD   -1
000354  000000                          .WORD   0
000356  177777                          .WORD   -1
000360  000000                          .WORD   0
000362  177777                          .WORD   -1
000364  000000                          .WORD   0
000366  177777                          .WORD   -1
000370  000001                  PAT10:  .WORD   1
000372  133331                          .WORD   -44447
000374  000020                  PAT11:  .WORD   20
000376  052525                          .WORD   52525
000400  052525                          .WORD   52525
000402  052525                          .WORD   52525
000404  125252                          .WORD   -52526
000406  125252                          .WORD   -52526
000410  125252                          .WORD   -52526
000412  052525                          .WORD   52525
000414  052525                          .WORD   52525
000416  125252                          .WORD   -52526
000420  125252                          .WORD   -52526
000422  052525                          .WORD   52525
000424  125252                          .WORD   -52526
000426  052525                          .WORD   52525
000430  125252                          .WORD   -52526
000432  052525                          .WORD   52525
000434  125252                          .WORD   -52526
000436  000024                  PAT12:  .WORD   24
000440  026455                          .WORD   26455
000442  026455                          .WORD   26455
000444  026455                          .WORD   26455
000446  151322                          .WORD   -26456
000450  151322                          .WORD   -26456
000452  151322                          .WORD   -26456
000454  026455                          .WORD   26455
000456  026455                          .WORD   26455
000460  151322                          .WORD   -26456
000462  151322                          .WORD   -26456
000464  026455                          .WORD   26455
000466  026455                          .WORD   26455
```

```
000470  151322                          .WORD   -26456
000472  026455                          .WORD   26455
000474  151322                          .WORD   -26456
000476  026455                          .WORD   26455
000500  151322                          .WORD   -26456
000502  026455                          .WORD   26455
000504  151322                          .WORD   -26456
000506  026455                          .WORD   26455
000510  000001                  PAT13:  .WORD   1
000512  066666                          .WORD   66666
000514  000020                  PAT14:  .WORD   20
000516  000001                          .WORD   1
000520  000002                          .WORD   2
000522  000004                          .WORD   4
000524  000010                          .WORD   10
000526  000020                          .WORD   20
000530  000040                          .WORD   40
000532  000100                          .WORD   100
000534  000200                          .WORD   200
000536  000400                          .WORD   400
000540  001000                          .WORD   1000
000542  002000                          .WORD   2000
000544  004000                          .WORD   4000
000546  010000                          .WORD   10000
000550  020000                          .WORD   20000
000552  040000                          .WORD   40000
000554  100000                          .WORD   -100000
000556  000020                  PAT15:  .WORD   20
000560  177776                          .WORD   -2
000562  177775                          .WORD   -3
000564  177773                          .WORD   -5
000566  177767                          .WORD   -11
000570  177757                          .WORD   -21
000572  177737                          .WORD   -41
000574  177677                          .WORD   -101
000576  177577                          .WORD   -201
000600  177377                          .WORD   -401
000602  176777                          .WORD   -1001
000604  175777                          .WORD   -2001
000606  173777                          .WORD   -4001
000610  167777                          .WORD   -10001
000612  157777                          .WORD   -20001
000614  137777                          .WORD   -40001
000616  077777                          .WORD   77777
000620  000020                  PAT16:  .WORD   20
000622  133331                          .WORD   -44447
000624  133331                          .WORD   -44447
000626  133331                          .WORD   -44447
000630  155554                          .WORD   -22224
000632  155554                          .WORD   -22224
000634  155554                          .WORD   -22224
000636  133331                          .WORD   -44447
000640  133331                          .WORD   -44447
```

```
000642  155554                          .WORD   -22224
000644  155554                          .WORD   -22224
000646  133331                          .WORD   -44447
000650  155554                          .WORD   -22224
000652  133331                          .WORD   -44447
000654  155554                          .WORD   -22224
000656  133331                          .WORD   -44447
000660  155554                          .WORD   -22224
000662  000025                  PAT17:  .WORD   25
000664  000000                          .WORD   0
000666  106466                          .WORD   -71312
000670  106466                          .WORD   -71312
000672  071311                          .WORD   71311
000674  071311                          .WORD   71311
000676  071311                          .WORD   71311
000700  106466                          .WORD   -71312
000702  106466                          .WORD   -71312
000704  106466                          .WORD   -71312
000706  106466                          .WORD   -71312
000710  071311                          .WORD   71311
000712  071311                          .WORD   71311
000714  071311                          .WORD   71311
000716  071311                          .WORD   71311
000720  071311                          .WORD   71311
000722  106466                          .WORD   -71312
000724  106466                          .WORD   -71312
000726  106466                          .WORD   -71312
000730  106466                          .WORD   -71312
000732  106466                          .WORD   -71312
000734  106466                          .WORD   -71312
000736  000025                  PAT18:  .WORD   25
000740  106466                          .WORD   -71312
000742  000000                          .WORD   0
000744  071311                          .WORD   71311
000746  106466                          .WORD   -71312
000750  106466                          .WORD   -71312
000752  106466                          .WORD   -71312
000754  071311                          .WORD   71311
000756  071311                          .WORD   71311
000760  071311                          .WORD   71311
000762  071311                          .WORD   71311
000764  106466                          .WORD   -71312
000766  106466                          .WORD   -71312
000770  106466                          .WORD   -71312
000772  106466                          .WORD   -71312
000774  106466                          .WORD   -71312
000776  071311                          .WORD   71311
001000  071311                          .WORD   71311
001002  071311                          .WORD   71311
001004  071311                          .WORD   71311
001006  071311                          .WORD   71311
001010  071311                          .WORD   71311
001012  000025                  PAT19:  .WORD   25
```

```
001014  000000                          .WORD   0
001016  134631                          .WORD   -43147
001020  134631                          .WORD   -43147
001022  043146                          .WORD   43146
001024  043146                          .WORD   43146
001026  043146                          .WORD   43146
001030  134631                          .WORD   -43147
001032  134631                          .WORD   -43147
001034  134631                          .WORD   -43147
001036  134631                          .WORD   -43147
001040  043146                          .WORD   43146
001042  043146                          .WORD   43146
001044  043146                          .WORD   43146
001046  043146                          .WORD   43146
001050  043146                          .WORD   43146
001052  134631                          .WORD   -43147
001054  134631                          .WORD   -43147
001056  134631                          .WORD   -43147
001060  134631                          .WORD   -43147
001062  134631                          .WORD   -43147
001064  134631                          .WORD   -43147
001066  000025                  PAT20:  .WORD   25
001070  134631                          .WORD   -43147
001072  000000                          .WORD   0
001074  043146                          .WORD   43146
001076  134631                          .WORD   -43147
001100  134631                          .WORD   -43147
001102  134631                          .WORD   -43147
001104  043146                          .WORD   43146
001106  043146                          .WORD   43146
001110  043146                          .WORD   43146
001112  043146                          .WORD   43146
001114  134631                          .WORD   -43147
001116  134631                          .WORD   -43147
001120  134631                          .WORD   -43147
001122  134631                          .WORD   -43147
001124  134631                          .WORD   -43147
001126  043146                          .WORD   43146
001130  043146                          .WORD   43146
001132  043146                          .WORD   43146
001134  043146                          .WORD   43146
001136  043146                          .WORD   43146
001140  043146                          .WORD   43146
001142  000001                  PAT21:  .WORD   1
001144  000000                          .WORD   0
001146  000114'                 DPA.TBL:.WORD   RDM.CNT
001150  000176'                          .WORD   PAT02
001152  000202'                          .WORD   PAT03
001154  000206'                          .WORD   PAT04
001156  000212'                          .WORD   PAT05
001160  000216'                          .WORD   PAT06
001162  000222'                          .WORD   PAT07
001164  000264'                          .WORD   PAT08
```

```
001166  000326'                         .WORD   PAT09
001170  000370'                         .WORD   PAT10
001172  000374'                         .WORD   PAT11
001174  000436'                         .WORD   PAT12
001176  000510'                         .WORD   PAT13
001200  000514'                         .WORD   PAT14
001202  000556'                         .WORD   PAT15
001204  000620'                         .WORD   PAT16
001206  000662'                         .WORD   PAT17
001210  000736'                         .WORD   PAT18
001212  001012'                         .WORD   PAT19
001214  001066'                         .WORD   PAT20
001216  001142'                         .WORD   PAT21
001220                          TRK.SGN:
001220  000001                          .WORD   1
001222  000001                          .WORD   1
001224  000001                          .WORD   1
001226  000001                          .WORD   1
001230  000000                  BST.CNT:.WORD   0
001232  000000                  BST.DEV:.WORD   0
001234                          CURRENT.VECTOR:
                                        .BLKW   1
001236                          BRLEVEL:.BLKW   1
001240                          COMPARE.DATA:
                                        .BLKB   1

                                .GLOBL  CST, CST.ADDR, DCT, DCT.ADDR, RDRX.ADDR
                                .GLOBL  IRDRX.ADDR, DUPPKT, TALLY, T.ADDR
                                .GLOBL  C.ERR.TBL, MSCP.PKT, IPKT.ADDR
                                .GLOBL  PKT.USE, RETPKT, RP.USE, RP.INDX
                                .GLOBL  RP.ADDR, ELOG.PKT, BUFF.ADDR, BUFF.OWN
                                .GLOBL  IODQ, IODQ.IN, IODQ.OUT, ENTRY.REASON
                                .GLOBL  EOP.FLAG, DUP.FLAGS, CCTLR, CDISK
                                .GLOBL  CUOFF, CTLR.CNT, DUR, QIO, FREE.MEM.ADDR
                                .GLOBL  BYTS.PER.QIO, ST.CODE, SB.CODE
                                .GLOBL  STEP, OF.RC, SA.REG, CMD.TIME
                                .GLOBL  NEX, CRN.LOW, CRN.HIGH, P.INDEX
                                .GLOBL  S.DUPPKT, S.PATTERN, CREDIT.BAL
                                .GLOBL  NEXT.PKT.USE, DBM12, DBM18, DBM19
                                .GLOBL  DBM20, DBM21, DBM23, DBM25, DBM26
                                .GLOBL  DBM29, DBM108, DBM109, DBM110
                                .GLOBL  DBM112, EH.0, EH.1, EH.2, EH.3
                                .GLOBL  EH.4, EH.5, EH.6, EH.7, EH.8, EH.9
                                .GLOBL  EH.10, EH.12, EH.13, MSG.02, MSG.03
                                .GLOBL  EGS.02, EGD.10, EGD.11, EGD.12
                                .GLOBL  EGD.13, EGD.14, EGD.15, EGD.16
                                .GLOBL  EGD.17, EGD.18, EGD.20, EGD.21
                                .GLOBL  EGD.22, EGD.23, EGH.30, CRLF, SWP.ERROR
                                .GLOBL  SWP.XFER, SWP.FLAGS, DUPROUND
                                .GLOBL  SWP.RAT, SWP.DPAT, SWP.UCNT, SWP.UDPAT
                                .GLOBL  L$LUN, L$UNIT, NEX.TRAP, SET.CPAR
                                .GLOBL  SET.UPAR, OUT.IODQ, IN.IODQ, GET.PKT
```

```
                        .GLOBL  PUT.PKT, GET.RETPKT, PUT.RETPKT
                        .GLOBL  GET.IO.BUFF, PUT.IO.BUFF, PUTA.BUFF
                        .GLOBL  SEND, WAIT, DROP.CTLR, DRV.CTLERR
                        .GLOBL  EMSCMD, EMS.22, EMS.EL, EMS.CMP
                        .GLOBL  EMS.10, EMS.12, EMS.13, EMS.14
                        .GLOBL  EMS.18, EMS.21, EMS.30


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==          1000
000400                  BIT8==          400
000200                  BIT7==          200
000100                  BIT6==          100
000040                  BIT5==          40
000020                  BIT4==          20
000010                  BIT3==          10
000004                  BIT2==          4
000002                  BIT1==          2
000001                  BIT0==          1
000040                  EF.START==      40
000037                  EF.RESTART==    37
000036                  EF.CONTINUE==   36
000035                  EF.NEW==        35
000034                  EF.PWR==        34
000340                  PRI07==         340
000300                  PRI06==         300
000240                  PRI05==         240
000200                  PRI04==         200
000140                  PRI03==         140
000100                  PRI02==         100
000040                  PRI01==         40
000000                  PRI00==         0
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


                                        .SBTTL  $T1 TEST SECTION
000000                                  .PSECT  $CODE$,  RO

000000 004137 000000G           $T1:    JSR     R1,$SAVE3               ;                       1830
000004 112737 000001 000000G            MOVB    #1,EOP.FLAG             ;                       1841
000012 112737 000001 001240'            MOVB    #1,COMPARE.DATA         ;                       1842
000020 004737 000000V                   JSR     PC,INIT.TEST            ;                       1843
000024 005003                           CLR     R3                      ; CTLR                  1845
000026 010346                   1$:     MOV     R3,-(SP)                ; CTLR,*                1847
000030 012746 000056                    MOV     #56,-(SP)
000034 004737 000000G                   JSR     PC,BL$MUL
000040 010001                           MOV     R0,R1
000042 022626                           CMP     (SP)+,(SP)+
000044 005761 000002G                   TST     CST+2(R1)
000050 100040                           BPL     5$
000052 010346                           MOV     R3,-(SP)                ; CTLR,*                1848
000054 012746 000022                    MOV     #22,-(SP)
000060 004737 000000G                   JSR     PC,BL$MUL
000064 022626                           CMP     (SP)+,(SP)+
000066 005760 000000G                   TST     DCT(R0)
000072 100027                           BPL     5$
000074 010346                           MOV     R3,-(SP)                ; CTLR,*                1854
000076 012746 000027                    MOV     #27,-(SP)
000102 004737 000000G                   JSR     PC,BL$MUL
000106 012702 000003                    MOV     #3,R2                   ; *,OFFSET              1852
000112 010001                   2$:     MOV     R0,R1                   ;                       1854
000114 060201                           ADD     R2,R1                   ; OFFSET,*
000116 006301                           ASL     R1
000120 032761 020000 000000G            BIT     #20000,CST(R1)
000126 001403                           BEQ     3$
000130 105037 000000G                   CLRB    EOP.FLAG                ;                       1857
000134 000405                           BR      4$                      ;                       1856
000136 062702 000005            3$:     ADD     #5,R2                   ; *,OFFSET              1852
000142 020227 000023                    CMP     R2,#23                  ; OFFSET,*
000146 003761                           BLE     2$
000150 022626                   4$:     CMP     (SP)+,(SP)+
000152 005203                   5$:     INC     R3                      ; CTLR                  1845
000154 000243                           .WORD CLV!CLC

000156 003723                           BLE     1$
000160 132737 000001 000000G            BITB    #1,EOP.FLAG             ;                       1861
000166 001002                           BNE     6$
000170 004737 000000V                   JSR     PC,MULTI.DRIVE          ;                       1863
```

```
000174  000207                          6$:     RTS     PC                              ;                       1830

; Routine Size:  63 words,      Routine Base:  $CODE$ + 0000
; Maximum stack depth per invocation:  7 words



                                        .SBTTL  T1 TEST SECTION
000000  004737  000000'                 T1::
000000                                  1$:     JSR     PC,$T1                          ;                       1863
000004  104466                                  TRAP    66
000006  006000                                  ROR     R0
000010  103773                                  BLO     1$
000012  000207                                  RTS     PC

; Routine Size:  6 words,       Routine Base:  $CODE$ + 0176
; Maximum stack depth per invocation:  2 words
```

```
;          1866  %sbttl 'INITIALIZATION TEST ROUTINES'
;          1867
;          1868  routine INIT_TEST : novalue =
;          1869
;          1870  !+
;          1871  !       THE INITIALIZATION TEST IS DESIGNED TO VERIFY THE EXISTENCE OF THE
;          1872  !       DEVICES AS CONFIGURED BY THE OPERATOR DURING THE HW DIALOG, AND TO
;          1873  !       BRING EACH DEVICE ONLINE IN PREPARATION FOR EITHER THE MULTI-DRIVE TEST
;          1874  !       OR THE DM EXERCISER.
;          1875  !
;          1876  !       BASICALLY, THE DEVICES ARE BROUGHT ONLINE VIA "DRIVER_INIT", WHICH IS
;          1877  !       INVOKED IMMEDIATELY. ANY DEVICES WHICH FAIL DURING THIS PHASE WILL BE
;          1878  !       MARKED OFFLINE IN THEIR DCT AND CST. FOR THOSE DEVICES WHICH SURVIVE
;          1879  !       THE INITIALIZATION, THIS ROUTINE WILL ATTEMPT 1 OR 2 ACCESS COMMANDS TO
;          1880  !       EACH DISK VIA ROUTINE "ACCESS". THE INITIALIZATION TEST IS DEEMED A
;          1881  !       SUCCESS IF A BLOCK ON THE INNER TRACK OF EACH DISK CAN BE ACCESSED.
;          1882  !-
;          1883
;          1884      begin
;          1885      DRIVER_INIT ();                         ! INIT DRIVER DATA AND DEVICES
;          1886
;          1887      incr CTLR from 0 to (MAX_CTLR - 1) do   ! FOR EACH CONTROLLER
;          1888          begin
;          1889          SET_CPAR (.CTLR);                   ! SET UP COMMONLY-USED CONTROLLER-RELATED DATA ITEMS
;          1890
;          1891          if .CST_ADDR [STATE] eql ONLINE     ! IF CONTROLLER IS STILL ALIVE
;          1892          then
;          1893
;          1894              incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do     ! FOR EACH DISK
;          1895
;          1896                  if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and
;          1897                     (.CST_ADDR [.OFFSET, D_STAT] eql ONLINE) and
;          1898                     (not .CST_ADDR [.OFFSET, D_FATAL])
;          1899                  then
;          1900                      begin
;          1901                      SET_UPAR (.OFFSET);         ! SET UP UNIT-RELATED DATA ITEMS
;          1902                      IF SWP_DINT NEQ (.DUP_FLAGS AND SWP_DINT)   ! IF DUP CAUSED INIT THEN SKIP THIS SECTION
;          1903                          THEN ACCESS ();                 ! TRY ACCESS TO INNER TRACK
;          1904                      end;                        ! IF UNIT IS PRESENT AND ONLINE
;          1905
;          1906          end;                                    ! CONTROLLER LOOP
;          1907
;          1908      end;                                        ! ROUTINE INIT_TEST


                                    .SBTTL  INIT.TEST INITIALIZATION TEST ROUTINES
000000  004137  000000G         INIT.TEST:
                                    JSR     R1,$SAVE2                       ;                       1868
000004  004737  000000V                     JSR     PC,DRIVER.INIT          ;                       1885
000010  005002                      CLR     R2                              ; CTLR                  1887
000012  010246                  1$: MOV     R2,-(SP)                        ; CTLR,*                1889
000014  004737  000000G             JSR     PC,SET.CPAR
000020  013700  000000G             MOV     CST.ADDR,R0                     ;                       1891
```

```
000024  005760  000002                          TST     2(R0)
000030  100035                                  BPL     4$
000032  012701  000003                          MOV     #3,R1           ; *,OFFSET                      1894
000036  010100                          2$:     MOV     R1,R0           ; OFFSET,*                      1896
000040  006300                                  ASL     R0
000042  063700  000000G                         ADD     CST.ADDR,R0
000046  032710  040000                          BIT     #40000,(R0)
000052  001417                                  BEQ     3$
000054  032710  020000                          BIT     #20000,(R0)     ;                               1897
000060  001414                                  BEQ     3$
000062  032710  010000                          BIT     #10000,(R0)     ;                               1898
000066  001011                                  BNE     3$
000070  010116                                  MOV     R1,(SP)         ; OFFSET,*                      1901
000072  004737  000000G                         JSR     PC,SET.UPAR
000076  032737  000002  000000G                 BIT     #2,DUP.FLAGS    ;                               1902
000104  001002                                  BNE     3$
000106  004737  000000V                         JSR     PC,ACCESS       ;                               1903
000112  062701  000005                  3$:     ADD     #5,R1           ; *,OFFSET                      1894
000116  020127  000022                          CMP     R1,#22          ; OFFSET,*
000122  003745                                  BLE     2$
000124  005726                          4$:     TST     (SP)+           ;                               1888
000126  005202                                  INC     R2              ; CTLR                          1887
000130  000243                                  .WORD CLV!CLC

000132  003727                                  BLE     1$
000134  000207                                  RTS     PC              ;                               1868

; Routine Size:  47 words,      Routine Base:  $CODE$ + 0212
; Maximum stack depth per invocation:  5 words
```

```
;       1909  routine DRIVER_INIT : novalue =
;       1910
;       1911  !+
;       1912  !       THIS ROUTINE IS EQUIVALENT IN FUNCTION TO THE INITIALIZATION ENTRY
;       1913  !       POINT OF A STANDARD DEVICE DRIVER. ITS RESPONSIBILITY IS TO INITIALIZE
;       1914  !       DRIVER DATA, AND TO BRING EACH RDRX CONTROLLER AND UNIT (DISK)
;       1915  !       ONLINE.
;       1916  !-
;       1917
;       1918      begin
;       1919
;       1920      local
;       1921          PKT_ADDR;
;       1922
;       1923      PKT_ADDR = MSCP_PKT + 10;                   ! ADDR (TEXT + 0) OF FIRST MSCP PACKET
;       1924      NEXT_PKT_USE = 0;                           ! NEXT PACKET TO ALLOCATE
;       1925
;       1926      incr COUNT from 0 to (PKT_CNT - 1) do       ! FOR EACH MSCP PACKET
;       1927          begin
;       1928          PKT_USE [.COUNT] = -1;                  ! MARK PACKET FREE
;       1929          MSCP_PKT [.COUNT, PKT_LO] = .PKT_ADDR;  ! LOAD PKT ADDR INTO BUFFER DESCRIPTOR
;       1930          MSCP_PKT [.COUNT, PKT_HI] = 0;
;       1931          MSCP_PKT [.COUNT, CONNID] = CID_MSCP;   ! SET CONNECTION ID TO MSCP ID
;       1932          PKT_ADDR = .PKT_ADDR + (PKT_LEN * 2); ! ADVANCE ADDR TO NEXT PACKET
;       1933          end;
;       1934
;       1935      incr CTLR from 0 to (MAX_CTLR - 1) do       ! FOR EACH CONTROLLER
;       1936
;       1937          if .CST [.CTLR, IP_ADDR] neqa 0         ! IF CONTROLLER IS PRESENT
;       1938          then
;       1939              begin
;       1940              SET_CPAR (.CTLR);                   ! SET UP CURRENT CONTROLLER PARAMETERS
;       1941              CURRENT_VECTOR = .CST_ADDR [VEC_ADDR];      ! SET CURRENT CONTROLLER'S VECTOR ADDRESS
;       1942              BRLEVEL = .CST_ADDR [BR_LEV] ^ 5;   ! SET CURRENT CONTROLLER'S BR LEVEL
;       1943              CTLR_INIT ();                       ! INIT DEVICE AND CTLR DATA
;       1944
;       1945              if .DCT_ADDR [STAT] eql ONLINE      ! IF CONTROLLER IS STILL ALIVE
;       1946              then
;       1947
;       1948                  incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do ! FOR EACH UNIT (DISK)
;       1949
;       1950                      if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and    ! IF UNIT EXISTS
;       1951                         (not .CST_ADDR [.OFFSET, D_FATAL])
;       1952                      then
;       1953                          begin
;       1954                          SET_UPAR (.OFFSET);     ! SET UP UNIT-RELATED DATA ITEMS
;       1955                          UNIT_INIT ();           ! BRING UNIT ONLINE
;       1956                          end;                    ! IF UNIT EXISTS
;       1957
;       1958              end;                                ! IF CONTROLLER IS PRESENT
;       1959
;       1960      end;                                        ! ROUTINE DRIVER_INIT
```

```
                                        .SBTTL  DRIVER.INIT INITIALIZATION TEST ROUTINES
000000  004137  000000G           DRIVER.INIT:
                                        JSR     R1,$SAVE2                       ;                               1909
000004  012702  000012G                 MOV     #MSCP.PKT+12,R2                 ; *,PKT.ADDR                    1923
000010  105037  000000G                 CLRB    NEXT.PKT.USE                    ;                               1924
000014  005001                          CLR     R1                              ; COUNT                         1926
000016  112761  000377  000000G  1$:    MOVB    #377,PKT.USE(R1)                ; *,*(COUNT)                    1928
000024  010146                          MOV     R1,-(SP)                        ; COUNT,*                       1929
000026  012746  000104                  MOV     #104,-(SP)
000032  004737  000000G                 JSR     PC,BL$MUL
000036  010260  000000G                 MOV     R2,MSCP.PKT(R0)                 ; PKT.ADDR,*
000042  005060  000002G                 CLR     MSCP.PKT+2(R0)                  ;                               1930
000046  105060  000011G                 CLRB    MSCP.PKT+11(R0)                 ;                               1931
000052  062702  000104                  ADD     #104,R2                         ; *,PKT.ADDR                    1932
000056  022626                          CMP     (SP)+,(SP)+                     ;                               1927
000060  005201                          INC     R1                              ; COUNT                         1926
000062  020127  000013                  CMP     R1,#13                          ; COUNT,*
000066  003753                          BLE     1$
000070  005002                          CLR     R2                              ; CTLR                          1935
000072  010246                  2$:     MOV     R2,-(SP)                        ; CTLR,*                        1937
000074  012746  000056                  MOV     #56,-(SP)
000100  004737  000000G                 JSR     PC,BL$MUL
000104  022626                          CMP     (SP)+,(SP)+
000106  005760  000000G                 TST     CST(R0)
000112  001460                          BEQ     6$
000114  010246                          MOV     R2,-(SP)                        ; CTLR,*                        1940
000116  004737  000000G                 JSR     PC,SET.CPAR
000122  013700  000000G                 MOV     CST.ADDR,R0                     ;                               1941
000126  016037  000002  001234'         MOV     2(R0),CURRENT.VECTOR
000134  042737  177000  001234'         BIC     #177000,CURRENT.VECTOR
000142  005016                          CLR     (SP)                            ;                               1942
000144  116016  000004                  MOVB    4(R0),(SP)
000150  012746  000005                  MOV     #5,-(SP)
000154  004737  000000G                 JSR     PC,BL$SHF
000160  010037  001236'                 MOV     R0,BRLEVEL
000164  004737  000000V                 JSR     PC,CTLR.INIT                    ;                               1943
000170  005777  000000G                 TST     @DCT.ADDR                       ;                               1945
000174  100026                          BPL     5$
000176  012701  000003                  MOV     #3,R1                           ; *,OFFSET                      1948
000202  010100                  3$:     MOV     R1,R0                           ; OFFSET,*                      1950
000204  006300                          ASL     R0
000206  063700  000000G                 ADD     CST.ADDR,R0
000212  032710  040000                  BIT     #40000,(R0)
000216  001410                          BEQ     4$
000220  032710  010000                  BIT     #10000,(R0)                     ;                               1951
000224  001005                          BNE     4$
000226  010116                          MOV     R1,(SP)                         ; OFFSET,*                      1954
000230  004737  000000G                 JSR     PC,SET.UPAR
000234  004737  000000V                 JSR     PC,UNIT.INIT                    ;                               1955
000240  062701  000005          4$:     ADD     #5,R1                           ; *,OFFSET                      1948
000244  020127  000022                  CMP     R1,#22                          ; OFFSET,*
000250  003754                          BLE     3$
```

```
000252  022626                          5$:     CMP     (SP)+,(SP)+                             ;                               1939
000254  005202                          6$:     INC     R2                                      ; CTLR                          1935
000256  000243                                  .WORD CLV!CLC

000260  003704                                  BLE     2$
000262  000207                                  RTS     PC                                      ;                               1909

; Routine Size:  90 words,      Routine Base:  $CODE$ + 0350
; Maximum stack depth per invocation:  6 words
```

```
;       1961  routine CTLR_INIT : novalue =
;       1962
;       1963  !+
;       1964  !       THIS "DRIVER" ROUTINE IS CALLED FROM DRIVER_INIT FOR EACH CONTROLLER
;       1965  !       CONFIGURED FOR TESTING. ITS GENERAL PURPOSE IS TO BRING THE RDRX ONLINE
;       1966  !       TO THE HOST. SPECIFICALLY, IT IS WRITTEN TO:
;       1967  !
;       1968  !       1. INITIALIZE DRIVER CONTROLLER DATA, INCLUDING THE DCT,
;       1969  !       2. SET UP THE DEVICE'S INTERRUPT VECTOR ADDRESS,
;       1970  !       3. PERFORM A REGISTER EXISTENCE TEST TO VERIFY THE DEVICE'S PRESENCE,
;       1971  !       4. PERFORM A VECTOR AND BR LEVEL TEST TO VERIFY THE DEVICE'S VECTOR
;       1972  !             ADDRESS AND INTERRUPT REQUEST LEVEL,
;       1973  !       5. DO A HARD INITIALIZATION (FOUR STEPS) ON THE DEVICE.
;       1974  !
;       1975  !       IF ANY OF THESE INITIAL TESTS FAIL, THEN ALL UNITS ASSOCIATED WITH THE
;       1976  !       DEVICE ARE DROPPED.
;       1977  !-
;       1978
;       1979      begin
;       1980
;       1981      local
;       1982          RESULT : byte;
;       1983
;       1984      INI_CTLR_DAT ();                                ! INITIALIZE CONTROLLER DATA
;       1985      SETVEC (.CURRENT_VECTOR, .INT_ADDR [.CCTLR], PRI07);        ! SET DEVICE'S ASSUMED VECTOR ADDRESS
;       1986      DCT_ADDR [IG_INT] = TRUE;                       ! SET "IGNORE INTERRUPT" BIT
;       1987      LSLUN = .CST_ADDR [OF_UN, D_UNIT];              ! GET FIRST UNIT NUMBER OF CONTROLLER
;       1988                                                      ! (USED BY DRS FOR DEVICE-FATAL CTLR ERRORS)
;       1989  IF SWP_DINT NEQ (.DUP_FLAGS AND SWP_DINT)          ! IF DUP CAUSED INIT THEN SKIP THIS SECTION
;       1990  THEN
;       1991      if REG_EXIST () eql FAILURE                     ! REGISTER EXISTENCE TEST
;       1992      then
;       1993          begin
;       1994          DROP_CTLR (.CCTLR, DU_INIT);                ! DROP ALL CONTROLLER'S UNITS
;       1995          return;
;       1996          end;
;       1997
;       1998  IF SWP_DINT NEQ (.DUP_FLAGS AND SWP_DINT)          ! IF DUP CAUSED INIT THEN SKIP THIS SECTION
;       1999  THEN
;       2000      if VEC_BR_TEST () eql FAILURE                   ! VECTOR ADDR AND BR LEVEL TEST
;       2001      then
;       2002          begin
;       2003          DROP_CTLR (.CCTLR, DU_INIT);                ! DROP ALL CONTROLLER'S UNITS
;       2004          return;
;       2005          end;
;       2006
;       2007      RESULT = HARD_INIT ();                          ! ATTEMPT HARD DEVICE INIT
;       2008      DCT_ADDR [IG_INT] = FALSE;                      ! CLAER "IGNORE INTERRUPT" BIT
;       2009
;       2010      if .RESULT eql SUCCESS                          ! IF HARD INIT WAS SUCCESSFUL
;       2011      then
;       2012          begin
;       2013          INI_RRING ();                               ! INITIALIZE RESPONSE RING
```

```
;         2014          WRT_RDRX (RCSA, RC_ALL, SA_GO);          ! SET "GO" BIT (START CTLR POLLING)
;         2015
;         2016          if SET_CTLR_CHAR () eql SUCCESS          ! SET CONTROLLER CHARACTERISTICS
;         2017          then
;         2018              begin
;         2019              DCT_ADDR [STAT] = ONLINE;            ! MARK CONTROLLER ONLINE IN "DRIVER"
;         2020              CST_ADDR [STATE] = ONLINE;           ! MARK CONTROLLER ONLINE IN "PROGRAM"
;         2021              end;
;         2022          end
;         2023
;         2024      else                                         ! HARD INIT FAILED
;         2025          begin
;         2026          DROP_CTLR (.CCTLR, DU_INIT);             ! DROP ALL CONTROLLER'S UNITS
;         2027          end;
;         2028
;         2029      end;                                         ! ROUTINE CTLR_INIT

                                    .SBTTL  CTLR.INIT INITIALIZATION TEST ROUTINES
000000  010146                  CTLR.INIT:
                                    MOV     R1,-(SP)                        ;                       1961
000002  004737  000000V             JSR     PC,INI.CTLR.DAT                 ;                       1984
000006  012746  000340              MOV     #340,-(SP)                      ;                       1985
000012  013700  000000G             MOV     CCTLR,R0
000016  006300                      ASL     R0
000020  016046  000112'              MOV     INT.ADDR(R0),-(SP)
000024  013746  001234'              MOV     CURRENT.VECTOR,-(SP)
000030  012746  000003              MOV     #3,-(SP)
000034  104437                      TRAP    37
000036  052777  040000  000000G     BIS     #40000,@DCT.ADDR                ;                       1986
000044  013700  000000G             MOV     CST.ADDR,R0                     ;                       1987
000050  016001  000006              MOV     6(R0),R1
000054  000301                      SWAB    R1
000056  042701  177760              BIC     #177760,R1
000062  010137  000000G             MOV     R1,L$LUN
000066  032737  000002  000000G     BIT     #2,DUP.FLAGS                    ;                       1989
000074  001025                      BNE     2$
000076  004737  000000V             JSR     PC,REG.EXIST                    ;                       1991
000102  005700                      TST     R0
000104  001410                      BEQ     1$                              ;                       1994
000106  032737  000002  000000G     BIT     #2,DUP.FLAGS                    ;                       1998
000114  001015                      BNE     2$
000116  004737  000000V             JSR     PC,VEC.BR.TEST                  ;                       2000
000122  005700                      TST     R0
000124  001011                      BNE     2$
000126  013716  000000G     1$:     MOV     CCTLR,(SP)                      ;                       2003
000132  012746  000002              MOV     #2,-(SP)
000136  004737  000000G             JSR     PC,DROP.CTLR
000142  062706  000012              ADD     #12,SP                          ;                       2000
000146  000450                      BR      5$                              ;                       2002
000150  004737  000000V     2$:     JSR     PC,HARD.INIT                    ;                       2007
000154  110001                      MOVB    R0,R1                           ; *,RESULT
000156  042777  040000  000000G     BIC     #40000,@DCT.ADDR                ;                       2008
```

```
000164  120127  000001                          CMPB    R1,#1                   ; RESULT,*                      2010
000170  001026                                  BNE     3$
000172  004737  000000V                         JSR     PC,INI.RRING            ;                               2013
000176  012701  000001                          MOV     #1,R1                   ; *,RC.REG                      2014
000202  013700  000000G                         MOV     RDRX.ADDR,R0
000206  010160  000002                          MOV     R1,2(R0)                ; RC.REG,*
000212  004737  000000V                         JSR     PC,SET.CTLR.CHAR        ;                               2016
000216  020027  000001                          CMP     R0,#1
000222  001020                                  BNE     4$
000224  052777  100000  000000G                 BIS     #100000,@DCT.ADDR       ;                               2019
000232  013700  000000G                         MOV     CST.ADDR,R0             ;                               2020
000236  052760  100000  000002                  BIS     #100000,2(R0)
000244  000407                                  BR      4$                      ;                               2010
000246  013716  000000G                 3$:     MOV     CCTLR,(SP)              ;                               2026
000252  012746  000002                          MOV     #2,-(SP)
000256  004737  000000G                         JSR     PC,DROP.CTLR
000262  005726                                  TST     (SP)+                   ;                               2025
000264  062706  000010                  4$:     ADD     #10,SP                  ;                               1979
000270  012601                          5$:     MOV     (SP)+,R1                ;                               1961
000272  000207                                  RTS     PC

; Routine Size:  94 words,      Routine Base:  $CODE$ + 0634
; Maximum stack depth per invocation:  7 words
```

```
;          2030  routine INI_CTLR_DAT : novalue =
;          2031
;          2032  !+
;          2033  !       THIS ROUTINE IS RESPONSIBLE FOR INITIALIZING ALL CONTROLLER-RELATED
;          2034  !       DATA IN THE "DRIVER" PORTION OF THE EXERCISER. THIS INCLUDES THE
;          2035  !       CONTROLLER'S DCT AND OUTSTANDING COMMAND LIST.
;          2036  !
;          2037  !       IMPLICIT INPUTS:
;          2038  !               CCTLR - CURRENT CONTROLLER NUMBER
;          2039  !               DCT_ADDR - ADDRESS OF CURENT CONTROLLER'S DCT
;          2040  !-
;          2041
;          2042      begin
;          2043      DCT_ADDR [WORD0] = 0;                                   ! CLEAR FIRST DCT WORD
;          2044      DCT_ADDR [RR_BEG] = COMM_AREA + 8 + (.CCTLR * COMM_LEN * 2);   ! START OF RESPONSE RING
;          2045      DCT_ADDR [RR_END] = .DCT_ADDR [RR_BEG] + ((RRING_LEN - 1) * 4);   ! LAST SLOT IN RESPONSE RING
;          2046      DCT_ADDR [CR_BEG] = .DCT_ADDR [RR_END] + 4;             ! START OF COMMAND RING
;          2047      DCT_ADDR [CR_END] = .DCT_ADDR [CR_BEG] + ((CRING_LEN - 1) * 4);   ! LAST SLOT IN COMMAND RING
;          2048      DCT_ADDR [RR_POLL] = .DCT_ADDR [RR_BEG];                ! FIRST RRING SLOT TO POLL
;          2049      DCT_ADDR [CR_POLL] = DCT_ADDR [CR_NEXT] = .DCT_ADDR [CR_BEG];   ! CRING POLL AND NEXT COMMAND POINTERS
;          2050      end;

                                .SBTTL  INI.CTLR.DAT INITIALIZATION TEST ROUTINES
000000  004137  000000G   INI.CTLR.DAT:
                            JSR     R1,$SAVE2                       ;       2030
000004  013701  000000G                 MOV     DCT.ADDR,R1             ;       2043
000010  005011                          CLR     (R1)
000012  012702  000004                  MOV     #4,R2                   ;       2044
000016  060102                          ADD     R1,R2
000020  013746  000000G                 MOV     CCTLR,-(SP)
000024  012746  000050                  MOV     #50,-(SP)
000030  004737  000000G                 JSR     PC,BL$MUL
000034  062700  000010'                 ADD     #COMM.AREA+10,R0
000040  010012                          MOV     R0,(R2)
000042  010061  000006                  MOV     R0,6(R1)                ;       2045
000046  062761  000014  000006         ADD     #14,6(R1)
000054  012700  000010                  MOV     #10,R0                  ;       2046
000060  060100                          ADD     R1,R0
000062  016110  000006                  MOV     6(R1),(R0)
000066  062710  000004                  ADD     #4,(R0)
000072  011061  000012                  MOV     (R0),12(R1)             ;       2047
000076  062761  000014  000012         ADD     #14,12(R1)
000104  011261  000014                  MOV     (R2),14(R1)             ;       2048
000110  011061  000020                  MOV     (R0),20(R1)             ;       2049
000114  011061  000016                  MOV     (R0),16(R1)
000120  022626                          CMP     (SP)+,(SP)+             ;       2042
000122  000207                          RTS     PC                      ;       2030

; Routine Size:  42 words,      Routine Base:  $CODE$ + 1130
; Maximum stack depth per invocation:  6 words
```

```
;         2051  routine REG_EXIST =
;         2052
;         2053  !+
;         2054  !       THIS IS THE REGISTER EXISTENCE (OR "PROBE") TEST DESIGNED TO VERIFY
;         2055  !       THE PRESENCE OF AN RDRX DEVICE. THIS OBJECTIVE IS ACCOMPLISHED BY
;         2056  !       SETTING UP THE NON-EXISTENT MEMORY (NEX) TRAP VECTOR (LOCATION 4) AND
;         2057  !       ATTEMPTING TO READ WHAT IS ASSUMED TO BE THE DEVICE'S SA AND IP
;         2058  !       REGISTERS. IF THE NEX TRAP HANDLER IS INVOKED DUE TO AN ABSENT DEVICE,
;         2059  !       THEN THE GLOBAL DATUM "NEX" WILL BE SET TO "TRUE". THIS DATUM
;         2060  !       DETERMINES THE SUCCESS / FAILURE VALUE OF THIS ROUTINE.
;         2061  !-
;         2062
;         2063      begin
;         2064
;         2065      local
;         2066          TEMP : word,                          ! TEMP FOR READING SA AND IP
;         2067          DUMMY : word;                         ! AS THE NAME IMPLIES
;         2068
;         2069      if .ENTRY_REASON eql NEW_PASS
;         2070      then
;         2071          return SUCCESS;                       ! SKIP TEST FOR NEXT PASS
;         2072
;         2073      OF_RC = 2;                                ! SET UP TO READ SA FIRST
;         2074
;         2075      do
;         2076          begin
;         2077          NEX = FALSE;                          ! SET TO "TRAP NOT RECEIVED"
;         2078          SETVEC (4, NEX_TRAP, PRI07);          ! SET LOCATION 4 TRAP VECTOR ADDRESS
;         2079          TEMP = .(.RDRX_ADDR + .OF_RC);        ! READ REGISTER (THEN TRAP OR CONTINUE)
;         2080          DUMMY = 0;                            ! DUMMY INSTRUCTION TO COVER TRAP RETURN BUG
;         2081                                                !       (TRAP RETURNS TO NEXT INSTRUCTION)
;         2082          CLRVEC (4);                           ! CLEAR LOCATION 4 TRAP VECTOR ADDRESS
;         2083
;         2084          if .NEX                               ! IF NEX TRAP OCCURRED
;         2085          then
;         2086              begin
;         2087              C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;         2088              ERRDF (10, EGD_10, EMS_10);       ! REGISTER EXISTENCE TEST FAILED
;         2089              SETPRI (PRI00);                   ! LOWER PRIORITY
;         2090              return FAILURE;
;         2091              end
;         2092          else
;         2093              OF_RC = .OF_RC - 2;               ! SET UP FOR IP REG OR QUIT
;         2094
;         2095          end
;         2096      until .OF_RC lss 0;
;         2097
;         2098      return SUCCESS;
;         2099      end;

                                        .SBTTL  REG.EXIST INITIALIZATION TEST ROUTINES
000000  004137  000000G           REG.EXIST:
```

```
                                                JSR     R1,$SAVE2                       ;                       2051
000004  123727  000000G 000005                 CMPB    ENTRY.REASON,#5                 ;                       2069
000012  001461                                  BEQ     3$                              ;                       2071
000014  012737  000002  000000G                MOV     #2,OF.RC                        ;                       2073
000022  005037  000000G                 1$:     CLR     NEX                             ;                       2077
000026  012746  000340                          MOV     #340,-(SP)                      ;                       2078
000032  012746  000000G                         MOV     #NEX.TRAP,-(SP)
000036  012746  000004                          MOV     #4,-(SP)
000042  012746  000003                          MOV     #3,-(SP)
000046  104437                                  TRAP    37
000050  013700  000000G                         MOV     RDRX.ADDR,R0                    ;                       2079
000054  063700  000000G                         ADD     OF.RC,R0
000060  011001                                  MOV     (R0),R1                         ; *,TEMP
000062  005002                                  CLR     R2                              ; DUMMY                 2080
000064  012700  000004                          MOV     #4,R0                           ;                       2082
000070  104436                                  TRAP    36
000072  032737  000001  000000G                BIT     #1,NEX                          ;                       2084
000100  001416                                  BEQ     2$                              ;                       2087
000102  013700  000000G                         MOV     CCTLR,R0                        ;
000106  006300                                  ASL     R0
000110  105260  000000G                         INCB    C.ERR.TBL(R0)
000114  104455                                  TRAP    55                              ;                       2088
000116  000012                                  .WORD   12
000120  000000G                                 .WORD   EGD.10
000122  000000G                                 .WORD   EMS.10
000124  005000                                  CLR     R0                              ;                       2089
000126  104441                                  TRAP    41
000130  062706  000010                          ADD     #10,SP                          ;                       2084
000134  000413                                  BR      4$                              ;                       2086
000136  162737  000002  000000G         2$:     SUB     #2,OF.RC                        ;                       2093
000144  062706  000010                          ADD     #10,SP                          ;                       2076
000150  005737  000000G                         TST     OF.RC                           ;                       2096
000154  002322                                  BGE     1$
000156  012700  000001                 3$:     MOV     #1,R0                           ;                       2063
000162  000207                                  RTS     PC
000164  005000                         4$:     CLR     R0                              ;                       2051
000166  000207                                  RTS     PC

; Routine Size:  60 words,      Routine Base:  $CODE$ + 1254
; Maximum stack depth per invocation:  9 words
```

```
;       2100  routine VEC_BR_TEST =
;       2101
;       2102  !+
;       2103  !       THIS ROUTINE ATTEMPTS TO VERIFY (A) THAT THE RDRX VECTOR ADDRESS GIVEN
;       2104  !       BY THE USER DURING THE HW DIALOG IS VALID, AND (B) THAT THE
;       2105  !       USER-SPECIFIED BUS REQUEST LEVEL FOR THE DEVICE IS CORRECT. THE FIRST
;       2106  !       OBJECTIVE IS ACCOMPLISHED BY SETTING THE CPU PRIORITY TO 0 AND FORCING
;       2107  !       AN RDRX INTERRUPT. IF THE USER SPECIFIED AND INCORRECT VECTOR ADDRESS,
;       2108  !       THEN THE RESULT MAY BE UNPREDICTABLE. FOR THIS REASON, THE MESSAGE
;       2109  !       "FUNCTIONAL TEST STARTED" IS PRINTED BEFORE THE TEST, AND
;       2110  !       "EXERCISER STARTED" IS PRINTED AT ITS SUCCESSFUL CONCLUSION. IF
;       2111  !       EITHER "FUNCTIONAL TEST ..." OR "EXERCISER ..." DOES NOT APPEAR, THEN
;       2112  !       PROGRAM CONTROL IS ASSUMED LOST AND A FATAL TRAP IS LIKELY TO OCCUR. AT
;       2113  !       THIS POINT, THE EXERCISER MUST BE STARTED AGAIN.
;       2114  !
;       2115  !       IF THIS TEST SUCCEEDS, THEN THE BR LEVEL TEST IS RUN BY SETTING THE
;       2116  !       PROCESSOR PRIORITY TO THE ASSUMED INTERRUPT PRIORITY GIVEN BY THE
;       2117  !       USER. A FORCED INTERRUPT SHOULD NOT OCCURR. THEN, BY LOWERING THE
;       2118  !       PRIORITY BY ONE, THE DELAYED INTERRUPT SHOULD OCCUR.
;       2119  !-
;       2120
;       2121      begin
;       2122
;       2123      if .ENTRY_REASON eql NEW_PASS
;       2124      then
;       2125          begin
;       2126          SETPRI (PRI00);                         ! LOWER PRIORITY
;       2127          return SUCCESS;                         ! SKIP TEST IF NEXT PASS
;       2128          end;
;       2129
;       2130      PRINTF (MSG_02);                            ! "FUNCTIONAL TEST STARTED"
;       2131
;       2132      if INT_GEN () eql FALSE                     ! FORCE AN INTERRUPT
;       2133      then
;       2134          begin                                   ! IF INTERRUPT DID NOT OCCUR
;       2135          C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;       2136          ERRDF (11, EGD_11, 0);                  ! VECTOR TEST FAILED
;       2137          return FAILURE;
;       2138          end
;       2139      else
;       2140          begin                                   ! INTERRUPT DID OCCUR
;       2141          PRINTF (MSG_03);                        ! "EXERCISER STARTED"
;       2142          SETPRI (.BRLEVEL);                      ! SET PRIORITY TO ASSUMED BR LEVEL
;       2143
;       2144          if INT_GEN () eql FALSE                 ! FORCE AN INTERRUPT (SHOULD NOT OCCUR)
;       2145          then
;       2146              begin                               ! IF INTERRUPT DID NOT OCCUR
;       2147              SETPRI (.BRLEVEL - %o'40');         ! LOWER PRIORITY BY 1
;       2148              DELAY (1);                          ! WAIT
;       2149
;       2150              if .DCT_ADDR [SA_SAVE] neq 0        ! IF INTERRUPT DID OCCUR (SA_SAVE WOULD BE NON-ZERO)
;       2151              then
;       2152                  begin
```

```
;          2153                    SETPRI (PRI00);                 ! RESTORE PROCESSOR PRIORITY TO 0
;          2154                    return SUCCESS;                 ! ONLY SUCCESSFUL EXIT POINT
;          2155                    end;
;          2156
;          2157                end;
;          2158
;          2159            end;
;          2160
;          2161        SETPRI (PRI00);                             ! COME HERE ONLY FOR BR TEST FAILURE
;          2162        C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;          2163        ERRDF (12, EGD_12, EMS_12);
;          2164        return FAILURE;
;          2165        end;

                                        .GLOBL  L$DLY


                                        .SBTTL  VEC.BR.TEST INITIALIZATION TEST ROUTINES
000000  010146                  VEC.BR.TEST:
                                        MOV     R1,-(SP)                ;                       2100
000002  005746                          TST     -(SP)
000004  123727  000000G 000005         CMPB    ENTRY.REASON,#5         ;                       2123
000012  001003                          BNE     1$
000014  005000                          CLR     R0                      ;                       2126
000016  104441                          TRAP    41
000020  000473                          BR      7$                      ;                       2125
000022  012746  000000G          1$:    MOV     #MSG.02,-(SP)           ;                       2130
000026  012746  000001                  MOV     #1,-(SP)
000032  010600                          MOV     SP,R0                   ; SP,*
000034  104417                          TRAP    17
000036  004737  000000V                JSR     PC,INT.GEN              ;                       2132
000042  005700                          TST     R0
000044  001012                          BNE     2$
000046  013700  000000G                MOV     CCTLR,R0                ;                       2135
000052  006300                          ASL     R0
000054  105260  000000G                INCB    C.ERR.TBL(R0)
000060  104455                          TRAP    55                      ;                       2136
000062  000013                          .WORD   13
000064  000000G                         .WORD   EGD.11
000066  000000                          .WORD   0
000070  000466                          BR      9$                      ;                       2132
000072  012716  000000G          2$:    MOV     #MSG.03,(SP)            ;                       2141
000076  012746  000001                  MOV     #1,-(SP)
000102  010600                          MOV     SP,R0                   ; SP,*
000104  104417                          TRAP    17
000106  013700  001236'                MOV     BRLEVEL,R0              ;                       2142
000112  104441                          TRAP    41
000114  004737  000000V                JSR     PC,INT.GEN              ;                       2144
000120  005700                          TST     R0
000122  001035                          BNE     8$
000124  013700  001236'                MOV     BRLEVEL,R0              ;                       2147
000130  162700  000040                  SUB     #40,R0
```

```
000134  104441                          TRAP    41
000136  012701  000001                  MOV     #1,R1           ; *,$$TMP2              2148
000142  001411                  3$:     BEQ     6$
000144  013700  000000G                 MOV     L$DLY,R0        ; *,$$TMP1
000150  001404                          BEQ     5$
000152  005066  000006          4$:     CLR     6(SP)           ; $$TMP
000156  005300                          DEC     R0              ; $$TMP1
000160  001374                          BNE     4$
000162  005301                  5$:     DEC     R1              ; $$TMP2
000164  000766                          BR      3$
000166  013700  000000G         6$:     MOV     DCT.ADDR,R0     ;                       2150
000172  005760  000002                  TST     2(R0)
000176  001407                          BEQ     8$
000200  005000                          CLR     R0              ;                       2153
000202  104441                          TRAP    41
000204  062706  000006                  ADD     #6,SP           ;                       2150
000210  012700  000001          7$:     MOV     #1,R0           ;                       2152
000214  000416                          BR      10$
000216  005726                  8$:     TST     (SP)+           ;                       2140
000220  005000                          CLR     R0              ;                       2161
000222  104441                          TRAP    41
000224  013700  000000G                 MOV     CCTLR,R0        ;                       2162
000230  006300                          ASL     R0
000232  105260  000000G                 INCB    C.ERR.TBL(R0)
000236  104455                          TRAP    55              ;                       2163
000240  000014                          .WORD   14
000242  000000G                         .WORD   EGD.12
000244  000000G                         .WORD   EMS.12
000246  022626                  9$:     CMP     (SP)+,(SP)+     ;                       2100
000250  005000                          CLR     R0
000252  005726                  10$:    TST     (SP)+
000254  012601                          MOV     (SP)+,R1
000256  000207                          RTS     PC

; Routine Size:  88 words,      Routine Base:  $CODE$ + 1444
; Maximum stack depth per invocation:  7 words
```

```
;        2166  routine INT_GEN =
;        2167
;        2168  !+
;        2169  !       THIS ROUTINE BEGINS AN RDRX INITIALIZATION SEQUENCE, BUT ONLY
;        2170  !       COMPLETES THROUGH THE STEP 1 WRITE. ITS PURPOSE IS TO CREATE AN RDRX
;        2171  !       INTERRUPT (AT THE COMPLETEION OF STEP 1) IN ORDER TO HELP VERIFY THE
;        2172  !       THE USER-SPECIFIED VECTOR ADDRESS AND BUS REQUEST INTERRUPT LEVEL.
;        2173  !       A VALUE OF "TRUE" IS RETURNED TO THE CALLER IF AN INTERRUPT OCCURS,
;        2174  !       AND "FALSE" OTHERWISE. THE INTERRUPT IS VERIFIED BY A NON-ZERO VALUE
;        2175  !       IN THE "SA SAVE" WORD IN THE DEVICE'S DCT.
;        2176  !-
;        2177
;        2178      begin
;        2179
;        2180      local
;        2181          SA : word;                              ! STORAGE FOR STEP 1 READ AND WRITE
;        2182
;        2183      DCT_ADDR [SA_SAVE] = 0;                     ! ZERO OUT SA SAVE WORD IN DCT
;        2184      WRT_RDRX (RCIP, RC_ALL, ALL_ONES);          ! WRITE IP TO START INIT SEQUENCE
;        2185      DELAY (10);                                 ! WAIT
;        2186      SA = .RDRX_ADDR [RCSA, RC_ALL];             ! STEP 1 READ
;        2187      SA = (WR_RING ^ 8) or (.CURRENT_VECTOR ^ -2) or SA_INT;     ! STEP 1 WRITE VALUE
;        2188      WRT_RDRX (RCSA, RC_ALL, .SA);               ! STEP 1 WRITE
;        2189
;        2190      incr COUNT from 1 to 1600 do
;        2191          begin
;        2192          DELAY (5);                              ! TOTAL DELAY COUNT OF 8,000
;        2193          BREAK;
;        2194
;        2195          if .DCT_ADDR [SA_SAVE] neq 0            ! IF SA WAS CHANGED
;        2196          then
;        2197              return TRUE;                        ! INTERRUPT OCCURED
;        2198          end;
;        2199
;        2200      return FALSE;                               ! IF INTERRUPT DID NOT OCCUR
;        2201      end;


                                .SBTTL  INT.GEN INITIALIZATION TEST ROUTINES
000000  004137  000000G  INT.GEN:JSR     R1,$SAVE2                       ;                       2166
000004  024646                   CMP     -(SP),-(SP)
000006  013700  000000G          MOV     DCT.ADDR,R0                     ;                       2183
000012  005060  000002           CLR     2(R0)
000016  012700  177777           MOV     #-1,R0                          ; *,RC.REG              2184
000022  010077  000000G          MOV     R0,@RDRX.ADDR                   ; RC.REG,*
000026  012701  000012           MOV     #12,R1                          ; *,$$TMP2              2185
000032  001411           1$:     BEQ     4$
000034  013700  000000G          MOV     L$DLY,R0                        ; *,$$TMP1
000040  001404                   BEQ     3$
000042  005066  000002   2$:     CLR     2(SP)                           ; $$TMP
000046  005300                   DEC     R0                              ; $$TMP1
000050  001374                   BNE     2$
000052  005301           3$:     DEC     R1                              ; $$TMP2
```

```
000054  000766                          BR      1$
000056  013700  000000G          4$:     MOV     RDRX.ADDR,R0            ;                               2186
000062  016016  000002                  MOV     2(R0),(SP)              ; *,RC.REG
000066  013701  001234'                 MOV     CURRENT.VECTOR,R1       ;                               2187
000072  006201                          ASR     R1
000074  006201                          ASR     R1
000076  010102                          MOV     R1,R2                   ; *,SA
000100  052702  111200                  BIS     #111200,R2              ; *,SA
000104  010201                          MOV     R2,R1                   ; SA,RC.REG                     2188
000106  010160  000002                  MOV     R1,2(R0)                ; RC.REG,*
000112  012702  003100                  MOV     #3100,R2                ; *,COUNT                       2190
000116  012701  000005          5$:     MOV     #5,R1                   ; *,$$TMP2                      2192
000122  001411                  6$:     BEQ     9$
000124  013700  000000G                 MOV     L$DLY,R0                ; *,$$TMP1
000130  001404                          BEQ     8$
000132  005066  000002          7$:     CLR     2(SP)                   ; $$TMP
000136  005300                          DEC     R0                      ; $$TMP1
000140  001374                          BNE     7$
000142  005301                  8$:     DEC     R1                      ; $$TMP2
000144  000766                          BR      6$
000146  104422                  9$:     TRAP    22
000150  013700  000000G                 MOV     DCT.ADDR,R0             ;                               2195
000154  005760  000002                  TST     2(R0)
000160  001403                          BEQ     10$
000162  012700  000001                  MOV     #1,R0                   ;                               2197
000166  000403                          BR      11$
000170  005302                  10$:    DEC     R2                      ; COUNT                         2190
000172  001351                          BNE     5$
000174  005000                          CLR     R0                      ;                               2178
000176  022626                  11$:    CMP     (SP)+,(SP)+             ;                               2166
000200  000207                          RTS     PC

; Routine Size:  65 words,      Routine Base:  $CODE$ + 1724
; Maximum stack depth per invocation:  7 words
```

```
;       2202  routine HARD_INIT =
;       2203
;       2204  !+
;       2205  !       THIS ROUTINE PERFORMS THE FOUR READ / WRITE STEPS REQUIRED TO
;       2206  !       INITIALIZE AN RDRX DEVICE. IF NO READ ERRORS ARE DETECTED IN ANY OF
;       2207  !       THE FOUR STEPS, THEN A SUCCESS VALUE IS RETURNED TO THE CALLER.
;       2208  !       OTHERWISE, ADDITIONAL ATTEMPTS MAY BE MADE TO INITIALIZE THE DEVICE.
;       2209  !       IF ALL ATTEMPTS FAIL, A FAILURE INDICATION IS RETURNED.
;       2210  !-
;       2211
;       2212      begin
;       2213
;       2214      local
;       2215          IE_VEC : word;                              ! IE-BIT-AND-VECTOR-ADDRESS/4 BYTE
;       2216                                                      !   (USED IN STEP 1 WRITE AND STEP 3 READ)
;       2217
;       2218      IE_VEC = .CURRENT_VECTOR ^ -2;                  ! GET VECTOR ADDR/4 (IE = 0)
;       2219
;       2220      incr ATTEMPTS from 1 to INI_ATT do
;       2221          begin
;       2222
;       2223          label
;       2224              STEP_1_READ,
;       2225              STEP_2_READ,
;       2226              STEP_3_READ,
;       2227              STEP_4_READ;
;       2228
;       2229          WRT_RDRX (RCIP, RC_ALL, ALL_ONES);          ! WRITE IP TO START INIT SEQUENCE
;       2230  !
;       2231  !       STEP 1 READ
;       2232  !
;       2233          STEP = 1;
;       2234          STEP_1_READ:
;       2235              begin
;       2236
;       2237              incr COUNT from 1 to 100 do
;       2238                  begin
;       2239                  DELAY (5);                          ! TOTAL DELAY COUNT OF 500 FOR STEP 1
;       2240                  BREAK;
;       2241                  SA_REG = .RDRX_ADDR [RCSA, RC_ALL]; ! READ SA
;       2242
;       2243                  if (.SA_REG and S1_MASK) eql SA_S1  ! IF STEP 1 READ IS O.K.
;       2244                  then
;       2245                      leave STEP_1_READ;
;       2246
;       2247                  end;
;       2248
;       2249              exitloop;
;       2250              end;
;       2251
;       2252  !
;       2253  !       STEP 1 WRITE
;       2254  !
```

```
2255          SA_REG = (WR_RING ^ 8) or .IE_VEC;              ! STEP 1 WRITE VALUE
2256          WRT_RDRX (RCSA, RC_ALL, .SA_REG);               ! STEP 1 WRITE
2257    !
2258    !     STEP 2 READ
2259    !
2260          STEP = .STEP + 1;
2261          STEP_2_READ:
2262              begin
2263
2264              incr COUNT from 1 to 2000 do
2265                  begin
2266                  DELAY (5);                              ! TOTAL DELAY COUNT OF 10,000 FOR STEP 2
2267                  BREAK;
2268                  SA_REG = .RDRX_ADDR [RCSA, RC_ALL];     ! READ SA
2269
2270                  if (.SA_REG and S2_MASK) eql (SA_S2 or WR_RING) ! IF STEP 2 READ IS O.K.
2271                  then
2272                      leave STEP_2_READ;
2273
2274                  end;
2275
2276              exitloop;
2277              end;
2278
2279    !
2280    !     STEP 2 WRITE
2281    !
2282          WRT_RDRX (RCSA, RC_ALL, .DCT_ADDR [RR_BEG]);    ! RINGBASE-LO, PI = 0
2283    !
2284    !     STEP 3 READ
2285    !
2286          STEP = .STEP + 1;
2287          STEP_3_READ:
2288              begin
2289
2290              incr COUNT from 1 to 2000 do
2291                  begin
2292                  DELAY (5);                              ! TOTAL DELAY COUNT OF 10,000 FOR STEP 3 READ
2293                  BREAK;
2294                  SA_REG = .RDRX_ADDR [RCSA, RC_ALL];     ! READ SA
2295
2296                  if (.SA_REG and S3_MASK) eql (SA_S3 or .IE_VEC) ! IF STEP 3 READ IS O.K.
2297                  then
2298                      leave STEP_3_READ;
2299
2300                  end;
2301
2302              exitloop;
2303              end;
2304
2305    !
2306    !     STEP 3 WRITE
2307    !
```

```
;        2308          WRT_RDRX (RCSA, RC_ALL, 0);                    ! PP, RINGBASE-HI = 0
;        2309      !
;        2310      !   STEP 4 READ
;        2311      !
;        2312          STEP = .STEP + 1;
;        2313          STEP_4_READ:
;        2314              begin
;        2315
;        2316              incr COUNT from 1 to 2000 do
;        2317                  begin
;        2318                  DELAY (5);                             ! TOTAL DELAY COUNT OF 10,000 FOR STEP 4 READ
;        2319                  BREAK;
;        2320                  SA_REG = .RDRX_ADDR [RCSA, RC_ALL];    ! READ SA
;        2321
;        2322                  if (.SA_REG and S4_MASK) eql SA_S4     ! IF STEP 4 READ IS O.K.
;        2323                  then
;        2324                      leave STEP_4_READ;
;        2325
;        2326                  end;
;        2327
;        2328              exitloop;
;        2329              end;
;        2330      !
;        2331      !   STEP 4 WRITE
;        2332      !
;        2333          CREDIT_BAL = 1;                                ! START WITH A CREDIT BALANCE = 1
;        2334          WRT_RDRX (RCSA, RC_ALL, 0);                    ! BURST, LF, GO = 0
;        2335          return SUCCESS;                                ! SUCCESS EXIT POINT
;        2336
;        2337          end;                                           ! TRY AGAIN OR GIVE UP
;        2338
;        2339      CREDIT_BAL = 0;                                    ! NO CREDIT BALANCE
;        2340      C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;        2341      ERRDF (13, EGD_13, EMS_13);                        ! INIT SEQUENCE FAILED
;        2342      return FAILURE;
;        2343      end;                                               ! ROUTINE HARD_INIT


                                        .SBTTL  HARD.INIT INITIALIZATION TEST ROUTINES
000000  004137  000000G          HARD.INIT:
                                        JSR     R1,$SAVE5                       ;                               2202
000004  162706  000012                  SUB     #12,SP
000010  013704  001234'                 MOV     CURRENT.VECTOR,R4               ; *,IE.VEC                      2218
000014  006204                          ASR     R4                              ; IE.VEC
000016  006204                          ASR     R4                              ; IE.VEC
000020  012705  000002                  MOV     #2,R5                           ; *,ATTEMPTS                    2220
000024  012700  177777                  MOV     #-1,R0                          ; *,RC.REG                      2229
000030  010077  000000G                 MOV     R0,@RDRX.ADDR                   ; RC.REG,*
000034  012737  000001  000000G         MOV     #1,STEP                         ;                               2233
000042  012702  000144                  MOV     #144,R2                         ; *,COUNT                       2237
000046  012701  000005          1$:     MOV     #5,R1                           ; *,$$TMP2                      2239
000052  001411                  2$:     BEQ     5$
000054  013700  000000G                 MOV     L$DLY,R0                        ; *,$$TMP1
```

```
000060  001404                          BEQ     4$
000062  005066  000010          3$:     CLR     10(SP)                  ; $$TMP
000066  005300                          DEC     R0                      ; $$TMP1
000070  001374                          BNE     3$
000072  005301                  4$:     DEC     R1                      ; $$TMP2
000074  000766                          BR      2$
000076  104422                  5$:     TRAP    22                                              2241
000100  013700  000000G                 MOV     RDRX.ADDR,R0            ;
000104  016066  000002  000006          MOV     2(R0),6(SP)             ; *,RC.REG
000112  016637  000006  000000G         MOV     6(SP),SA.REG            ; RC.REG,*
000120  016600  000006                  MOV     6(SP),R0                ; SA.REG,*              2243
000124  042700  001777                  BIC     #1777,R0
000130  020027  004000                  CMP     R0,#4000
000134  001403                          BEQ     6$                      ;                       2245
000136  005302                          DEC     R2                      ; COUNT                 2237
000140  001342                          BNE     1$
000142  000576                          BR      24$                     ;                       2221
000144  010437  000000G         6$:     MOV     R4,SA.REG               ; IE.VEC,*              2255
000150  052737  111000  000000G         BIS     #111000,SA.REG
000156  013701  000000G                 MOV     SA.REG,R1               ; *,RC.REG              2256
000162  013700  000000G                 MOV     RDRX.ADDR,R0
000166  010160  000002                  MOV     R1,2(R0)                ; RC.REG,*
000172  005237  000000G                 INC     STEP                    ;                       2260
000176  012702  003720                  MOV     #3720,R2                ; *,COUNT               2264
000202  012701  000005          7$:     MOV     #5,R1                   ; *,$$TMP2              2266
000206  001411                  8$:     BEQ     11$
000210  013700  000000G                 MOV     L$DLY,R0                ; *,$$TMP1
000214  001404                          BEQ     10$
000216  005066  000010          9$:     CLR     10(SP)                  ; $$TMP
000222  005300                          DEC     R0                      ; $$TMP1
000224  001374                          BNE     9$
000226  005301                  10$:    DEC     R1                      ; $$TMP2
000230  000766                          BR      8$
000232  104422                  11$:    TRAP    22                                              2268
000234  013700  000000G                 MOV     RDRX.ADDR,R0            ;
000240  016066  000002  000004          MOV     2(R0),4(SP)             ; *,RC.REG
000246  016637  000004  000000G         MOV     4(SP),SA.REG            ; RC.REG,*
000254  016600  000004                  MOV     4(SP),R0                ; SA.REG,*              2270
000260  042700  003400                  BIC     #3400,R0
000264  020027  010222                  CMP     R0,#10222
000270  001403                          BEQ     12$                     ;                       2272
000272  005302                          DEC     R2                      ; COUNT                 2264
000274  001342                          BNE     7$
000276  000534                          BR      26$                     ;                       2221
000300  013700  000000G         12$:    MOV     DCT.ADDR,R0             ;                       2282
000304  016001  000004                  MOV     4(R0),R1                ; *,RC.REG
000310  013700  000000G                 MOV     RDRX.ADDR,R0
000314  010160  000002                  MOV     R1,2(R0)                ; RC.REG,*
000320  005237  000000G                 INC     STEP                    ;                       2286
000324  010403                          MOV     R4,R3                   ; IE.VEC,*              2296
000326  052703  020000                  BIS     #20000,R3
000332  012702  003720                  MOV     #3720,R2                ; *,COUNT               2290
000336  012701  000005          13$:    MOV     #5,R1                   ; *,$$TMP2              2292
```

```
000342  001411                          14$:    BEQ     17$
000344  013700  000000G                         MOV     L$DLY,R0                ; *,$$TMP1
000350  001404                                  BEQ     16$
000352  005066  000010                  15$:    CLR     10(SP)                  ; $$TMP
000356  005300                                  DEC     R0                      ; $$TMP1
000360  001374                                  BNE     15$
000362  005301                          16$:    DEC     R1                      ; $$TMP2
000364  000766                                  BR      14$
000366  104422                          17$:    TRAP    22                                                      2294
000370  013700  000000G                         MOV     RDRX.ADDR,R0            ;
000374  016066  000002  000002                  MOV     2(R0),2(SP)             ; *,RC.REG
000402  016637  000002  000000G                 MOV     2(SP),SA.REG            ; RC.REG,*
000410  016600  000002                          MOV     2(SP),R0                ; SA.REG,*                      2296
000414  042700  003400                          BIC     #3400,R0
000420  020003                                  CMP     R0,R3
000422  001403                                  BEQ     18$                     ;                               2298
000424  005302                                  DEC     R2                      ; COUNT                         2290
000426  001343                                  BNE     13$
000430  000457                                  BR      26$                     ;                               2221
000432  013700  000000G                 18$:    MOV     RDRX.ADDR,R0            ;                               2308
000436  005060  000002                          CLR     2(R0)
000442  005237  000000G                         INC     STEP                    ;                               2312
000446  012703  003720                          MOV     #3720,R3                ; *,COUNT                       2316
000452  012701  000005                  19$:    MOV     #5,R1                   ; *,$$TMP2                      2318
000456  001411                          20$:    BEQ     23$
000460  013700  000000G                         MOV     L$DLY,R0                ; *,$$TMP1
000464  001404                                  BEQ     22$
000466  005066  000010                  21$:    CLR     10(SP)                  ; $$TMP
000472  005300                                  DEC     R0                      ; $$TMP1
000474  001374                                  BNE     21$
000476  005301                          22$:    DEC     R1                      ; $$TMP2
000500  000766                                  BR      20$
000502  104422                          23$:    TRAP    22                                                      2320
000504  013700  000000G                         MOV     RDRX.ADDR,R0            ;
000510  016016  000002                          MOV     2(R0),(SP)              ; *,RC.REG
000514  011637  000000G                         MOV     (SP),SA.REG             ; RC.REG,*
000520  011600                                  MOV     (SP),R0                 ; SA.REG,*                      2322
000522  042700  003777                          BIC     #3777,R0
000526  020027  040000                          CMP     R0,#40000
000532  001403                                  BEQ     25$                     ;                               2324
000534  005303                                  DEC     R3                      ; COUNT                         2316
000536  001345                                  BNE     19$
000540  000413                          24$:    BR      26$                     ;                               2221
000542  012737  000001  000000G        25$:    MOV     #1,CREDIT.BAL           ;                               2333
000550  005001                                  CLR     R1                      ; RC.REG                        2334
000552  013700  000000G                         MOV     RDRX.ADDR,R0
000556  005060  000002                          CLR     2(R0)
000562  012700  000001                          MOV     #1,R0                   ;                               2221
000566  000414                                  BR      27$
000570  005037  000000G                 26$:    CLR     CREDIT.BAL              ;                               2339
000574  013700  000000G                         MOV     CCTLR,R0                ;                               2340
000600  006300                                  ASL     R0
000602  105260  000000G                         INCB    C.ERR.TBL(R0)
```

```
000606  104455                          TRAP    55                      ;                       2341
000610  000015                          .WORD   15
000612  000000G                         .WORD   EGD.13
000614  000000G                         .WORD   EMS.13
000616  005000                          CLR     R0                      ;                       2212
000620  062706  000012          27$:    ADD     #12,SP                  ;                       2202
000624  000207                          RTS     PC

; Routine Size:  203 words,     Routine Base:  $CODE$ + 2126
; Maximum stack depth per invocation:  13 words
```

```
;        2344  routine INI_RRING : novalue =
;        2345
;        2346  !+
;        2347  !       THIS ROUTINE IS RESPONSIBLE FOR ALLOCATING ENOUGH MSCP PACKETS TO
;        2348  !       FILL AN RDRX RESPONSE RING. THE BUFFER DESCRIPTOR OF EACH PACKET
;        2349  !       (LOCATED IN FRONT OF THE PACKET ITSELF) IS LOADED INTO SUCCESSIVE
;        2350  !       RRING SLOTS. NOTE THAT THE BUFFER DESCRIPTORS HAVE BEEN INITIALIZED
;        2351  !       WITH THE FLAG AND OWNERSHIP BITS SET TO "1", MAKING EACH SLOT
;        2352  !       CONTROLLER-OWNED.
;        2353  !
;        2354  !       IMPLICIT INPUTS:
;        2355  !               CCTLR - CURRENT CONTROLLER NUMBER
;        2356  !               DCT_ADDR - ADDRESS OF CURRENT CONTROLLER'S DCT
;        2357  !-
;        2358
;        2359      begin
;        2360
;        2361      local
;        2362          index : word,
;        2363          RRING_ADDR;
;        2364
;        2365      RRING_ADDR = .DCT_ADDR [RR_BEG];                    ! FIRST RESPONSE RING SLOT
;        2366
;        2367      incr COUNT from 1 to RRING_LEN do
;        2368          begin
;        2369          index = GET_PKT (.CCTLR);                       ! GET AN MSCP PACKET
;        2370          .RRING_ADDR = .MSCP_PKT [.index, PKT_LO];       ! LOAD LO-ORDER BUFF DESC INTO SLOT
;        2371          RRING_ADDR = .RRING_ADDR + 2;                   ! ADVANCE TO SECOND WORD
;        2372          .RRING_ADDR = .MSCP_PKT [.index, PKT_HI];       ! LOAD HI-ORDER BUFF DESC INTO SLOT
;        2373          PKT_USE [.index] = .CCTLR;                      ! PACKET IN USE
;        2374          .RRING_ADDR = ..RRING_ADDR or ED_OWN or ED_FLAG;! GIVE OWNERSHIP TO CONTRLLER
;        2375          RRING_ADDR = .RRING_ADDR + 2;                   ! ADVANCE TO NEXT SLOT
;        2376          end;
;        2377
;        2378      end;


                                .SBTTL  INI.RRING INITIALIZATION TEST ROUTINES
000000  004137  000000G   INI.RRING:
                            JSR     R1,$SAVE4                       ;                       2344
000004  013700  000000G             MOV     DCT.ADDR,R0             ;                       2365
000010  016004  000004              MOV     4(R0),R4                ; *,RRING.ADDR
000014  013703  000000G             MOV     CCTLR,R3                ;                       2369
000020  012702  000004              MOV     #4,R2                   ; *,COUNT               2367
000024  010346              1$:     MOV     R3,-(SP)                ;                       2369
000026  004737  000000G             JSR     PC,GET.PKT
000032  010001                      MOV     R0,R1                   ; *,INDEX
000034  010116                      MOV     R1,(SP)                 ; INDEX,*               2370
000036  012746  000104              MOV     #104,-(SP)
000042  004737  000000G             JSR     PC,BL$MUL
000046  016024  000000G             MOV     MSCP.PKT(R0),(R4)+      ; *,RRING.ADDR
000052  016014  000002G             MOV     MSCP.PKT+2(R0),(R4)     ; *,RRING.ADDR          2372
000056  013703  000000G             MOV     CCTLR,R3                ;                       2373
```

```
000062  110361  000000G                        MOVB    R3,PKT.USE(R1)          ; *,*(INDEX)
000066  052724  140000                          BIS     #140000,(R4)+           ; *,RRING.ADDR                  2374
000072  022626                                  CMP     (SP)+,(SP)+             ;                               2368
000074  005302                                  DEC     R2                      ; COUNT                         2367
000076  001352                                  BNE     1$
000100  000207                                  RTS     PC                      ;                               2344

; Routine Size:  33 words,      Routine Base:  $CODE$ + 2754
; Maximum stack depth per invocation:  8 words
```

```
2379  routine SET_CTLR_CHAR =
2380
2381  !+
2382  !       THIS ROUTINE IS CALLED BY CTLR_INIT AFTER THE RDRX HAS BEEN HARD-
2383  !       INITIALIZED. ITS PURPOSE IS TO FORMAT AND SEND THE "SET CONTROLLER
2384  !       CHARACTERISTICS" COMMAND, AND TO VALIDATE THE RESPONSE (END MESSAGE).
2385  !
2386  !       IMPLICIT INPUTS:
2387  !               CCTLR - CURRENT CONTROLLER NUMBER
2388  !-
2389
2390      begin
2391
2392  !+
2393  !       MISCELLANEOUS INITIALIZATON
2394  !-
2395
2396
2397  QIO [.CCTLR] = 0;                                   ! INITIALIZE NO. OF OUTSTANDING QIOS
2398
2399  incr COUNT from 0 to (RP_CNT - 1) do                ! INITIALIZE RETURN PACKET POOL
2400      RP_USE [.COUNT] = -1;
2401
2402  IODQ_IN = IODQ_OUT = 0;                             ! INITIALIZE I/O DONE QUEUE POINTERS
2403
2404
2405      P_INDEX = GET_PKT (.CCTLR);                     ! GET AN MSCP PACKET
2406      MSCP_PKT [.P_INDEX, MSGLEN] = SZ_SCC;           ! PACKET SIZE
2407      MSCP_PKT [.P_INDEX, OPCODE] = OP_SCC;           ! OPCODE = SET CTLR CHAR
2408      MSCP_PKT [.P_INDEX, C_FLAGS] = CF_MASK;         ! CONTROLLER FLAGS
2409      MSCP_PKT [.P_INDEX, CMD_TYPE] = IMM_CMD;        ! IMMEDIATE COMMAND
2410
2411  if SEND (.P_INDEX) eql FAILURE                      ! ATTEMPT SEND
2412  then
2413      begin                                           ! IF SEND WAS UNSUCCESSFUL
2414      C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
2415      ERRDF (20, EGD_20, 0);                          ! FATAL ERROR
2416      PUT_PKT (.P_INDEX);                             ! RETURN PACKET TO POOL
2417      DROP_CTLR (.CCTLR, DU_CFATAL);                  ! DROP CONTROLLER
2418      return FAILURE;
2419      end
2420  else
2421      begin                                           ! IF SEND WAS SUCCESSFUL
2422
2423      do
2424          begin
2425          WAIT ();                                    ! WAIT FOR RETPKT RESPONSE
2426          RP_INDX = OUT_IODQ ();                      ! GET INDEX OF RETPKT
2427          RP_ADDR = RETPKT + (.RP_INDX * RP_LEN * 2); ! CALCULATE RETPKT ADDRESS
2428
2429          if .RP_ADDR [MESTYP] neq MT_SEQ             ! RETURN ALL RETPKTS NOT SENT BY CONTROLLER
2430          then
2431              PUT_RETPKT (.RP_INDX);
```

```
;        2432
;        2433          end
;        2434      until (.RP_ADDR [CONID] eql CID_DRIVER) or
;        2435            ((.RP_ADDR [MESTYP] eql MT_SEQ) and
;        2436              ((.RP_ADDR [ENDCOD] and OP_END) eql OP_END));
;        2437
;        2438      if .RP_ADDR [CONID] eql CID_DRIVER      ! IF RETPKT IS FROM "DRIVER"
;        2439      then
;        2440          begin
;        2441          PRINTF (DBM23);                     ! "ERROR IN SET_CTLR_CHAR"
;        2442          PUT_RETPKT (.RP_INDX);              ! RELEASE RETURN PACKET
;        2443          DR_ERR ();                          ! DROP CONTROLLER
;        2444          return FAILURE;
;        2445          end
;        2446      else
;        2447          begin                               ! ELSE - RETPKT IS FROM DISK MSCP
;        2448
;        2449          if (.RP_ADDR [ENDCOD] neq (OP_SCC or OP_END)) or    ! IF WRONG ENDCODE
;        2450             ((.RP_ADDR [C_FLGS] and CF_MASK) neq CF_MASK)    ! OR FLAGS IN ERROR
;        2451          then
;        2452              begin
;        2453              C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;        2454              ERRDF (21, EGD_21, EMS_21);     ! FATAL ERROR
;        2455              DROP_CTLR (.CCTLR, DU_CFATAL);  ! DROP CONTROLLER
;        2456              PUT_RETPKT (.RP_INDX);          ! RELEASE RETURN PACKET
;        2457              return FAILURE;
;        2458              end
;        2459          else
;        2460              begin                           ! RETPKT HAS CORRECT ENDCODE
;        2461              CMD_TIME = .RP_ADDR [C_TIME] * 2;
;        2462
;        2463              if (.SWP_FLAGS and SWF_TRC) eql SWF_TRC
;        2464              then
;        2465                  PRINTF (DBM25, .RP_ADDR [C_TIME]);
;        2466
;        2467              end; -                          ! RETPKT HAS CORRECT ENDCODE
;        2468
;        2469          end;                                ! IF RETPKT WAS SENT BY DISK MSCP
;        2470
;        2471      PUT_RETPKT (.RP_INDX);
;        2472      return SUCCESS;                         ! IF SEND WAS SUCCESSFUL
;        2473      end;
;        2474
;        2475  end;                                        ! ROUTINE SET-CTLR_CHAR


                                        .SBTTL  SET.CTLR.CHAR INITIALIZATION TEST ROUTINES
000000  010146                  SET.CTLR.CHAR:
                                        MOV     R1,-(SP)                        ;                       2379
000002  013701  000000G                 MOV     CCTLR,R1                        ;                       2397
000006  105061  000000G                 CLRB    QIO(R1)
000012  005000                          CLR     R0                              ; COUNT                 2399
000014  112760  000377  000000G 1$:     MOVB    #377,RP.USE(R0)                 ; *,*(COUNT)            2400
```

```
000022  005200                          INC     R0                              ; COUNT                         2399
000024  020027  000003                  CMP     R0,#3                           ; COUNT,*
000030  003771                          BLE     1$
000032  005037  000000G                 CLR     IODQ.OUT                        ;                               2402
000036  005037  000000G                 CLR     IODQ.IN
000042  010146                          MOV     R1,-(SP)                        ;                               2405
000044  004737  000000G                 JSR     PC,GET.PKT
000050  010037  000000G                 MOV     R0,P.INDEX
000054  010016                          MOV     R0,(SP)                         ; P.INDEX,*                     2406
000056  012746  000104                  MOV     #104,-(SP)
000062  004737  000000G                 JSR     PC,BL$MUL
000066  012760  000040  000006G         MOV     #40,MSCP.PKT+6(R0)
000074  112760  000004  000022G         MOVB    #4,MSCP.PKT+22(R0)              ;                               2407
000102  012760  000120  000030G         MOV     #120,MSCP.PKT+30(R0)            ;                               2408
000110  005060  000004G                 CLR     MSCP.PKT+4(R0)                  ;                               2409
000114  013716  000000G                 MOV     P.INDEX,(SP)                    ;                               2411
000120  004737  000000G                 JSR     PC,SEND
000124  005700                          TST     R0
000126  001026                          BNE     2$
000130  013700  000000G                 MOV     CCTLR,R0                        ;                               2414
000134  006300                          ASL     R0
000136  105260  000000G                 INCB    C.ERR.TBL(R0)
000142  104455                          TRAP    55                              ;                               2415
000144  000024                          .WORD   24
000146  000000G                         .WORD   EGD.20
000150  000000                          .WORD   0
000152  013716  000000G                 MOV     P.INDEX,(SP)                    ;                               2416
000156  004737  000000G                 JSR     PC,PUT.PKT
000162  013716  000000G                 MOV     CCTLR,(SP)                      ;                               2417
000166  012746  000006                  MOV     #6,-(SP)
000172  004737  000000G                 JSR     PC,DROP.CTLR
000176  005726                          TST     (SP)+                           ;                               2413
000200  005000                          CLR     R0                              ;                               2390
000202  000560                          BR      11$
000204  004737  000000G         2$:     JSR     PC,WAIT                         ;                               2425
000210  004737  000000G                 JSR     PC,OUT.IODQ                     ;                               2426
000214  010037  000000G                 MOV     R0,RP.INDX
000220  010016                          MOV     R0,(SP)                         ; RP.INDX,*                     2427
000222  012746  000060                  MOV     #60,-(SP)
000226  004737  000000G                 JSR     PC,BL$MUL
000232  062700  000000G                 ADD     #RETPKT,R0
000236  010037  000000G                 MOV     R0,RP.ADDR
000242  132760  000360  000002         BITB    #360,2(R0)                      ;                               2429
000250  001404                          BEQ     3$
000252  013716  000000G                 MOV     RP.INDX,(SP)                    ;                               2431
000256  004737  000000G                 JSR     PC,PUT.RETPKT
000262  005726                  3$:     TST     (SP)+                           ;                               2424
000264  013701  000000G                 MOV     RP.ADDR,R1                      ;                               2434
000270  005000                          CLR     R0
000272  126127  000003  000003         CMPB    3(R1),#3
000300  001002                          BNE     4$
000302  005200                          INC     R0
000304  000407                          BR      5$
```

```
000306  132761  000360  000002          4$:     BITB    #360,2(R1)                      ;                       2435
000314  001333                                  BNE     2$
000316  105761  000014                          TSTB    14(R1)                          ;                       2436
000322  100330                                  BPL     2$
000324  006000                          5$:     ROR     R0                              ;                       2438
000326  103015                                  BCC     6$
000330  012716  000000G                         MOV     #DBM23,(SP)                     ;                       2441
000334  012746  000001                          MOV     #1,-(SP)
000340  010600                                  MOV     SP,R0                           ; SP,*
000342  104417                                  TRAP    17
000344  013716  000000G                         MOV     RP.INDX,(SP)                    ;                       2442
000350  004737  000000G                         JSR     PC,PUT.RETPKT
000354  004737  000000V                         JSR     PC,DR.ERR                       ;                       2443
000360  000436                                  BR      8$                              ;                       2438
000362  126127  000014  000204          6$:     CMPB    14(R1),#204                     ;                       2449
000370  001007                                  BNE     7$
000372  016100  000022                          MOV     22(R1),R0                       ;                       2450
000376  042700  177657                          BIC     #177657,R0
000402  020027  000120                          CMP     R0,#120
000406  001426                                  BEQ     9$
000410  013700  000000G                 7$:     MOV     CCTLR,R0                        ;                       2453
000414  006300                                  ASL     R0
000416  105260  000000G                         INCB    C.ERR.TBL(R0)
000422  104455                                  TRAP    55                              ;                       2454
000424  000025                                  .WORD   25
000426  000000G                                 .WORD   EGD.21
000430  000000G                                 .WORD   EMS.21
000432  013716  000000G                         MOV     CCTLR,(SP)                      ;                       2455
000436  012746  000006                          MOV     #6,-(SP)
000442  004737  000000G                         JSR     PC,DROP.CTLR
000446  013716  000000G                         MOV     RP.INDX,(SP)                    ;                       2456
000452  004737  000000G                         JSR     PC,PUT.RETPKT
000456  062706  000006                  8$:     ADD     #6,SP                           ;                       2449
000462  000432                                  BR      12$                             ;                       2452
000464  016137  000024  000000G         9$:     MOV     24(R1),CMD.TIME                 ;                       2461
000472  006337  000000G                         ASL     CMD.TIME
000476  032737  000001  000000G                 BIT     #1,SWP.FLAGS                    ;                       2463
000504  001411                                  BEQ     10$
000506  016116  000024                          MOV     24(R1),(SP)                     ;                       2465
000512  012746  000000G                         MOV     #DBM25,-(SP)
000516  012746  000002                          MOV     #2,-(SP)
000522  010600                                  MOV     SP,R0                           ; SP,*
000524  104417                                  TRAP    17
000526  022626                                  CMP     (SP)+,(SP)+
000530  013716  000000G                 10$:    MOV     RP.INDX,(SP)                    ;                       2471
000534  004737  000000G                         JSR     PC,PUT.RETPKT
000540  012700  000001                          MOV     #1,R0                           ;                       2390
000544  022626                          11$:    CMP     (SP)+,(SP)+                     ;                       2379
000546  000401                                  BR      13$                             ;                       2390
000550  005000                          12$:    CLR     R0                              ;                       2379
000552  012601                          13$:    MOV     (SP)+,R1
000554  000207                                  RTS     PC
```

```
; Routine Size:  183 words,     Routine Base:  $CODE$ + 3056
; Maximum stack depth per invocation:  7 words
```

```
;          2476  routine UNIT_INIT : novalue =
;          2477
;          2478  !+
;          2479  !       THIS ROUTINE IS CALLED FROM DRIVER_INIT FOR EACH CONFIGURED UNIT
;          2480  !       (DISK) WHICH IS ATTACHED TO A CONTROLLER THAT SURVIVED
;          2481  !       INITIALIZATION. ITS PURPOSE IS TO FORMAT AND SEND AN "ONLINE"
;          2482  !       MESSAGE, AND TO VERIFY THE RESPONSE.
;          2483  !
;          2484  !       IMPLICIT INPUTS:
;          2485  !               CCTLR - CURRENT CONTROLLER NUMBER
;          2486  !               CDISK - CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;          2487  !               L$LUN - CURRENT (DRS) UNIT NUMBER
;          2488  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;          2489  !-
;          2490
;          2491      begin
;          2492
;          2493
;          2494      P_INDEX = GET_PKT (.CCTLR);                 ! GET AN MSCP PACKET
;          2495      MSCP_PKT [.P_INDEX, MSGLEN] = SZ_ONL;       ! PACKET SIZE
;          2496      MSCP_PKT [.P_INDEX, DK_NUM] = .CDISK;       ! SET DISK ADDRESS (RD/RX DISK NUMBER)
;          2497      MSCP_PKT [.P_INDEX, OPCODE] = OP_ONL;       ! OPCODE FOR "ONLINE"
;          2498      MSCP_PKT [.P_INDEX, DDPAR] = BIT00;         ! SHOW ALL ECC ERRORS IN ERROR LOG MESSAGES
;          2499      MSCP_PKT [.P_INDEX, CMD_TYPE] = SEQ_CMD;    ! SEQUENTIAL COMMAND
;          2500
;          2501      if SEND (.P_INDEX) eql FAILURE              ! ATTEMPT TO SEND; IF CTLR IS OFFLINE
;          2502      then
;          2503          begin
;          2504          T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;          2505          CST_ADDR [.CUOFF, D_FATAL] = TRUE;      ! FATAL ERROR
;          2506          ERRDF (22, EGD_22, 0);                  !
;          2507          DUR [.L$LUN] = DU_ONLINE;               ! SETUP REASON TO DROP UNIT
;          2508          DODU (.L$LUN);                          ! DROP UNIT
;          2509          PUT_PKT (.P_INDEX);                     ! RETURN PACKET TO POOL
;          2510          end
;          2511      else
;          2512          begin                                   ! OTHERWISE (SEND WAS SUCCESSFUL)
;          2513
;          2514          do
;          2515              begin
;          2516              WAIT ();                            ! WAIT FOR RETPKT RESPONSE
;          2517              RP_INDX = OUT_IODQ ();              ! GET INDEX OF RETPKT
;          2518              RP_ADDR = RETPKT + (.RP_INDX * RP_LEN * 2); ! CALCULATE RETPKT ADDRESS
;          2519
;          2520              if .RP_ADDR [MESTYP] neq MT_SEQ     ! RETURN ALL RETPKTS NOT SENT BY CONTROLLER
;          2521              then
;          2522                  PUT_RETPKT (.RP_INDX);
;          2523
;          2524              end
;          2525          until (.RP_ADDR [CONID] eql CID_DRIVER) or
;          2526                ((.RP_ADDR [MESTYP] eql MT_SEQ) and
;          2527                 ((.RP_ADDR [ENDCOD] and OP_END) eql OP_END));
;          2528
```

```
;       2529            if .RP_ADDR [CONID] eql CID_DRIVER      ! IF RETPKT IS FROM "DRIVER"
;       2530            then
;       2531                begin
;       2532                PRINTF (DBM26);                     ! "ERROR IN UNIT_INIT"
;       2533                DR_ERR ();                          ! DROP CONTROLLER
;       2534                end
;       2535            else
;       2536
;       2537                if .RP_ADDR [ENDCOD] neq (OP_ONL or OP_END) ! IF RETPKT IS FROM DISK MSCP
;       2538                then
;       2539                    begin
;       2540                    T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;       2541                    CST_ADDR [.CUOFF, D_FATAL] = TRUE;
;       2542                    ERRDF (23, EGD_23, EMS_21);     ! FATAL ERROR
;       2543                    DUR [.L$LUN] = DU_ONLINE;       ! SETUP REASON TO DROP UNIT
;       2544                    DODU (.L$LUN);                  ! DROP UNIT
;       2545                    end
;       2546                else
;       2547                    begin                           ! RETPKT HAS GOOD ENDCODE
;       2548                    ST_CODE = .RP_ADDR [STSCOD];    ! GET STATUS CODE
;       2549                    SB_CODE = .RP_ADDR [SUBCOD];    ! GET SUB-CODE
;       2550
;       2551                    if .ST_CODE neq ST_SUC          ! IF STATUS CODE IS NOT SUCCESSFUL
;       2552                    then
;       2553                        begin
;       2554                        T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;       2555                        CST_ADDR [.CUOFF, D_FATAL] = TRUE;
;       2556                        ERRDF (15, EGD_15, EMS_30); ! ONLINE FAILED
;       2557                        DUR [.L$LUN] = DU_ONLINE;   ! SET UP REASON FOR DROPPING UNIT
;       2558                        DODU (.L$LUN);              ! DROP UNIT
;       2559                        end
;       2560                    else
;       2561
;       2562                        if ((.RP_ADDR [U_FLGS] and UF_WPH) eql UF_WPH) and  ! STATUS CODE IS O.K.
;       2563                           (.CST_ADDR [.CUOFF, D_PROT] eql UNPROTECTED)
;       2564                        then
;       2565                            begin
;       2566                            T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;       2567                            CST_ADDR [.CUOFF, D_FATAL] = TRUE;
;       2568                            ERRDF (16, EGD_16, EMS_30);             ! WRITE-PROTECT CONFLICT
;       2569                            DUR [.L$LUN] = DU_PROTECT;      ! SET REASON TO DROP UNIT
;       2570                            DODU (.L$LUN);                  ! DROP UNIT
;       2571                            end
;       2572                        else
;       2573                            begin                                       ! WRITE PROTECT SWITCH IS O.K.
;       2574                            MAX_LBN [.L$LUN] = .RP_ADDR [USIZ_LO] - 1;  ! LARGEST LBN
;       2575                            CST_ADDR [.CUOFF, D_STAT] = ONLINE;
;       2576                            CST [.CCTLR, U_CNT] = .CST [.CCTLR, U_CNT] + 1;
;       2577                            end;
;       2578
;       2579                    end;                            ! IF RETPKT HAS CORRECT ENDCODE
;       2580
;       2581        PUT_RETPKT (.RP_INDX);
```

```
;       2582            end;                                    ! IF SEND WAS SUCCESSFUL
;       2583
;       2584        end;                                        ! ROUTINE UNIT_INIT


                                        .SBTTL  UNIT.INIT INITIALIZATION TEST ROUTINES
000000  004137  000000G           UNIT.INIT:
                                        JSR     R1,$SAVE3                       ;                       2476
000004  013746  000000G                 MOV     CCTLR,-(SP)                     ;                       2494
000010  004737  000000G                 JSR     PC,GET.PKT
000014  010037  000000G                 MOV     R0,P.INDEX
000020  010016                          MOV     R0,(SP)                         ; P.INDEX,*             2495
000022  012746  000104                  MOV     #104,-(SP)
000026  004737  000000G                 JSR     PC,BL$MUL
000032  012760  000044  000006G         MOV     #44,MSCP.PKT+6(R0)
000040  013760  000000G  000016G         MOV     CDISK,MSCP.PKT+16(R0)           ;                       2496
000046  112760  000011  000022G         MOVB    #11,MSCP.PKT+22(R0)             ;                       2497
000054  012760  000001  000046G         MOV     #1,MSCP.PKT+46(R0)              ;                       2498
000062  012760  000001  000004G         MOV     #1,MSCP.PKT+4(R0)               ;                       2499
000070  013716  000000G                 MOV     P.INDEX,(SP)                    ;                       2501
000074  004737  000000G                 JSR     PC,SEND
000100  005700                          TST     R0
000102  001033                          BNE     1$
000104  013700  000000G                 MOV     T.ADDR,R0                       ;                       2504
000110  105260  000063                  INCB    63(R0)
000114  013700  000000G                 MOV     CUOFF,R0                        ;                       2505
000120  006300                          ASL     R0
000122  063700  000000G                 ADD     CST.ADDR,R0
000126  052710  010000                  BIS     #10000,(R0)
000132  104455                          TRAP    55                              ;                       2506
000134  000026                          .WORD   26
000136  000000G                         .WORD   EGD.22
000140  000000                          .WORD   0
000142  013700  000000G                 MOV     L$LUN,R0                        ;                       2507
000146  112760  000007  000000G         MOVB    #7,DUR(R0)
000154  104451                          TRAP    51                              ;                       2508
000156  013716  000000G                 MOV     P.INDEX,(SP)                    ;                       2509
000162  004737  000000G                 JSR     PC,PUT.PKT
000166  000137  004534'                 JMP     11$                             ;                       2501
000172  004737  000000G           1$:   JSR     PC,WAIT                         ;                       2516
000176  004737  000000G                 JSR     PC,OUT.IODQ                     ;                       2517
000202  010037  000000G                 MOV     R0,RP.INDX
000206  010016                          MOV     R0,(SP)                         ; RP.INDX,*             2518
000210  012746  000060                  MOV     #60,-(SP)
000214  004737  000000G                 JSR     PC,BL$MUL
000220  062700  000000G                 ADD     #RETPKT,R0
000224  010037  000000G                 MOV     R0,RP.ADDR
000230  132760  000360  000002         BITB    #360,2(R0)                      ;                       2520
000236  001404                          BEQ     2$
000240  013716  000000G                 MOV     RP.INDX,(SP)                    ;                       2522
000244  004737  000000G                 JSR     PC,PUT.RETPKT
000250  005726                    2$:   TST     (SP)+                           ;                       2515
000252  013701  000000G                 MOV     RP.ADDR,R1                      ;                       2525
```

```
000256  005000                          CLR     R0
000260  126127  000003  000003          CMPB    3(R1),#3
000266  001002                          BNE     3$
000270  005200                          INC     R0
000272  000407                          BR      4$
000274  132761  000360  000002  3$:     BITB    #360,2(R1)              ;                       2526
000302  001333                          BNE     1$
000304  105761  000014                  TSTB    14(R1)                  ;                       2527
000310  100330                          BPL     1$
000312  006000                  4$:     ROR     R0                      ;                       2529
000314  103011                          BCC     5$
000316  012716  000000G                 MOV     #DBM26,(SP)             ;                       2532
000322  012746  000001                  MOV     #1,-(SP)
000326  010600                          MOV     SP,R0                   ; SP,*
000330  104417                          TRAP    17
000332  004737  000000V                 JSR     PC,DR.ERR               ;                       2533
000336  000553                          BR      9$                      ;                       2531
000340  013703  000000G          5$:     MOV     CUOFF,R3                ;                       2541
000344  006303                          ASL     R3
000346  063703  000000G                 ADD     CST.ADDR,R3
000352  013702  000000G                 MOV     L$LUN,R2                ;                       2543
000356  126127  000014  000211          CMPB    14(R1),#211             ;                       2537
000364  001420                          BEQ     6$
000366  013700  000000G                 MOV     T.ADDR,R0               ;                       2540
000372  105260  000062                  INCB    62(R0)
000376  052713  010000                  BIS     #10000,(R3)             ;                       2541
000402  104455                          TRAP    55                      ;                       2542
000404  000027                          .WORD   27
000406  000000G                         .WORD   EGD.23
000410  000000G                         .WORD   EMS.21
000412  112762  000007  000000G         MOVB    #7,DUR(R2)              ;                       2543
000420  010200                          MOV     R2,R0                   ;                       2544
000422  104451                          TRAP    51
000424  000521                          BR      10$                     ;                       2537
000426  116137  000016  000000G  6$:     MOVB    16(R1),ST.CODE          ;                       2548
000434  042737  177740  000000G         BIC     #177740,ST.CODE
000442  016100  000016                  MOV     16(R1),R0               ;                       2549
000446  006200                          ASR     R0
000450  006200                          ASR     R0
000452  006200                          ASR     R0
000454  006200                          ASR     R0
000456  006200                          ASR     R0
000460  042700  174000                  BIC     #174000,R0
000464  010037  000000G                 MOV     R0,SB.CODE
000470  005737  000000G                 TST     ST.CODE                 ;                       2551
000474  001420                          BEQ     7$
000476  013700  000000G                 MOV     T.ADDR,R0               ;                       2554
000502  105260  000062                  INCB    62(R0)
000506  052713  010000                  BIS     #10000,(R3)             ;                       2555
000512  104455                          TRAP    55                      ;                       2556
000514  000017                          .WORD   17
000516  000000G                         .WORD   EGD.15
000520  000000G                         .WORD   EMS.30
```

```
000522  112762  000007  000000G          MOVB    #7,DUR(R2)                      ;                       2557
000530  010200                           MOV     R2,R0                           ;                       2558
000532  104451                           TRAP    51
000534  000455                           BR      10$                             ;                       2551
000536  032761  020000  000022  7$:      BIT     #20000,22(R1)                   ;                       2562
000544  001427                           BEQ     8$
000546  013700  000000G                  MOV     CUOFF,R0                        ;                       2563
000552  006300                           ASL     R0
000554  063700  000000G                  ADD     CST.ADDR,R0
000560  005710                           TST     (R0)
000562  100020                           BPL     8$
000564  013700  000000G                  MOV     T.ADDR,R0                       ;                       2566
000570  105260  000062                   INCB    62(R0)
000574  052713  010000                   BIS     #10000,(R3)                     ;                       2567
000600  104455                           TRAP    55                              ;                       2568
000602  000020                           .WORD   20
000604  000000G                          .WORD   EGD.16
000606  000000G                          .WORD   EMS.30
000610  112762  000011  000000G          MOVB    #11,DUR(R2)                     ;                       2569
000616  010200                           MOV     R2,R0                           ;                       2570
000620  104451                           TRAP    51
000622  000422                           BR      10$                             ;                       2562
000624  010200              8$:          MOV     R2,R0                           ;                       2574
000626  006300                           ASL     R0
000630  016160  000044  000064'          MOV     44(R1),MAX.LBN(R0)
000636  005360  000064'                   DEC     MAX.LBN(R0)
000642  052713  020000                   BIS     #20000,(R3)                     ;                       2575
000646  013716  000000G                  MOV     CCTLR,(SP)                      ;                       2576
000652  012746  000056                   MOV     #56,-(SP)
000656  004737  000000G                  JSR     PC,BL$MUL
000662  105260  000005G                  INCB    CST+5(R0)
000666  005726              9$:          TST     (SP)+                           ;                       2573
000670  013716  000000G     10$:         MOV     RP.INDX,(SP)                    ;                       2581
000674  004737  000000G                  JSR     PC,PUT.RETPKT
000700  022626              11$:         CMP     (SP)+,(SP)+                     ;                       2491
000702  000207                           RTS     PC                              ;                       2476

; Routine Size:  226 words,    Routine Base:  $CODE$ + 3634
; Maximum stack depth per invocation:  9 words
```

```
;        2585  routine DR_ERR : novalue =
;        2586
;        2587  !+
;        2588  !       THIS ROUTINE IS DESIGNED TO PROCESS RETURN PACKETS THAT ORIGINATE AT
;        2589  !       THE "DRIVER" RATHER THAN THE DEVICE. DRIVER-ORIGINATED PACKETS INDICATE
;        2590  !       EITHER A FATAL DEVICE ERROR OR A COMMAND TIMEOUT. SINCE THIS ROUTINE IS
;        2591  !       ONLY CALLED DURING THE INITIALIZATION TEST, IT TREATS A COMMAND TIMEOUT
;        2592  !       AS AN INITIALIZATION ERROR.
;        2593  !
;        2594  !       IMPLICIT INPUTS:
;        2595  !               RP_ADDR - ADDRESS OF A RETPKT THAT ORIGINATED AT THE "DRIVER"
;        2596  !                       (I.E., CONNECTION ID = CID_DRIVER)
;        2597  !-
;        2598
;        2599      begin
;        2600
;        2601      local
;        2602          REASON : word initial (DU_TIME);        ! ASSUME COMMAND TIMEOUT
;        2603
;        2604      if .RP_ADDR [MESTYP] eql MT_FATAL           ! IF FATAL DEVICE ERROR
;        2605      then
;        2606          REASON = DU_DFATAL;                     ! CHANGE REASON TO FATAL ERROR
;        2607
;        2608      DROP_CTLR (.CCTLR, .REASON);                ! DROP ALL UNITS ON CONTROLLER
;        2609      end;


                                .SBTTL  DR.ERR INITIALIZATION TEST ROUTINES
000000  010146                  DR.ERR: MOV     R1,-(SP)                        ;                       2585
000002  012701  000012                  MOV     #12,R1                          ; *,REASON              2599
000006  013700  000000G                 MOV     RP.ADDR,R0                      ;                       2604
000012  116000  000002                  MOVB    2(R0),R0
000016  042700  177417                  BIC     #177417,R0
000022  020027  000060                  CMP     R0,#60
000026  001002                          BNE     1$
000030  012701  000005                  MOV     #5,R1                           ; *,REASON              2606
000034  013746  000000G         1$:     MOV     CCTLR,-(SP)                     ;                       2608
000040  010146                          MOV     R1,-(SP)                        ; REASON,*
000042  004737  000000G                 JSR     PC,DROP.CTLR
000046  022626                          CMP     (SP)+,(SP)+                     ;                       2599
000050  012601                          MOV     (SP)+,R1                        ;                       2585
000052  000207                          RTS     PC

; Routine Size:  22 words,      Routine Base:  $CODE$ + 4540
; Maximum stack depth per invocation:  4 words
```

```
;
;
;
```

```
;          2663                    end
;          2664                until (.RP_ADDR [CONID] eql CID_DRIVER) or
;          2665                      ((.RP_ADDR [MESTYP] eql MT_SEQ) and
;          2666                       ((.RP_ADDR [ENDCOD] and OP_END) eql OP_END));
;          2667
;          2668                if .RP_ADDR [CONID] eql CID_DRIVER  ! IF RETPKT CAME FROM "DRIVER"
;          2669                then
;          2670                    PASS = 2                        ! NO MORE TRIES
;          2671                else
;          2672
;          2673                    if .RP_ADDR [ENDCOD] neq (OP_ACC or OP_END)
;          2674                    then
;          2675                        begin
;          2676                        PRINTF (DBM29);             ! "RETPKT HAS BAD ENDCODE"
;          2677                        EMSCMD ();
;          2678                        end
;          2679                    else
;          2680                        begin                       ! RETPKT HAS CORRECT ENDCODE
;          2681                        ST_CODE = .RP_ADDR [STSCOD];! GET STATUS CODE FROM PACKET
;          2682                        SB_CODE = .RP_ADDR [SUBCOD];! GET SUB-CODE FROM PACKET
;          2683
;          2684                        if .ST_CODE eql ST_SUC      ! IF STATUS CODE INDICATES SUCCESS
;          2685                        then
;          2686                            begin
;          2687                            RESULT = SUCCESS;
;          2688                            PASS = 2;               ! NO NEED TO TRY AGAIN
;          2689                            end;
;          2690
;          2691                        end;                        ! IF RETPKT HAS CORRECT ENDCODE
;          2692
;          2693                    PUT_RETPKT (.RP_INDX);
;          2694                    end;                            ! IF SEND WAS SUCCESSFUL
;          2695
;          2696                LBN = .LBN + 1;                     ! ADVANCE TO FIRST LBN OF BOTTOM SURFACE
;          2697                PASS = .PASS + 1;                   ! SECOND PASS
;          2698                end                                 ! END OF PASS LOOP
;          2699            until .PASS gequ 3;
;          2700
;          2701            if .RESULT eql FAILURE
;          2702            then
;          2703                begin
;          2704                T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;          2705                CST_ADDR [.CUOFF, D_FATAL] = TRUE;  ! FATAL ERROR
;          2706                ERRDF (17, EGD_17, EMS_30);                 ! ACCESS FAILED
;          2707                DUR [.L$LUN] = DU_ACCESS;           ! SET REASON TO DROP UNIT
;          2708                DODU (.L$LUN);                      ! DROP UNIT
;          2709                end;                                ! IF ACCESS FAILED
;          2710
;          2711            end;                                    ! ROUTINE ACCESS

                                        .SBTTL  ACCESS INITIALIZATION TEST ROUTINES
000000  004137  000000G           ACCESS: JSR     R1,$SAVE4                       ;                                   2610
```

```
000004  005003                          CLR     R3                              ; RESULT                        2624
000006  012702  000001                  MOV     #1,R2                           ; *,PASS
000012  005037  000000G                 CLR     SB.CODE                         ;                               2631
000016  005037  000000G                 CLR     ST.CODE
000022  013700  000000G                 MOV     L$LUN,R0                        ;                               2632
000026  006300                          ASL     R0
000030  016000  000064'                  MOV     MAX.LBN(R0),R0
000034  060200                          ADD     R2,R0
000036  006200                          ASR     R0
000040  010004                          MOV     R0,R4                           ; *,LBN
000042  042704  100000                  BIC     #100000,R4                      ; *,LBN
000046  005304                          DEC     R4                              ; LBN
000050  013746  000000G         1$:     MOV     CCTLR,-(SP)                     ;                               2637
000054  004737  000000G                 JSR     PC,GET.PKT
000060  010037  000000G                 MOV     R0,P.INDEX
000064  010016                          MOV     R0,(SP)                         ; P.INDEX,*                     2638
000066  012746  000104                  MOV     #104,-(SP)
000072  004737  000000G                 JSR     PC,BL$MUL
000076  013760  000000G  000016G        MOV     CDISK,MSCP.PKT+16(R0)
000104  112760  000020  000022G        MOVB    #20,MSCP.PKT+22(R0)             ;                               2639
000112  012760  001000  000026G        MOV     #1000,MSCP.PKT+26(R0)           ;                               2640
000120  010460  000046G                 MOV     R4,MSCP.PKT+46(R0)              ; LBN,*                         2641
000124  012760  000002  000004G        MOV     #2,MSCP.PKT+4(R0)               ;                               2642
000132  013716  000000G                 MOV     P.INDEX,(SP)                    ;                               2644
000136  004737  000000G                 JSR     PC,SEND
000142  005700                          TST     R0
000144  001007                          BNE     2$
000146  013716  000000G                 MOV     P.INDEX,(SP)                    ;                               2647
000152  004737  000000G                 JSR     PC,PUT.PKT
000156  012702  000002                  MOV     #2,R2                           ; *,PASS                        2648
000162  000524                          BR      9$                              ;                               2644
000164  004737  000000G         2$:     JSR     PC,WAIT                         ;                               2655
000170  004737  000000G                 JSR     PC,OUT.IODQ                     ;                               2656
000174  010037  000000G                 MOV     R0,RP.INDX
000200  010016                          MOV     R0,(SP)                         ; RP.INDX,*                     2657
000202  012746  000060                  MOV     #60,-(SP)
000206  004737  000000G                 JSR     PC,BL$MUL
000212  062700  000000G                 ADD     #RETPKT,R0
000216  010037  000000G                 MOV     R0,RP.ADDR
000222  132760  000360  000002        BITB    #360,2(R0)                      ;                               2659
000230  001404                          BEQ     3$
000232  013716  000000G                 MOV     RP.INDX,(SP)                    ;                               2661
000236  004737  000000G                 JSR     PC,PUT.RETPKT
000242  005726                  3$:     TST     (SP)+                           ;                               2654
000244  013701  000000G                 MOV     RP.ADDR,R1                      ;                               2664
000250  005000                          CLR     R0
000252  126127  000003  000003        CMPB    3(R1),#3
000260  001002                          BNE     4$
000262  005200                          INC     R0
000264  000407                          BR      5$
000266  132761  000360  000002  4$:     BITB    #360,2(R1)                      ;                               2665
000274  001333                          BNE     2$
000276  105761  000014                  TSTB    14(R1)                          ;                               2666
```

```
000302  100330                          BPL     2$
000304  006000                  5$:     ROR     R0                      ;                       2668
000306  103444                          BLO     7$                      ;                       2670
000310  126127  000014  000220          CMPB    14(R1),#220             ;                       2673
000316  001412                          BEQ     6$
000320  012716  000000G                 MOV     #DBM29,(SP)             ;                       2676
000324  012746  000001                  MOV     #1,-(SP)
000330  010600                          MOV     SP,R0                   ; SP,*
000332  104417                          TRAP    17
000334  004737  000000G                 JSR     PC,EMSCMD               ;                       2677
000340  005726                          TST     (SP)+                   ;                       2675
000342  000430                          BR      8$                      ;                       2673
000344  116137  000016  000000G  6$:     MOVB    16(R1),ST.CODE          ;                       2681
000352  042737  177740  000000G          BIC     #177740,ST.CODE
000360  016100  000016                  MOV     16(R1),R0               ;                       2682
000364  006200                          ASR     R0
000366  006200                          ASR     R0
000370  006200                          ASR     R0
000372  006200                          ASR     R0
000374  006200                          ASR     R0
000376  042700  174000                  BIC     #174000,R0
000402  010037  000000G                 MOV     R0,SB.CODE
000406  005737  000000G                 TST     ST.CODE                 ;                       2684
000412  001004                          BNE     8$
000414  012703  000001                  MOV     #1,R3                   ; *,RESULT              2687
000420  012702  000002          7$:     MOV     #2,R2                   ; *,PASS                2688
000424  013716  000000G          8$:     MOV     RP.INDX,(SP)            ;                       2693
000430  004737  000000G                 JSR     PC,PUT.RETPKT
000434  005204                  9$:     INC     R4                      ; LBN                   2696
000436  005202                          INC     R2                      ; PASS                  2697
000440  022626                          CMP     (SP)+,(SP)+             ;                       2636
000442  020227  000003                  CMP     R2,#3                   ; PASS,*                2699
000446  103600                          BLO     1$
000450  005703                          TST     R3                      ; RESULT                2701
000452  001025                          BNE     10$
000454  013700  000000G                 MOV     T.ADDR,R0               ;                       2704
000460  105260  000062                  INCB    62(R0)
000464  013700  000000G                 MOV     CUOFF,R0                ;                       2705
000470  006300                          ASL     R0
000472  063700  000000G                 ADD     CST.ADDR,R0
000476  052710  010000                  BIS     #10000,(R0)
000502  104455                          TRAP    55                      ;                       2706
000504  000021                          .WORD   21
000506  000000G                         .WORD   EGD.17
000510  000000G                         .WORD   EMS.30
000512  013700  000000G                 MOV     L$LUN,R0                ;                       2707
000516  112760  000010  000000G         MOVB    #10,DUR(R0)
000524  104451                          TRAP    51                      ;                       2708
000526  000207                  10$:    RTS     PC                      ;                       2610

; Routine Size:  172 words,    Routine Base:  $CODE$ + 4614
; Maximum stack depth per invocation:  10 words
```

```
;       2712  %sbttl 'MULTI-DRIVE TEST ROUTINES'
;       2713
;       2714  routine MULTI_DRIVE : novalue =
;       2715
;       2716  !+
;       2717  !       THIS SUBTEST IS THE MOST SIGNIFICANT PART OF THE ENTIRE PROGRAM. THE
;       2718  !       MULTI-DRIVE TEST IS A HOST-CONTROLLED EXERCISER DESIGNED TO GIVE THE
;       2719  !       USER AN INDICATION OF HOW ONE OR SEVERAL RDRX DRIVES WOULD PERFORM IN
;       2720  !       AN OPERATING SYSTEM ENVIRONMENT.
;       2721  !
;       2722  !       THIS ROUTINE ACTS AS AN "EXECUTIVE" TO THE WHOLE PROCESS. AFTER
;       2723  !       INVOKING MD_INIT TO INITIALIZE MULTI-DRIVE TEST DATA, THIS ROUTINE
;       2724  !       ENTERS A LOOP WHICH ISSUES QIOS TO ALL ACTIVE CONTROLLERS AND PROCESSES
;       2725  !       ANY RESPONSES. IN ADDITION, ALL OUTSTANDING COMMANDS ARE TIMED IN
;       2726  !       DRV_TIMCHK WHICH IS INVOKED EVERY SECOND. NORMAL TERMINATION OF THIS
;       2727  !       LOOP OCCURS WHEN QIOS ARE NO LONGER BEING ISSUED, AND ALL OUTSTANDING
;       2728  !       QIOS HAVE COMPLETED.
;       2729  !-
;       2730
;       2731      begin
;       2732
;       2733      MD_INIT ();                                   ! INIT MULTI-DRIVE TEST DATA
;       2734
;       2735      do  begin                                     ! START OF EXECUTIVE LOOP
;       2736
;       2737          incr CTLR from 0 to (MAX_CTLR - 1) do     ! FOR EACH CONTROLLER
;       2738              begin
;       2739              SET_CPAR (.CTLR);                     ! SET UP CURRENT CONTROLLER PARAMETERS
;       2740
;       2741              SETPRI (PRI07);                       ! NO INTERRUPTS WHEN EXAMINING SA
;       2742              ICTLR = .CCTLR;                       ! FAKE INTERRUPTING CONTROLLER'S NUMBER
;       2743              ICST_ADDR = .CST_ADDR;                ! FAKE INTERRUPTING CONTROLLER'S CST ADDRESS
;       2744              IDCT_ADDR = .DCT_ADDR;                ! FAKE INTERRUPTING CONTROLLER'S DCT ADDRESS
;       2745              IRDRX_ADDR = .ICST_ADDR [IP_ADDR];    ! FAKE INTERRUPTING CONTROLLER'S ADDRESS
;       2746              IDCT_ADDR [SA_SAVE] = .IRDRX_ADDR [RCSA, RC_ALL];
;       2747                                                    ! CONTENTS OF THE SA REGISTER
;       2748
;       2749              if BIT_TST (IDCT_ADDR [SA_SAVE], SA_ERR)    ! IF SA SHOWS AN ERROR
;       2750              then
;       2751                  begin
;       2752                  FATAL_ERROR ();                   ! DECLARE FATAL ERROR
;       2753                  SETPRI (PRI00);                   ! LOWER PRIORITY
;       2754                  exitloop;                         ! QUIT
;       2755                  end
;       2756              else
;       2757                  SETPRI (PRI00);                   ! IF NO ERROR, CONTINUE
;       2758
;       2759              if QIO_OK ()                          ! IF O.K. TO ISSUE QIO(S) TO THIS CONTROLLER
;       2760              then
;       2761                  begin                             ! THEN
;       2762                  QIO_GEN ();                       ! GENERATE 1 OR 2 QIOS
;       2763
;       2764                  if (.MX1 geq 0) and               ! IF SUCCESS ON FIRST QIO
```

```
;          2765                    (not .EOP_FLAG)
;          2766                then
;          2767
;          2768                    if SEND (.MX1) eql SUCCESS          ! ATTEMPT TO SEND IT. IF SUCCESS
;          2769                    then
;          2770                        QIO [.CTLR] = .QIO [.CTLR] + 1  ! INCR OUTSTANDING QIO COUNT
;          2771                    else
;          2772                        PUT_PKT (.MX1);                 ! RETURN PACKET TO POOL
;          2773
;          2774                if (.MX2 geq 0) and                     ! IF SUCCESS ON SECOND QIO
;          2775                   (not .EOP_FLAG)
;          2776                then
;          2777
;          2778                    if SEND (.MX2) eql SUCCESS          ! ATTEMPT TO SEND IT. IF SUCCESS
;          2779                    then
;          2780                        QIO [.CTLR] = .QIO [.CTLR] + 1  ! INCR OUTSTANDING QIO COUNT
;          2781                    else
;          2782                        PUT_PKT (.MX2);                 ! RETURN PACKET TO POOL
;          2783
;          2784                end;                                    ! O.K. TO ISSUE QIO(S)
;          2785
;          2786            end;                                        ! CONTROLLER LOOP
;          2787
;          2788        BREAK;                                          ! BREAK FOR SUPERVISOR TO CATCH USER REQUESTS
;          2789
;          2790        if not .EOP_FLAG
;          2791        then
;          2792            PROC_RETPKT ();                             ! PROCESS ANY RETURN PACKETS
;          2793
;          2794        end                                             ! EXECUTIVE PROCESSING LOOP
;          2795
;          2796    until ((not QIO_OUT ()) or .EOP_FLAG);
;          2797
;          2798    end;                                                ! EXERCISER


                                        .SBTTL  MULTI.DRIVE MULTI-DRIVE TEST ROUTINES
000000  004137  000000G         MULTI.DRIVE:
                                        JSR     R1,$SAVE2                       ;                       2714
000004  005746                         TST     -(SP)
000006  004737  000000V                 JSR     PC,MD.INIT                      ;                       2733
000012  005002                 1$:     CLR     R2                              ; CTLR                  2737
000014  010246                 2$:     MOV     R2,-(SP)                        ; CTLR,*                2739
000016  004737  000000G                 JSR     PC,SET.CPAR
000022  012700  000340                 MOV     #340,R0                         ;                       2741
000026  104441                         TRAP    41
000030  013737  000000G 000156'         MOV     CCTLR,ICTLR                     ;                       2742
000036  013737  000000G 000106'         MOV     CST.ADDR,ICST.ADDR              ;                       2743
000044  013737  000000G 000110'         MOV     DCT.ADDR,IDCT.ADDR              ;                       2744
000052  017737  000106' 000000G         MOV     @ICST.ADDR,IRDRX.ADDR           ;                       2745
000060  013701  000110'                 MOV     IDCT.ADDR,R1                    ;                       2746
000064  013700  000000G                 MOV     IRDRX.ADDR,R0
000070  016066  000002  000002         MOV     2(R0),2(SP)                     ; *,RC.REG
```

```
000076  016661  000002  000002          MOV     2(SP),2(R1)             ; RC.REG,*
000104  016601  000002                  MOV     2(SP),R1                ;                               2749
000110  042701  077777                  BIC     #77777,R1
000114  020127  100000                  CMP     R1,#-100000
000120  001006                          BNE     3$
000122  004737  000000V                 JSR     PC,FATAL.ERROR          ;                               2752
000126  005000                          CLR     R0                      ;                               2753
000130  104441                          TRAP    41
000132  005726                          TST     (SP)+                   ;                               2751
000134  000464                          BR      8$
000136  005000                  3$:     CLR     R0                      ;                               2757
000140  104441                          TRAP    41
000142  004737  000000V                 JSR     PC,QIO.OK               ;                               2759
000146  006000                          ROR     R0
000150  103052                          BCC     7$
000152  004737  000000V                 JSR     PC,QIO.GEN              ;                               2762
000156  013700  000160'                 MOV     MX1,R0                  ;                               2764
000162  002421                          BLT     5$
000164  132737  000001  000000G         BITB    #1,EOP.FLAG             ;                               2765
000172  001015                          BNE     5$
000174  010016                          MOV     R0,(SP)                 ;                               2768
000176  004737  000000G                 JSR     PC,SEND
000202  020027  000001                  CMP     R0,#1
000206  001003                          BNE     4$
000210  105262  000000G                 INCB    QIO(R2)                 ; *(CTLR)                       2770
000214  000404                          BR      5$                      ;                               2768
000216  013716  000160'          4$:     MOV     MX1,(SP)                ;                               2772
000222  004737  000000G                 JSR     PC,PUT.PKT
000226  013700  000162'          5$:     MOV     MX2,R0                  ;                               2774
000232  002421                          BLT     7$
000234  132737  000001  000000G         BITB    #1,EOP.FLAG             ;                               2775
000242  001015                          BNE     7$
000244  010016                          MOV     R0,(SP)                 ;                               2778
000246  004737  000000G                 JSR     PC,SEND
000252  020027  000001                  CMP     R0,#1
000256  001003                          BNE     6$
000260  105262  000000G                 INCB    QIO(R2)                 ; *(CTLR)                       2780
000264  000404                          BR      7$                      ;                               2778
000266  013716  000162'          6$:     MOV     MX2,(SP)                ;                               2782
000272  004737  000000G                 JSR     PC,PUT.PKT
000276  005726                  7$:     TST     (SP)+                   ;                               2738
000300  005202                          INC     R2                      ; CTLR                          2737
000302  000243                          .WORD CLV!CLC

000304  003643                          BLE     2$
000306  104422                  8$:     TRAP    22                      ;                               2786
000310  132737  000001  000000G         BITB    #1,EOP.FLAG             ;                               2790
000316  001002                          BNE     9$
000320  004737  000000V                 JSR     PC,PROC.RETPKT          ;                               2792
000324  004737  000000V          9$:     JSR     PC,QIO.OUT              ;                               2796
000330  006000                          ROR     R0
000332  103004                          BCC     10$
000334  132737  000001  000000G         BITB    #1,EOP.FLAG
```

```
000342  001623                          BEQ     1$
000344  005726                  10$:     TST     (SP)+                          ;                                    2714
000346  000207                          RTS     PC

; Routine Size:  116 words,     Routine Base:  $CODE$ + 5344
; Maximum stack depth per invocation:  7 words
```

```
;          2799  routine MD_INIT : novalue =
;          2800
;          2801  !+
;          2802  !       THIS ROUTINE IS CALLED BY ROUTINE MULTI_DRIVE TO INITIALIZE DATA ITEMS
;          2803  !       USED BY THE MULTI-DRIVE TEST.
;          2804  !-
;          2805
;          2806      begin
;          2807
;          2808      if .ENTRY_REASON eql START                                  ! IF THIS IS A START
;          2809      then
;          2810          INIT_IO_BUFF ();                                        ! PARTITION FREE MEMORY INTO I/O BUFFERS
;          2811
;          2812      if (.ENTRY_REASON neq CONT) and                             ! IF START, RESTART, OR PWR FAIL
;          2813         (.ENTRY_REASON neq NEW_PASS)
;          2814      then
;          2815
;          2816          incr CTLR from 0 to (MAX_CTLR - 1) do
;          2817              begin
;          2818              SET_CPAR (.CTLR);
;          2819
;          2820              incr DISK from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;          2821                  begin
;          2822                  SET_UPAR (.DISK);
;          2823                  BST [.L$LUN] = .CST_ADDR [(.DISK + 1), D_BEG];  ! INITIALIZE BLOCK SEQUENCE TABLE
;          2824                  DPST [.L$LUN] = DP_CNT;                         ! INITIALIZE DATA PATTERN SEQUENCE TABLE
;          2825                  end;
;          2826
;          2827              end;
;          2828
;          2829      incr COUNT from 0 to (QIO_PER_CTLR * MAX_CTLR - 1) do       ! INITIALIZE I/O BUFFER ALLOCATION
;          2830          BUFF_OWN [.COUNT] = -1;                                 !     TABLE
;          2831
;          2832      end;                                                        ! ROUTINE MD_INIT


                                        .SBTTL  MD.INIT MULTI-DRIVE TEST ROUTINES
000000  004137  000000G           MD.INIT:JSR     R1,$SAVE4                       ;                       2799
000004  123727  000000G 000001            CMPB    ENTRY.REASON,#1                 ;                       2808
000012  001002                          BNE     1$
000014  004737  000000V                   JSR     PC,INIT.IO.BUFF                 ;                       2810
000020  123727  000000G 000003    1$:     CMPB    ENTRY.REASON,#3                 ;                       2812
000026  001444                          BEQ     4$
000030  123727  000000G 000005            CMPB    ENTRY.REASON,#5                 ;                       2813
000036  001440                          BEQ     4$
000040  005004                          CLR     R4                              ; CTLR                  2816
000042  010446                  2$:     MOV     R4,-(SP)                        ; CTLR,*                2818
000044  004737  000000G                 JSR     PC,SET.CPAR
000050  012703  000003                  MOV     #3,R3                           ; *,DISK                2820
000054  010316                  3$:     MOV     R3,(SP)                         ; DISK,*                2822
000056  004737  000000G                 JSR     PC,SET.UPAR
000062  013702  000000G                 MOV     L$LUN,R2                        ;                       2823
000066  010201                          MOV     R2,R1
```

```
000070  006301                          ASL     R1
000072  010300                          MOV     R3,R0                   ; DISK,*
000074  006300                          ASL     R0
000076  063700  000000G                 ADD     CST.ADDR,R0
000102  016061  000002  000050'         MOV     2(R0),BST(R1)
000110  112762  000025  000060'         MOVB    #25,DPST(R2)            ;                       2824
000116  062703  000005                  ADD     #5,R3                   ; *,DISK                2820
000122  020327  000022                  CMP     R3,#22                  ; DISK,*
000126  003752                          BLE     3$
000130  005726                          TST     (SP)+                   ;                       2817
000132  005204                          INC     R4                      ; CTLR                  2816
000134  000243                          .WORD CLV!CLC

000136  003741                          BLE     2$
000140  005000                  4$:     CLR     R0                      ; COUNT                 2829
000142  112760  000377  000000G   5$:   MOVB    #377,BUFF.OWN(R0)       ; *,*(COUNT)            2830
000150  005200                          INC     R0                      ; COUNT                 2829
000152  020027  000007                  CMP     R0,#7                   ; COUNT,*
000156  003771                          BLE     5$
000160  000207                          RTS     PC                      ;                       2799

; Routine Size:  57 words,      Routine Base:  $CODE$ + 5714
; Maximum stack depth per invocation:  7 words
```

```
;         2833  routine INIT_IO_BUFF : novalue =
;         2834
;         2835  !+
;         2836  !       THIS ROUTINE IS CALLED BY MD_INIT WHEN THE MULTI-DRIVE TEST IS FIRST
;         2837  !       STARTED. IT IS RESPONSIBLE FOR PARTITIONING FREE MEMORY INTO A
;         2838  !       COLLECTION OF I/O BUFFERS. THE SIZE OF EACH I/O BUFFER IS DETERMINED
;         2839  !       BY A NUMBER OF FACTORS, INCLUDING THE NUMBER OF UNITS, THE NUMBER OF
;         2840  !       CONTROLLERS, AND THE SIZE OF FREE MEMORY.
;         2841  !
;         2842  !       ONCE THE BUFFER SIZE IS DETERMINED, THE NUMBER OF I/O BUFFERS IS
;         2843  !       CALCULATED. FINALLY, THE BUFFER ADDRESS (BUFF_ADDR) TABLE IS LOADED
;         2844  !       WITH FIXED BUFFER DESCRIPTORS THAT ARE USED IN THE ALLOCATION AND
;         2845  !       DEALLOCATION PROCESS.
;         2846  !
;         2847  !       IMPLICIT INPUTS:
;         2848  !               CTLR_CNT - THE NUMBER OF CONTROLLERS CONFIGURED
;         2849  !               LSUNIT - THE NUMBER OF UNITS AVAILABLE FOR TESTING
;         2850  !               FREE_MEM_ADDR - START OF FREE MEMORY
;         2851  !-
;         2852
;         2853      begin
;         2854
;         2855      local
;         2856          DRS_START;
;         2857
;         2858      DRS_START = .FREE_MEM_ADDR + 2 + (..FREE_MEM_ADDR * 2);     ! START OF SUPERVISOR
;         2859      BUFF_ADDR [0] = (.FREE_MEM_ADDR + 2 + 1) and %o'177776';    ! START OF READ/WRITE BUFFERS
;         2860
;         2861      while (.BUFF_ADDR [0] and %o'37') neq 0 do                  ! FORCE FIRST I/O BUFFER TO START
;         2862          BUFF_ADDR [0] = .BUFF_ADDR [0] + 2;                     ! ON EVEN BOUNDARY
;         2863
;         2864      BYTS_PER_QIO = ((.DRS_START - .BUFF_ADDR [0]) / (QIO_PER_CTLR * MAX_CTLR)) and %o'177740';
;         2865                                                                  ! MAX TRANSFER SIZE
;         2866
;         2867      if .BYTS_PER_QIO gtru MAX_XFER_SIZE
;         2868      then
;         2869          BYTS_PER_QIO = MAX_XFER_SIZE;                           ! ADJUST TRANSFER SIZE LOWER
;         2870
;         2871      if .BYTS_PER_QIO lssu 32
;         2872      then
;         2873          begin
;         2874          ERRSF (2, EGS_02, 0);                                   ! ERROR IF NOT ENOUGH MEMORY
;         2875          DOCLN;
;         2876          end;
;         2877
;         2878      if (QIO_PER_CTLR * MAX_CTLR) gtru 1
;         2879      then
;         2880
;         2881      incr index from 1 to (QIO_PER_CTLR * MAX_CTLR - 1) do       ! INIT REMAINING TABLE ENTRIES
;         2882          BUFF_ADDR [.index] = .BUFF_ADDR [.index - 1] + .BYTS_PER_QIO;   ! FIXED BUFFER ADDRESS
;         2883
;         2884      end;                                                        ! ROUTINE INIT_IO_BUFF
```

```
                                        .SBTTL  INIT.IO.BUFF MULTI-DRIVE TEST ROUTINES
000000  004137  000000G           INIT.IO.BUFF:
                                        JSR     R1,$SAVE3                       ;                       2833
000004  013701  000000G                 MOV     FREE.MEM.ADDR,R1                ;                       2858
000010  011100                          MOV     (R1),R0
000012  006300                          ASL     R0
000014  060100                          ADD     R1,R0                           ;                       2853
000016  062700  000002                  ADD     #2,R0                           ;                       2858
000022  062701  000003                  ADD     #3,R1                           ;                       2859
000026  010137  000000G                 MOV     R1,BUFF.ADDR
000032  042737  000001  000000G         BIC     #1,BUFF.ADDR
000040  032737  000037  000000G 1$:     BIT     #37,BUFF.ADDR                   ;                       2861
000046  001404                          BEQ     2$
000050  062737  000002  000000G         ADD     #2,BUFF.ADDR                    ;                       2862
000056  000770                          BR      1$                              ;                       2861
000060  010046                  2$:     MOV     R0,-(SP)                        ; DRS.START,*           2864
000062  163716  000000G                 SUB     BUFF.ADDR,(SP)
000066  012746  000010                  MOV     #10,-(SP)
000072  004737  000000G                 JSR     PC,BL$DIV
000076  010037  000000G                 MOV     R0,BYTS.PER.QIO
000102  042737  000037  000000G         BIC     #37,BYTS.PER.QIO
000110  023727  000000G 002000         CMP     BYTS.PER.QIO,#2000              ;                       2867
000116  101403                          BLOS    3$
000120  012737  002000  000000G         MOV     #2000,BYTS.PER.QIO              ;                       2869
000126  023727  000000G 000040 3$:     CMP     BYTS.PER.QIO,#40                ;                       2871
000134  103005                          BHIS    4$
000136  104454                          TRAP    54                              ;                       2874
000140  000002                          .WORD   2
000142  000000G                         .WORD   EGS.02
000144  000000                          .WORD   0
000146  104444                          TRAP    44
000150  012702  000001          4$:     MOV     #1,R2                           ; *,INDEX               2878
000154  010201                  5$:     MOV     R2,R1                           ; INDEX,*               2882
000156  006301                          ASL     R1
000160  010200                          MOV     R2,R0                           ; INDEX,*
000162  006300                          ASL     R0
000164  016003  177776G                 MOV     BUFF.ADDR-2(R0),R3
000170  063703  000000G                 ADD     BYTS.PER.QIO,R3
000174  010361  000000G                 MOV     R3,BUFF.ADDR(R1)
000200  005202                          INC     R2                              ; INDEX                 2878
000202  020227  000007                  CMP     R2,#7                           ; INDEX,*
000206  003762                          BLE     5$
000210  022626                          CMP     (SP)+,(SP)+                     ;                       2853
000212  000207                          RTS     PC                              ;                       2833

; Routine Size:  70 words,      Routine Base:  $CODE$ + 6076
; Maximum stack depth per invocation:  8 words
```

```
;        2885  routine QIO_OK =
;        2886
;        2887  !+
;        2888  !       THIS ROUTINE IS CALLED BY THE MULTI DRIVE "EXECUTIVE" IN ORDER TO
;        2889  !       DETERMINE WHETHER OR NOT A QIO REQUEST (OR QIO PAIR) SHOULD BE
;        2890  !       GENERATED TO THE CURRENT CONTROLLER. A VALUE OF "TRUE" IS RETURNED IF
;        2891  !       THE CONTROLLER MEETS 3 REQUIREMENTS:
;        2892  !
;        2893  !               A. THE CONTROLLER IS ONLINE;
;        2894  !               B. THE NUMBER OF OUTSTANDING QIOS IS AT LEAST 2 LESS THAN THE
;        2895  !                       MAXIMUM ALLOWED FOR ANY ONE CONTROLLER;
;        2896  !               C. THERE IS AT LEAST ONE DISK ONLINE TO THE CONTROLLER.
;        2897  !
;        2898  !       IF ANY OF THESE TEST FAIL, THEN A VALUE OF "FALSE" IS RETURNED.
;        2899  !
;        2900  !       IMPLICIT INPUTS:
;        2901  !               CCTLR - CURRENT CONTROLLER NUMBER
;        2902  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;        2903  !-
;        2904
;        2905      if (.CST_ADDR [STATE] eql ONLINE) and                 ! IF CONTROLLER IS ONLINE
;        2906         (not .EOP_FLAG) and
;        2907         ((.QIO [.CCTLR] + 2) lequ QIO_PER_CTLR) and        ! IF OUTSTANDING QIO COUNT IS O.K.
;        2908         (.CST_ADDR [U_CNT] neq 0)                          ! IF THERE IS VALID UNIT
;        2909      then
;        2910          return TRUE                                       ! "TRUE" EXIT POINT
;        2911      else
;        2912          return FALSE;                                     ! "FALSE" EXIT POINT


                                        .SBTTL  QIO.OK MULTI-DRIVE TEST ROUTINES
000000  013700  000000G                QIO.OK: MOV     CST.ADDR,R0                     ;       2905
000004  005760  000002                          TST     2(R0)
000010  100027                                  BPL     1$
000012  132737  000001  000000G                BITB    #1,EOP.FLAG                     ;       2906
000020  001023                                  BNE     1$
000022  013700  000000G                        MOV     CCTLR,R0                        ;       2907
000026  116000  000000G                        MOVB    QIO(R0),R0
000032  042700  177400                        BIC     #177400,R0
000036  062700  000002                        ADD     #2,R0
000042  020027  000010                        CMP     R0,#10
000046  101010                                BHI     1$
000050  013700  000000G                        MOV     CST.ADDR,R0                     ;       2908
000054  105760  000005                        TSTB    5(R0)
000060  001403                                BEQ     1$
000062  012700  000001                        MOV     #1,R0                           ;       2885
000066  000207                                RTS     PC
000070  005000                        1$:     CLR     R0
000072  000207                                RTS     PC

; Routine Size:  30 words,      Routine Base:  $CODE$ + 6312
; Maximum stack depth per invocation:  0 words
```

```
;         2913  routine QIO_OUT =
;         2914
;         2915  !+
;         2916  !       THIS ROUTINE IS CALLED BY THE MULTI_DRIVE EXECUTIVE FOR DETERMINING THE
;         2917  !       END OF THE MULTI-DRIVE TEST. ITS PURPOSE IS TO EXAMINE THE QIO VECTOR
;         2918  !       FOR ANY OUTSTANDING QIOS ON ANY CONTROLLER. A VALUE OF "TRUE" IS
;         2919  !       RETURNED IF THERE IS AT LEAST ONE QIO OUTSTANDING ON ANY CONTROLLER.
;         2920  !       OTHERWISE, "FALSE" IS RETURNED INDICATING NO OUTSTANDING QIOS.
;         2921  !-
;         2922
;         2923      begin
;         2924
;         2925      incr CTLR from 0 to (MAX_CTLR - 1) do
;         2926          begin
;         2927          SET_CPAR (.CTLR);                       ! SET UP CURRENT CONTROLLER PARAMETERS
;         2928
;         2929          if .CST_ADDR [STATE] eql ONLINE         ! IF CONTROLLER IS ONLINE
;         2930          then
;         2931              return TRUE;
;         2932
;         2933          end;
;         2934
;         2935      return FALSE;                               ! EXIT - NO CONTROLLERS ONLINE
;         2936      end;

                                .SBTTL  QIO.OUT MULTI-DRIVE TEST ROUTINES
000000  010146                  QIO.OUT:MOV     R1,-(SP)                ;                               2913
000002  005001                          CLR     R1                      ; CTLR                          2925
000004  010146                  1$:     MOV     R1,-(SP)                ; CTLR,*                        2927
000006  004737  000000G                 JSR     PC,SET.CPAR
000012  013700  000000G                 MOV     CST.ADDR,R0             ;                               2929
000016  005760  000002                  TST     2(R0)
000022  100004                          BPL     2$
000024  005726                          TST     (SP)+                   ;                               2931
000026  012700  000001                  MOV     #1,R0
000032  000405                          BR      3$
000034  005726                  2$:     TST     (SP)+                   ;                               2926
000036  005201                          INC     R1                      ; CTLR                          2925
000040  000243                          .WORD CLV!CLC

000042  003760                          BLE     1$
000044  005000                          CLR     R0                      ;                               2923
000046  012601                  3$:     MOV     (SP)+,R1                ;                               2913
000050  000207                          RTS     PC

; Routine Size:  21 words,      Routine Base:  $CODE$ + 6406
; Maximum stack depth per invocation:  3 words
```

```
;       2937  routine QIO_GEN : novalue =
;       2938
;       2939  !+
;       2940  !       THIS ROUTINE IS CALLED BY THE MULTI_DRIVE EXECUTIVE FOR AN ONLINE
;       2941  !       CONTROLLER ELIGIBLE TO RECEIVE I/O TRANSFER REQUESTS. IT IS
;       2942  !       RESPONSIBLE FOR SECURING ONE OR TWO MSCP PACKETS AND LOADING THEM
;       2943  !       WITH VARIOUS PARAMETERS COMPRISING THE I/O REQUEST. THE I/O REQUEST
;       2944  !       GENERATED HERE IS DESTINED TO A PARTICULAR UNIT SELECTED AT RANDOM FROM
;       2945  !       THOSE CONFIGURED UNDER THE CURRENT CONTROLLER.
;       2946  !
;       2947  !       EACH FIELD OF THE PACKET(S) IS LOADED WITHIN INDIVIDUAL ROUTINES
;       2948  !       (QIO_FUNC, QIO_LBN, QIO_SIZE, ETC.). MOST OF THE VALUES SELECTED FOR
;       2949  !       EACH FIELD ARE BASED ON A SET OF RANDOM NUMBER GENERATED AT THE START.
;       2950  !
;       2951  !       UNDER NORMAL CIRCUMSTANCES, ONLY ONE I/O REQUEST IS GENERATED. HOWEVER,
;       2952  !       IF THIS I/O REQUEST IS A "WRITE", AND IF THE OPERATOR SELECTED THE
;       2953  !       OPTION FOR HOST WRITE-COMPARES, THEN A SECOND "READ" REQUEST WILL BE
;       2954  !       GENERATED WITH THE SAME LBN AND BYTE COUNT.
;       2955  !
;       2956  !       AFTER THE PACKET(S) HAVE BEEN LOADED, THIS ROUTINE REGAINS CONTROL
;       2957  !       AND ATTEMPTS TO GET ONE OR TWO I/O BUFFERS FOR THE ACTUAL DATA
;       2958  !       TRANSFERS. THE SUCCESS / FAIL STATUS OF THIS ENTIRE OPERATION IS
;       2959  !       PASSED BACK TO THE CALLER THROUGH THE GLOBALS "MX1" AND "MX2"; THEY
;       2960  !       CONTAIN VALID MSCP PACKET INDECES, OR -1.
;       2961  !
;       2962  !       Note that the DUP Exerciser is located inside the QIO_FUNC routine.
;       2963  !       Every so often the Dup exerciser will run and return the MSCP Exerciser
;       2964  !       to it's normal state.
;       2965  !
;       2966  !
;       2967  !       IMPLICIT INPUTS:
;       2968  !               CCTLR - CURRENT CONTROLLER NUMBER
;       2969  !-
;       2970
;       2971      begin
;       2972      MX2 = -1;                                   ! ASSUME FAILURE IN SECURING 2ND PACKET
;       2973
;       2974      if (MX1 = GET_PKT (.CCTLR)) lss 0           ! TRY TO GET 1ST PACKET. IF FAILURE
;       2975      then
;       2976          return;                                 ! NO POINT IN CONTINUING
;       2977
;       2978      if (MX2 = GET_PKT (.CCTLR)) lss 0           ! TRY TO GET 2ND PACKET. IF FAILURE
;       2979      then
;       2980          begin
;       2981          PUT_PKT (.MX1);                         ! RETURN 1ST PACKET TO POOL
;       2982          MX1 = -1;                               ! INDICATE FAILURE
;       2983          return;                                 ! DONE
;       2984          end;
;       2985
;       2986      MAD1 = MSCP_PKT + (.MX1 * PKT_LEN * 2);     ! CALCULATE STARTING ADDRESSES
;       2987      MAD2 = MSCP_PKT + (.MX2 * PKT_LEN * 2);     !     OF BOTH PACKETS
;       2988      GET_RANDOM ();                              ! GENERATE A SET OF RANDOM NUMBERS
;       2989      QIO_UNIT ();                                ! LOAD RANDOM UNIT NUMBER INTO PACKETS
```

```
;
;
;
```

```
                                        .SBTTL  QIO.GEN MULTI-DRIVE TEST ROUTINES
000000  012737  177777  000162'   QIO.GEN:MOV   #-1,MX2                          ;                                       2972
000006  013746  000000G           QIO.GEN:MOV   CCTLR,-(SP)                      ;                                       2974
000012  004737  000000G                   JSR   PC,GET.PKT
000016  010037  000160'                   MOV   R0,MX1
000022  005726                            TST   (SP)+
000024  005700                            TST   R0                               ; MX1
000026  002572                            BLT   6$                               ;                                       2976
000030  013746  000000G                   MOV   CCTLR,-(SP)                      ;                                       2978
000034  004737  000000G                   JSR   PC,GET.PKT
000040  010037  000162'                   MOV   R0,MX2
000044  005726                            TST   (SP)+
000046  005700                            TST   R0                               ; MX2
000050  002011                            BGE   1$
000052  013746  000160'                   MOV   MX1,-(SP)                        ;                                       2981
000056  004737  000000G                   JSR   PC,PUT.PKT
000062  012737  177777  000160'           MOV   #-1,MX1                          ;                                       2982
000070  005726                            TST   (SP)+                            ;                                       2978
000072  000207                            RTS   PC                               ;                                       2980
000074  013746  000160'           1$:     MOV   MX1,-(SP)                        ;                                       2986
000100  012746  000104                    MOV   #104,-(SP)
000104  004737  000000G                   JSR   PC,BL$MUL
000110  062700  000000G                   ADD   #MSCP.PKT,R0
000114  010037  000164'                   MOV   R0,MAD1
000120  013716  000162'                   MOV   MX2,(SP)                         ;                                       2987
000124  012746  000104                    MOV   #104,-(SP)
000130  004737  000000G                   JSR   PC,BL$MUL
000134  062700  000000G                   ADD   #MSCP.PKT,R0
000140  010037  000166'                   MOV   R0,MAD2
000144  004737  000000V                   JSR   PC,GET.RANDOM                    ;                                       2988
000150  004737  000000V                   JSR   PC,QIO.UNIT                      ;                                       2989
000154  132737  000001  000000G         BITB  #1,EOP.FLAG                      ;                                       2991
000162  001112                            BNE   5$                               ;                                       2937
000164  004737  000000V                   JSR   PC,QIO.FUNC                      ;                                       2995
000170  005737  000160'                   TST   MX1                              ;                                       2999
000174  002505                            BLT   5$
000176  132737  000001  000000G         BITB  #1,EOP.FLAG
000204  001101                            BNE   5$                               ;                                       2937
000206  004737  000000V                   JSR   PC,QIO.LBN                       ;                                       3002
000212  004737  000000V                   JSR   PC,QIO.SIZE                      ;                                       3003
000216  013716  000164'                   MOV   MAD1,(SP)                        ;                                       3004
000222  062716  000032                    ADD   #32,(SP)
000226  004737  000000G                   JSR   PC,GET.IO.BUFF
000232  005737  000162'                   TST   MX2                              ;                                       3006
000236  002437                            BLT   3$
000240  013716  000166'                   MOV   MAD2,(SP)                        ;                                       3009
000244  062716  000032                    ADD   #32,(SP)
000250  004737  000000G                   JSR   PC,GET.IO.BUFF
000254  013700  000166'                   MOV   MAD2,R0                          ;                                       3011
000260  005760  000032                    TST   32(R0)
000264  001024                            BNE   3$
000266  013700  000164'                   MOV   MAD1,R0                          ;                                       3015
000272  062700  000032                    ADD   #32,R0
```

```
000276  005710                          TST     (R0)
000300  001407                          BEQ     2$
000302  010016                          MOV     R0,(SP)                 ;                       3018
000304  004737  000000G                 JSR     PC,PUT.IO.BUFF
000310  013700  000164'                 MOV     MAD1,R0                 ;                       3019
000314  005060  000032                  CLR     32(R0)
000320  013716  000162'          2$:    MOV     MX2,(SP)                ;                       3022
000324  004737  000000G                 JSR     PC,PUT.PKT
000330  012737  177777  000162'        MOV     #-1,MX2                 ;                       3023
000336  013700  000164'          3$:    MOV     MAD1,R0                 ;                       3028
000342  005760  000032                  TST     32(R0)
000346  001010                          BNE     4$
000350  013716  000160'                 MOV     MX1,(SP)                ;                       3031
000354  004737  000000G                 JSR     PC,PUT.PKT
000360  012737  177777  000160'        MOV     #-1,MX1                 ;                       3032
000366  000410                          BR      5$                      ;                       3028
000370  013700  000164'          4$:    MOV     MAD1,R0                 ;                       3036
000374  126027  000022  000042         CMPB    22(R0),#42
000402  001002                          BNE     5$
000404  004737  000000V                 JSR     PC,FILL.BUFF            ;                       3038
000410  062706  000006          5$:    ADD     #6,SP                   ;                       2971
000414  000207                  6$:    RTS     PC                      ;                       2937

; Routine Size:  135 words,     Routine Base:  $CODE$ + 6460
; Maximum stack depth per invocation:  4 words
```

```
;         3041  routine GET_RANDOM : novalue =
;         3042
;         3043  !+
;         3044  !       THIS ROUTINE IS CALLED BY QIO_GEN TO GENERATE A SET OF RANDOM NUMBERS,
;         3045  !       AND TO STORE THEM INTO THE RANDOM NUMBER TABLE (RANDOM). THE RANDOM
;         3046  !       NUMBERS ARE USED TO SELECT I/O REQUEST PARAMETERS FOR THE CURRENT QIO
;         3047  !       OR QIO PAIR. IN ADDITION, IF DATA PATTERN #1 IS BEING USED, THESE
;         3048  !       RANDOM NUMBERS WILL BE USED IN THE WRITE OPERATION.
;         3049  !-
;         3050
;         3051      begin
;         3052
;         3053      own
;         3054          SEED : word initial (173),
;         3055          NEXT_RANDNUM : word initial (245);
;         3056
;         3057      incr COUNT from 0 to (RDM_LEN - 1) do
;         3058          begin
;         3059          SEED = (.SEED + .NEXT_RANDNUM + 1) * 4;
;         3060          NEXT_RANDNUM = (.NEXT_RANDNUM / 4) + .SEED;
;         3061          RANDOM [.COUNT] = .NEXT_RANDNUM;
;         3062          end;
;         3063
;         3064      end;


001241                                  .PSECT  $GGG$,  RO
                                        .EVEN
001242  000255                  SEED:   .WORD   255
001244  000365                  NEXT.RANDNUM:
                                        .WORD   365


                                        .SBTTL  GET.RANDOM MULTI-DRIVE TEST ROUTINES
007076                                  .PSECT  $CODE$,  RO

000000  004137  000000G          GET.RANDOM:
                                        JSR     R1,$SAVE3                       ;               3041
000004  013702  001242'                 MOV     SEED,R2                         ;               3059
000010  013703  001244'                 MOV     NEXT.RANDNUM,R3
000014  005001                          CLR     R1                              ; COUNT         3057
000016  010300                  1$:     MOV     R3,R0                           ;               3059
000020  060200                          ADD     R2,R0
000022  006300                          ASL     R0                              ;               3058
000024  006300                          ASL     R0
000026  010037  001242'                 MOV     R0,SEED                         ;               3059
000032  062737  000004  001242'         ADD     #4,SEED
000040  010346                          MOV     R3,-(SP)                        ;               3060
000042  012746  000004                  MOV     #4,-(SP)
000046  004737  000000G                 JSR     PC,BL$DIV
000052  013702  001242'                 MOV     SEED,R2
000056  060200                          ADD     R2,R0
```

```
000060  010037  001244'                    MOV     R0,NEXT.RANDNUM
000064  010003                             MOV     R0,R3                   ; NEXT.RANDNUM,*                3061
000066  010361  000116'                    MOV     R3,RANDOM(R1)           ; *,*(COUNT)
000072  022626                             CMP     (SP)+,(SP)+             ;                               3058
000074  062701  000002                     ADD     #2,R1                   ; *,COUNT                       3057
000100  020127  000036                     CMP     R1,#36                  ; COUNT,*
000104  003744                             BLE     1$
000106  000207                             RTS     PC                      ;                               3041

; Routine Size:  36 words,      Routine Base:  $CODE$ + 7076
; Maximum stack depth per invocation:  7 words
```

```
;       3065  routine QIO_UNIT : novalue =
;       3066
;       3067  !+
;       3068  !       THIS ROUTINE IS CALLED BY QIO_GEN TO RANDOMLY SELECT ONE UNIT
;       3069  !       CONFIGURED UNDER THE CURRENT CONTROLLER (CCTLR) TO BE USED FOR THE
;       3070  !       CURRENT QIO OR QIO PAIR. THE UNIT SELECTED IS BASED ON THE NUMBER OF
;       3071  !       UNITS ELIGIBLE TO RECEIVE AN I/O REQUEST (FROM 1 TO 4) AND THE FIRST
;       3072  !       RANDOM NUMBER IN THE RANDOM NUMBER TABLE (RANDOM).
;       3073  !
;       3074  !       IMPLICIT INPUTS:
;       3075  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;       3076  !
;       3077  !       IMPLICIT OUTPUTS:
;       3078  !               THE RD/RX DISK NUMBER (DISK ADDRESS) IS LOADED INTO THE
;       3079  !                       APPROPRIATE FIELD OF BOTH MSCP PACKETS.
;       3080  !-
;       3081
;       3082      begin
;       3083      own
;       3084          RAT_COUNT : word initial (0);
;       3085      local
;       3086          MOD_COUNT : byte,
;       3087          TBL_COUNT : byte,
;       3088          SELECT_RD : byte;
;       3089
;       3090  !
;       3091  ! THE UNITS WILL BE SELECTED ON AN ADJUSTABLE RATIO, RD51 TO RX50,
;       3092  ! SELECTED VIA THE SOFTWARE PARAMETERS
;       3093  !
;       3094  ! THIS MODE IS FOR SELECTING DEVICES ON THE FOLLOWING SCHEME:
;       3095  ! CHOOSE A DEVICE AND KEEP IT SELECTED FOR A CONSTANT TIME, THEN
;       3096  ! MOVE TO THE NEXT. THIS IS NON-RANDOM, FIXED SEQUENTIAL OPERATIONAL
;       3097  ! MODE
;       3098  !
;       3099
;       3100      if ((.SWP_FLAGS and SWF_RDM) neq SWF_RDM) and       ! NOT RANDOM MODE
;       3101         ((.SWP_FLAGS and SWF_SEQ) neq SWF_SEQ)           ! NOT RANDOM SEQUEUNTIAL MODE
;       3102      then
;       3103
;       3104          if (.BST_CNT neq 0) and
;       3105             (.CST_ADDR [.BST_DEV, D_PRES] eql PRESENT) and
;       3106             (.CST_ADDR [.BST_DEV, D_STAT] eql ONLINE) and
;       3107             (not .CST_ADDR [.BST_DEV, D_FATAL])
;       3108          then
;       3109              begin                                       ! ALREADY WITHIN DEVICE
;       3110              BST_CNT = .BST_CNT - 1;
;       3111              SET_UPAR (.BST_DEV);
;       3112              MAD1 [DK_NUM] = .CDISK;
;       3113              MAD2 [DK_NUM] = .CDISK;
;       3114              return;
;       3115              end
;       3116          else
;       3117              begin                                       ! GET NEW DEVICE
```

```
;       3118
;       3119            incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;       3120                begin
;       3121
;       3122                if (.BST_DEV eql 0) or
;       3123                   (.BST_DEV eql (3 * UNIT_SIZE + OF_UN))
;       3124                then
;       3125                    BST_DEV = OF_UN
;       3126                else
;       3127                    BST_DEV = .BST_DEV + UNIT_SIZE;
;       3128
;       3129                if (.CST_ADDR [.BST_DEV, D_PRES] eql PRESENT) and
;       3130                   (.CST_ADDR [.BST_DEV, D_STAT] eql ONLINE) and
;       3131                   (not .CST_ADDR [.BST_DEV, D_FATAL])
;       3132                then
;       3133                    begin
;       3134
;       3135                    if .CST_ADDR [.BST_DEV, D_TYPE] eql RX_50
;       3136                    then
;       3137                        BST_CNT = RX_SEQ_CNT
;       3138                    else
;       3139                        BST_CNT = RD_SEQ_CNT;
;       3140
;       3141                    SET_UPAR (.BST_DEV);
;       3142                    MAD1 [DK_NUM] = .CDISK;
;       3143                    MAD2 [DK_NUM] = .CDISK;
;       3144                    return;
;       3145                    end;
;       3146
;       3147                end;
;       3148
;       3149            end;
;       3150
;       3151    !
;       3152    ! RANDOM SELECTION OF DRIVES
;       3153    !
;       3154    !       this part selects the device by the ratio
;       3155
;       3156    rat_count = .rat_count + 1;                         ! increment counter from 0 to 100
;       3157    if (.rat_count geq 100) then rat_count = 0;         ! in case counter gets to large reinit to 0
;       3158    if (.rat_count lss .swp_rat)
;       3159    then
;       3160        SELECT_RD = true                                ! if counter greater than swap ratio then do a rx-50
;       3161    else                                                ! and reiniate the counter
;       3162        SELECT_RD = false;                              ! if counter less than ratio do a rd-51
;       3163
;       3164    if (100 eql .swp_rat)                               ! if ratio equal 100 do rd
;       3165    then
;       3166        SELECT_RD = true;
;       3167
;       3168    if (0 eql .swp_rat)                                 ! if ratio equals 0 do rx
;       3169    then
;       3170        SELECT_RD = FALSE;
```

```
3171
3172
3173    !
3174    ! IF RD51s SELECTED
3175    !
3176    ! COUNT NUMBER OF RD51s AVAILABLE
3177    !
3178
3179        if .SELECT_RD
3180        then
3181            begin
3182            MOD_COUNT = 0;                                  ! COUNT THE NUMBER OF RDs IN THE SYSTEM
3183
3184            incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
3185
3186                if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and
3187                   (.CST_ADDR [.OFFSET, D_STAT] eql ONLINE) and
3188                   (.CST_ADDR [.OFFSET, D_TYPE] eql RD_51) and
3189                   (not .CST_ADDR [.OFFSET, D_FATAL])
3190                then
3191                    begin
3192                    STORAGE [.MOD_COUNT] = .OFFSET;
3193                    MOD_COUNT = .MOD_COUNT + 1;
3194                    end;
3195
3196    !
3197    ! SELECT ON OF THE RD51s
3198    !
3199
3200            if .MOD_COUNT neq 0                             ! IF AT LEAST ON RD51 PRESENT
3201            then
3202                begin
3203                TBL_COUNT = 0;
3204
3205                do
3206                    begin
3207                    SET_UPAR (.STORAGE [(.RANDOM [.TBL_COUNT] and %o'077777') mod .MOD_COUNT]);
3208                    TBL_COUNT = .TBL_COUNT + 1;
3209                    end
3210                until ((.CST_ADDR [.CUOFF, D_PRES] eql PRESENT) and
3211                       (.CST_ADDR [.CUOFF, D_STAT] eql ONLINE) and
3212                       (not .CST_ADDR [.CUOFF, D_FATAL])) or
3213                      (.TBL_COUNT eql RDM_LEN);
3214
3215                MAD1 [DK_NUM] = .CDISK;
3216                MAD2 [DK_NUM] = .CDISK;
3217                return;
3218                end;
3219
3220            end;
3221
3222    !
3223    ! IF NO RD51 SELECTED, SELECT AN RX50
```

```
3224  !
3225  ! COUNT THE NUMBER OF RX50s
3226  !
3227
3228      MOD_COUNT = 0;
3229
3230      incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
3231
3232          if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and
3233             (.CST_ADDR [.OFFSET, D_STAT] eql ONLINE) and
3234             (.CST_ADDR [.OFFSET, D_TYPE] eql RX_50) and
3235             (not .CST_ADDR [.OFFSET, D_FATAL])
3236          then
3237              begin
3238              STORAGE [.MOD_COUNT] = .OFFSET;
3239              MOD_COUNT = .MOD_COUNT + 1;
3240              end;
3241
3242  !
3243  ! AND CHOOSE ONE!
3244  !
3245
3246      if .MOD_COUNT neq 0
3247      then
3248          begin
3249          TBL_COUNT = 0;
3250
3251          do
3252              begin
3253              SET_UPAR (.STORAGE [(.RANDOM [.TBL_COUNT] and %o'077777') mod .MOD_COUNT]);
3254              TBL_COUNT = .TBL_COUNT + 1;
3255              end
3256          until ((.CST_ADDR [.CUOFF, D_PRES] eql PRESENT) and
3257                 (.CST_ADDR [.CUOFF, D_STAT] eql ONLINE) and
3258                 (not .CST_ADDR [.CUOFF, D_FATAL])) or
3259                (.TBL_COUNT eql RDM_LEN);
3260
3261          MAD1 [DK_NUM] = .CDISK;
3262          MAD2 [DK_NUM] = .CDISK;
3263          return;
3264          end;
3265
3266  !
3267  ! IF NO UNIT SELECTED SO FAR BY ABOVE METHOD, SELECT ANY ONE AT RANDOM
3268  !
3269  ! COUNT ALL UNITS AVAILABLE
3270  !
3271
3272      MOD_COUNT = 0;
3273
3274      incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
3275
3276          if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and
```

```
                3277            (.CST_ADDR [.OFFSET, D_STAT] eql ONLINE) and
                3278            (not .CST_ADDR [.OFFSET, D_FATAL])
                3279        then
                3280            begin
                3281            STORAGE [.MOD_COUNT] = .OFFSET;
                3282            MOD_COUNT = .MOD_COUNT + 1;
                3283            end;
                3284
                3285    !
                3286    ! SELECT ANY ONE ONE UNIT AT RANDOM
                3287    !
                3288        if .MOD_COUNT neq 0
                3289        then
                3290            begin
                3291            TBL_COUNT = 0;
                3292
                3293            do
                3294                begin
                3295                SET_UPAR (.STORAGE [(.RANDOM [.TBL_COUNT] and %o'077777') mod .MOD_COUNT]);
                3296                TBL_COUNT = .TBL_COUNT + 1;
                3297                end
                3298            until ((.CST_ADDR [.CUOFF, D_PRES] eql PRESENT) and
                3299                   (.CST_ADDR [.CUOFF, D_STAT] eql ONLINE) and
                3300                   (not .CST_ADDR [.CUOFF, D_FATAL])) or
                3301                  (.TBL_COUNT eql RDM_LEN);
                3302
                3303            MAD1 [DK_NUM] = .CDISK;
                3304            MAD2 [DK_NUM] = .CDISK;
                3305            return
                3306            end
                3307
                3308    !
                3309    ! DECLARE END-OF-PASS IF NO UNIT ONLINE
                3310    !
                3311
                3312        else
                3313            begin
                3314            EOP_FLAG = TRUE;
                3315            DCT_ADDR [IG_INT] = TRUE;
                3316            end;
                3317
                3318        end;                                        ! ROUTINE QIO_UNIT


001246                                      .PSECT  $GGG$,  RO
001246  000000                      RAT.COUNT:
                                            .WORD   0


                                            .SBTTL  QIO.UNIT MULTI-DRIVE TEST ROUTINES
007206                                      .PSECT  $CODE$,  RO
```

```
000000  004137  000000G                QIO.UNIT:
                                                JSR     R1,$SAVE3                       ;                               3065
000004  032737  000002  000000G                 BIT     #2,SWP.FLAGS                    ;                               3100
000012  001136                                  BNE     9$
000014  032737  001000  000000G                 BIT     #1000,SWP.FLAGS                 ;                               3101
000022  001132                                  BNE     9$
000024  005737  001230'                        TST     BST.CNT                         ;                               3104
000030  001433                                  BEQ     1$
000032  013700  001232'                        MOV     BST.DEV,R0                      ;                               3105
000036  006300                                  ASL     R0
000040  063700  000000G                        ADD     CST.ADDR,R0
000044  032710  040000                        BIT     #40000,(R0)
000050  001423                                  BEQ     1$
000052  013700  001232'                        MOV     BST.DEV,R0                      ;                               3106
000056  006300                                  ASL     R0
000060  063700  000000G                        ADD     CST.ADDR,R0
000064  032710  020000                        BIT     #20000,(R0)
000070  001413                                  BEQ     1$
000072  013700  001232'                        MOV     BST.DEV,R0                      ;                               3107
000076  006300                                  ASL     R0
000100  063700  000000G                        ADD     CST.ADDR,R0
000104  032710  010000                        BIT     #10000,(R0)
000110  001003                                  BNE     1$
000112  005337  001230'                        DEC     BST.CNT                         ;                               3110
000116  000447                                  BR      7$                              ;                               3111
000120  012701  000003            1$:         MOV     #3,R1                           ; *,OFFSET                      3119
000124  013700  001232'            2$:         MOV     BST.DEV,R0                      ;                               3122
000130  001403                                  BEQ     3$
000132  020027  000022                        CMP     R0,#22                          ;                               3123
000136  001004                                  BNE     4$
000140  012737  000003  001232'    3$:         MOV     #3,BST.DEV                      ;                               3125
000146  000403                                  BR      5$                              ;                               3122
000150  062737  000005  001232'    4$:         ADD     #5,BST.DEV                      ;                               3127
000156  013700  001232'            5$:         MOV     BST.DEV,R0                      ;                               3129
000162  006300                                  ASL     R0
000164  063700  000000G                        ADD     CST.ADDR,R0
000170  032710  040000                        BIT     #40000,(R0)
000174  001440                                  BEQ     8$
000176  032710  020000                        BIT     #20000,(R0)                     ;                               3130
000202  001435                                  BEQ     8$
000204  032710  010000                        BIT     #10000,(R0)                     ;                               3131
000210  001032                                  BNE     8$
000212  132710  000004                        BITB    #4,(R0)                         ;                               3135
000216  001004                                  BNE     6$
000220  012737  000120  001230'                MOV     #120,BST.CNT                    ;                               3137
000226  000403                                  BR      7$                              ;                               3135
000230  012737  004160  001230'    6$:         MOV     #4160,BST.CNT                   ;                               3139
000236  013746  001232'            7$:         MOV     BST.DEV,-(SP)                   ;                               3141
000242  004737  000000G                        JSR     PC,SET.UPAR
000246  013700  000164'                        MOV     MAD1,R0                         ;                               3142
000252  013760  000000G  000016                MOV     CDISK,16(R0)
000260  013700  000166'                        MOV     MAD2,R0                         ;                               3143
000264  013760  000000G  000016                MOV     CDISK,16(R0)
```

```
000272  005726                          TST     (SP)+                   ;                               3129
000274  000207                          RTS     PC                      ;                               3133
000276  062701  000005          8$:     ADD     #5,R1                   ; *,OFFSET                      3119
000302  020127  000022                  CMP     R1,#22                  ; OFFSET,*
000306  003706                          BLE     2$
000310  005237  001246'          9$:     INC     RAT.COUNT               ;                               3156
000314  023727  001246' 000144          CMP     RAT.COUNT,#144          ;                               3157
000322  002402                          BLT     10$
000324  005037  001246'                  CLR     RAT.COUNT
000330  013701  000000G         10$:     MOV     SWP.RAT,R1              ;                               3158
000334  023701  001246'                  CMP     RAT.COUNT,R1
000340  002003                          BGE     11$
000342  112700  000001                  MOVB    #1,R0                   ; *,SELECT.RD                   3160
000346  000401                          BR      12$                     ;                               3158
000350  105000                  11$:    CLRB    R0                      ; SELECT.RD                     3162
000352  020127  000144          12$:    CMP     R1,#144                 ;                               3164
000356  001002                          BNE     13$
000360  112700  000001                  MOVB    #1,R0                   ; *,SELECT.RD                   3166
000364  005701                  13$:    TST     R1                      ;                               3168
000366  001001                          BNE     14$
000370  105000                          CLRB    R0                      ; SELECT.RD                     3170
000372  006000                  14$:    ROR     R0                      ; SELECT.RD                     3179
000374  103105                          BCC     19$
000376  105003                          CLRB    R3                      ; MOD.COUNT                     3182
000400  012701  000003                  MOV     #3,R1                   ; *,OFFSET                      3184
000404  010100                  15$:    MOV     R1,R0                   ; OFFSET,*                      3186
000406  006300                          ASL     R0
000410  063700  000000G                 ADD     CST.ADDR,R0
000414  032710  040000                  BIT     #40000,(R0)
000420  001417                          BEQ     16$
000422  032710  020000                  BIT     #20000,(R0)             ;                               3187
000426  001414                          BEQ     16$
000430  132710  000004                  BITB    #4,(R0)                 ;                               3188
000434  001411                          BEQ     16$
000436  032710  010000                  BIT     #10000,(R0)             ;                               3189
000442  001006                          BNE     16$
000444  005000                          CLR     R0                      ;                               3192
000446  150300                          BISB    R3,R0                   ; MOD.COUNT,*
000450  006300                          ASL     R0
000452  010160  000074'                 MOV     R1,STORAGE(R0)          ; OFFSET,*
000456  105203                          INCB    R3                      ; MOD.COUNT                     3193
000460  062701  000005          16$:    ADD     #5,R1                   ; *,OFFSET                      3184
000464  020127  000022                  CMP     R1,#22                  ; OFFSET,*
000470  003745                          BLE     15$
000472  105703                          TSTB    R3                      ; MOD.COUNT                     3200
000474  001445                          BEQ     19$
000476  105002                          CLRB    R2                      ; TBL.COUNT                     3203
000500  005000                  17$:    CLR     R0                      ;                               3207
000502  150200                          BISB    R2,R0                   ; TBL.COUNT,*
000504  006300                          ASL     R0
000506  016046  000116'                 MOV     RANDOM(R0),-(SP)
000512  042716  100000                  BIC     #100000,(SP)
000516  005046                          CLR     -(SP)
```

```
000520  110316                          MOVB    R3,(SP)                 ; MOD.COUNT,*
000522  004737  000000G                 JSR     PC,BL$MOD
000526  006300                          ASL     R0
000530  016016  000074'                 MOV     STORAGE(R0),(SP)
000534  004737  000000G                 JSR     PC,SET.UPAR
000540  105202                          INCB    R2                      ; TBL.COUNT                     3208
000542  022626                          CMP     (SP)+,(SP)+             ;                               3206
000544  013700  000000G                 MOV     CUOFF,R0                ;                               3210
000550  006300                          ASL     R0
000552  063700  000000G                 ADD     CST.ADDR,R0
000556  032710  040000                  BIT     #40000,(R0)
000562  001406                          BEQ     18$
000564  032710  020000                  BIT     #20000,(R0)             ;                               3211
000570  001403                          BEQ     18$
000572  032710  010000                  BIT     #10000,(R0)             ;                               3212
000576  001510                          BEQ     24$
000600  120227  000020          18$:    CMPB    R2,#20                  ; TBL.COUNT,*                   3213
000604  001335                          BNE     17$
000606  000504                          BR      24$                     ;                               3215
000610  105003                  19$:    CLRB    R3                      ; MOD.COUNT                     3228
000612  012701  000003                  MOV     #3,R1                   ; *,OFFSET                      3230
000616  010100                  20$:    MOV     R1,R0                   ; OFFSET,*                      3232
000620  006300                          ASL     R0
000622  063700  000000G                 ADD     CST.ADDR,R0
000626  032710  040000                  BIT     #40000,(R0)
000632  001417                          BEQ     21$
000634  032710  020000                  BIT     #20000,(R0)             ;                               3233
000640  001414                          BEQ     21$
000642  132710  000004                  BITB    #4,(R0)                 ;                               3234
000646  001011                          BNE     21$
000650  032710  010000                  BIT     #10000,(R0)             ;                               3235
000654  001006                          BNE     21$
000656  005000                          CLR     R0                      ;                               3238
000660  150300                          BISB    R3,R0                   ; MOD.COUNT,*
000662  006300                          ASL     R0
000664  010160  000074'                 MOV     R1,STORAGE(R0)          ; OFFSET,*
000670  105203                          INCB    R3                      ; MOD.COUNT                     3239
000672  062701  000005          21$:    ADD     #5,R1                   ; *,OFFSET                      3230
000676  020127  000022                  CMP     R1,#22                  ; OFFSET,*
000702  003745                          BLE     20$
000704  105703                          TSTB    R3                      ; MOD.COUNT                     3246
000706  001445                          BEQ     25$
000710  105002                          CLRB    R2                      ; TBL.COUNT                     3249
000712  005000                  22$:    CLR     R0                      ;                               3253
000714  150200                          BISB    R2,R0                   ; TBL.COUNT,*
000716  006300                          ASL     R0
000720  016046  000116'                 MOV     RANDOM(R0),-(SP)
000724  042716  100000                  BIC     #100000,(SP)
000730  005046                          CLR     -(SP)
000732  110316                          MOVB    R3,(SP)                 ; MOD.COUNT,*
000734  004737  000000G                 JSR     PC,BL$MOD
000740  006300                          ASL     R0
000742  016016  000074'                 MOV     STORAGE(R0),(SP)
```

```
000746  004737  000000G                        JSR     PC,SET.UPAR
000752  105202                                  INCB    R2              ; TBL.COUNT                     3254
000754  022626                                  CMP     (SP)+,(SP)+     ;                               3252
000756  013700  000000G                        MOV     CUOFF,R0        ;                               3256
000762  006300                                  ASL     R0
000764  063700  000000G                        ADD     CST.ADDR,R0
000770  032710  040000                        BIT     #40000,(R0)
000774  001406                                  BEQ     23$
000776  032710  020000                        BIT     #20000,(R0)     ;                               3257
001002  001403                                  BEQ     23$
001004  032710  010000                        BIT     #10000,(R0)     ;                               3258
001010  001505                                  BEQ     30$
001012  120227  000020          23$:    CMPB    R2,#20          ; TBL.COUNT,*                   3259
001016  001335                                  BNE     22$
001020  000501                          24$:    BR      30$             ;                               3261
001022  105003                          25$:    CLRB    R3              ; MOD.COUNT                     3272
001024  012701  000003                        MOV     #3,R1           ; *,OFFSET                      3274
001030  010100                          26$:    MOV     R1,R0           ; OFFSET,*                      3276
001032  006300                                  ASL     R0
001034  063700  000000G                        ADD     CST.ADDR,R0
001040  032710  040000                        BIT     #40000,(R0)
001044  001414                                  BEQ     27$
001046  032710  020000                        BIT     #20000,(R0)     ;                               3277
001052  001411                                  BEQ     27$
001054  032710  010000                        BIT     #10000,(R0)     ;                               3278
001060  001006                                  BNE     27$
001062  005000                                  CLR     R0              ;                               3281
001064  150300                                  BISB    R3,R0           ; MOD.COUNT,*
001066  006300                                  ASL     R0
001070  010160  000074'                        MOV     R1,STORAGE(R0)  ; OFFSET,*
001074  105203                                  INCB    R3              ; MOD.COUNT                     3282
001076  062701  000005          27$:    ADD     #5,R1           ; *,OFFSET                      3274
001102  020127  000022                        CMP     R1,#22          ; OFFSET,*
001106  003750                                  BLE     26$
001110  105703                                  TSTB    R3              ; MOD.COUNT                     3288
001112  001457                                  BEQ     31$
001114  105002                                  CLRB    R2              ; TBL.COUNT                     3291
001116  005000                          28$:    CLR     R0              ;                               3295
001120  150200                                  BISB    R2,R0           ; TBL.COUNT,*
001122  006300                                  ASL     R0
001124  016046  000116'                        MOV     RANDOM(R0),-(SP)
001130  042716  100000                        BIC     #100000,(SP)
001134  005046                                  CLR     -(SP)
001136  110316                                  MOVB    R3,(SP)         ; MOD.COUNT,*
001140  004737  000000G                        JSR     PC,BL$MOD
001144  006300                                  ASL     R0
001146  016016  000074'                        MOV     STORAGE(R0),(SP)
001152  004737  000000G                        JSR     PC,SET.UPAR
001156  105202                                  INCB    R2              ; TBL.COUNT                     3296
001160  022626                                  CMP     (SP)+,(SP)+     ;                               3294
001162  013700  000000G                        MOV     CUOFF,R0        ;                               3298
001166  006300                                  ASL     R0
001170  063700  000000G                        ADD     CST.ADDR,R0
```

```
001174  032710  040000                          BIT     #40000,(R0)
001200  001406                                  BEQ     29$
001202  032710  020000                          BIT     #20000,(R0)             ;                               3299
001206  001403                                  BEQ     29$
001210  032710  010000                          BIT     #10000,(R0)             ;                               3300
001214  001403                                  BEQ     30$
001216  120227  000020                  29$:    CMPB    R2,#20                  ; TBL.COUNT,*                   3301
001222  001335                                  BNE     28$
001224  013700  000164'                  30$:    MOV     MAD1,R0                 ;                               3303
001230  013760  000000G 000016                  MOV     CDISK,16(R0)
001236  013700  000166'                          MOV     MAD2,R0                 ;                               3304
001242  013760  000000G 000016                  MOV     CDISK,16(R0)
001250  000207                                  RTS     PC                      ;                               3290
001252  112737  000001  000000G         31$:    MOVB    #1,EOP.FLAG             ;                               3314
001260  052777  040000  000000G                 BIS     #40000,@DCT.ADDR        ;                               3315
001266  000207                                  RTS     PC                      ;                               3065

; Routine Size:  348 words,     Routine Base:  $CODE$ + 7206
; Maximum stack depth per invocation:  7 words
```

```
;        3319  routine QIO_FUNC : novalue =
;        3320
;        3321  !+
;        3322  !       THIS ROUTINE IS CALLED BY QIO_GEN TO SELECT THE I/O FUNCTION (OPCODE)
;        3323  !       TO BE USED FOR THE CURRENT QIO OR QIO PAIR. THE FUNCTION IS DETERMINED
;        3324  !       BY THE FOLLWING ALGORITHM:
;        3325  !
;        3326  !               IF THE CHOSEN UNIT IS PROTECTED
;        3327  !               THEN
;        3328  !                       FUNCTION = READ
;        3329  !               ELSE (UNPROTECTED)
;        3330  !                       FUNCTION (WRITE OR READ) IS BASED ON A RANDOM
;        3331  !                       NUMBER
;        3332  !
;        3333  !       IN ADDITION, IF THE OPERATOR SELECTED THE OPTION OF PERFORMING WRITE-
;        3334  !       COMPARES AT THE HOST, AND IF A "WRITE" FUNCTION WAS CHOSEN ABOVE FOR
;        3335  !       THE FIRST QIO, THEN A "READ" OPCODE IS LOADED INTO THE SECOND MSCP
;        3336  !       ENVELOPE. OTHERWISE, THE SECOND MSCP ENVELOPE IS RETURNED TO THE POOL.
;        3337  !
;        3338  !       THIS ROUTINE ALSO DECIDES WHEATHER IT IS TIME TO RUN THE DUP EXERCISER.
;        3339  !       THE EXERCISER WRITES 25 LBNS FOR EVERY 1 DBN. FOR INITIALIZATION
;        3340  !       REASONS THE MULTIPLE OF 25 LBNS CAN BE READ FOR 1 * MULTIPLE OF DBNS.
;        3341  !       DUP EXERCISER WRITES X BLOCKS PER PASS THRU THE DUP ROUTINE.  THERFORE
;        3342  !       THE RATIO OF 1 TO 25 MUST BE MULTIPLIED BY  X FOR A RATIO OF 1 TO
;        3343  !       25 * X.  SO EVERY 25*X LBN'S READ OFF A WINCHESTER UNIT X  DBN'S WILL
;        3344  !       BE READ IF ASKED BY THE USER.  X is represented by the VARIABLE "DUPROUND".
;        3345  !       DUPROUND MAY BE CHANGED IN THE SOFTWARE QUESTION GIVEN AT THE START OF
;        3346  !       RUNNING THE EXERCISER PROGARM.  THE DUP EXERCISER THEN REINITIALIZES
;        3347  !       THE CONTROLLER AND CONTINUES AS IF THE REGULAR MSCP EXERCISER WAS
;        3348  !       NEVER INTERRUPTED.  NOTE THAT THE REINITIALIZATION PROCESSES TAKES
;        3349  !       A COUPLE OF SECONDS WHICH ONCE ADDED UP AFTER A COUPLE MILLION READS
;        3350  !       CAN PROVE TO BE QUITE TIMELY.  THERFORE I SUGGEST THAT IF A LARGE AMOUNT OF
;        3351  !       I/O TRANFERS ARE TO BE DONE THE "DUPROUND" VARIABLE BE RAISED TO SPEED UP
;        3352  !       UP THE PROCESS.
;        3353  !
;        3354  !       IMPLICIT INPUTS:
;        3355  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;        3356  !               CUOFF - CURRENT UNIT CST OFFSET
;        3357  !
;        3358  !       IMPLICIT OUTPUTS:
;        3359  !               THE OPCODE FIELD OF ONE OR BOTH MSCP ENVELOPES IS LOADED.
;        3360  !-
;        3361
;        3362      begin
;        3363
;        3364      local
;        3365          FUNC : word;                                ! OPCODE (READ OR WRITE)
;        3366  !PRINTX (DER9);
;        3367  !PRINTF (DER9);
;        3368  !^
;        3369      IF ((.CST_ADDR [.CUOFF + 4,D_COUNT] LEQ 0) AND              ! IF MSCP FUNC CNT EQUAL TO 0
;        3370          (.CST_ADDR [.CUOFF,D_TYPE] EQL RD_51) AND               !IF WINCHESTER DISK
;        3371          (.CST_ADDR [.CUOFF + 3, NODUPMEDIA] NEQ 1))             !IF NODUPMEDIA FLAG BIT IS CLEAR
```

```
3372            THEN
3373        BEGIN
3374
3375                    PUT_PKT (.MX2);                             ! RETURN 1ST ENVELOPE TO POOL
3376                    MX2 = -1;                                       ! INDICATE FAILURE
3377                    CST_ADDR [.CUOFF + 3, DUPERROR] = 0;            !CLEAR DUP ERROR FLAG
3378                    DUP ();                                     ! DO DUP TEST
3379                    CST_ADDR [.CUOFF + 3, DUPERROR] = 0;            !CLEAR DUP ERROR FLAG
3380                    !PRINTX (DBM111);                           ! PRINT MSCP EXERCISER REINIT
3381                    CST_ADDR [.CUOFF + 4, D_COUNT]  = (25 * .duproun);     ! REINITIALIZE MSCP FUNC COUNTER
3382
3383        ! ********************************* THIS SECTION REINTIALIZES 2 ENVOLOPES SO THE MSCP EXERCISER CAN
3384        ! ********************************* PROCEED AS BEFORE THE DUP EXERCISER STARTED ********************
3385
3386        DUP_FLAGS = .DUP_FLAGS OR SWP_DINT;                     ! SET DUP INIT FLAG
3387        INIT_TEST ();                                           ! THIS REINITIALIZES THE CONTROLLER FOR MSCP MODE
3388        DUP_FLAGS = .DUP_FLAGS AND (NOT SWP_DINT);              ! CLEAR DUP INIT FLAG
3389
3390
3391        MX2 = -1;                                               ! ASSUME FAILURE IN SECURING 2ND ENVELOPE
3392        IF (MX1 = GET_PKT (.CCTLR)) LSS 0                       ! TRY TO GET 1ST ENVELOPE.
3393            OR (.EOP_FLAG)                                      ! IF FAILURE
3394        THEN    RETURN;                                         ! NO POINT IN CONTINUING
3395        IF (MX2 = GET_PKT (.CCTLR)) LSS 0                       ! TRY TO GET 2ND ENVELOPE.
3396            OR (.EOP_FLAG)                                      ! IF FAILURE
3397        THEN    BEGIN
3398                PUT_PKT (.MX1);                                 ! RETURN 1ST ENVELOPE TO POOL
3399                MX1 = -1;                                           ! INDICATE FAILURE
3400                RETURN;                                         ! DONE
3401                END;
3402
3403        MAD1 = MSCP_PKT + (.MX1 * PKT_LEN * 2);                 ! CALCULATE STARTING ADDRESSES
3404        MAD2 = MSCP_PKT + (.MX2 * PKT_LEN * 2);                 !     OF BOTH ENVELOPES
3405        GET_RANDOM ();                                          ! GENERATE A SET OF RANDOM NUMBERS
3406        QIO_UNIT ();                                            ! LOAD RANDOM UNIT NUMBER INTO ENVELOPES
3407
3408        END;
3409
3410        !*******************************************************************************
3411        !                           START OF ROUTINE MSCP
3412        !*******************************************************************************
3413
3414        CST_ADDR [.CUOFF + 4, D_COUNT]  = .CST_ADDR [.CUOFF + 4, D_COUNT]  - 1;                 ! DECREMENT MSCP FUN
3415
3416
3417            MAD2 [OPCODE] = 0;                                  ! ASSUME 2ND PACKET NOT NEEDED
3418
3419            if .CST_ADDR [.CUOFF, D_PROT] eql PROTECTED         ! IF UNIT IS PROTECTED
3420            then
3421                FUNC = OP_RD                                    ! SET FUNCTION TO READ
3422            else
3423
3424                if (.RANDOM [1] and 1)                          ! USE 2ND RANDOM NUMBER TO SELECT
```

```
;         3425            then
;         3426                FUNC = OP_RD                    !     READ
;         3427            else
;         3428                FUNC = OP_WRT;                  !     WRITE
;         3429
;         3430        if (MAD1 [OPCODE] = .FUNC) eql OP_WRT   ! LOAD CHOSEN OPCODE. IF WRITE
;         3431        then
;         3432            begin
;         3433            MAD1 [CMD_TYPE] = NON_SEQ_CMD;      ! NON-SEQUENTIAL COMMAND
;         3434
;         3435            if BIT_TST (SWP_FLAGS, SWF_CWC)     ! IF CONTROLLER DOES WRITE-COMPARES
;         3436            then
;         3437                MAD1 [MODIFY] = MD_CMP;         ! ADD COMPARE MODIFIER
;         3438
;         3439            if BIT_TST (SWP_FLAGS, SWF_HWC)     ! IF HOST DOES WRITE-COMPARES
;         3440            then
;         3441                begin
;         3442                MAD2 [OPCODE] = OP_RD;          ! SET READ OPCODE INTO 2ND MSCP PACKET
;         3443                MAD2 [CMD_TYPE] = NON_SEQ_CMD;  ! NON-SEQUENTIAL COMMAND
;         3444                end;
;         3445
;         3446            end
;         3447        else
;         3448            begin
;         3449            MAD1 [CMD_TYPE] = NON_SEQ_CMD;      ! NON-SEQUENTIAL COMMAND
;         3450
;         3451            if BIT_TST (SWP_FLAGS, SWF_CRC)     ! IF CONTROLLER DOES READ-COMPARES - FUNCTION IS READ
;         3452            then
;         3453                MAD1 [MODIFY] = MD_CMP;         ! ADD COMPARE MODIFIER
;         3454
;         3455            end;
;         3456
;         3457        if .MAD2 [OPCODE] eql 0                 ! IF NO OPCODE IN 2ND PACKET
;         3458        then
;         3459            begin
;         3460            PUT_PKT (.MX2);                     ! RETURN 2ND PACKET TO POOL
;         3461            MX2 = -1;                           ! MARK IT UNUSED
;         3462            end;
;         3463
;         3464        end;                                    ! ROUTINE QIO_FUNC


                                        .SBTTL  QIO.FUNC MULTI-DRIVE TEST ROUTINES
000000  004137  000000G          QIO.FUNC:
                                        JSR     R1,$SAVE3                       ;       3319
000004  013702  000000G                 MOV     CST.ADDR,R2                     ;       3369
000010  013701  000000G                 MOV     CUOFF,R1
000014  010100                          MOV     R1,R0
000016  006300                          ASL     R0
000020  060200                          ADD     R2,R0
000022  005760  000010                  TST     10(R0)
000026  003166                          BGT     4$
000030  010100                          MOV     R1,R0                           ;       3370
```

```
000032  006300                          ASL     R0
000034  060200                          ADD     R2,R0
000036  132710  000004                  BITB    #4,(R0)
000042  001560                          BEQ     4$
000044  010100                          MOV     R1,R0                   ;                               3371
000046  006300                          ASL     R0
000050  060200                          ADD     R2,R0
000052  005760  000006                  TST     6(R0)
000056  100552                          BMI     4$
000060  013746  000162'                 MOV     MX2,-(SP)               ;                               3375
000064  004737  000000G                 JSR     PC,PUT.PKT
000070  012737  177777  000162'         MOV     #-1,MX2                 ;                               3376
000076  013700  000000G                 MOV     CUOFF,R0                ;                               3377
000102  006300                          ASL     R0
000104  063700  000000G                 ADD     CST.ADDR,R0
000110  042760  040000  000006         BIC     #40000,6(R0)
000116  004737  000000V                 JSR     PC,DUP                  ;                               3378
000122  013700  000000G                 MOV     CUOFF,R0                ;                               3379
000126  006300                          ASL     R0
000130  063700  000000G                 ADD     CST.ADDR,R0
000134  042760  040000  000006         BIC     #40000,6(R0)
000142  013701  000000G                 MOV     CUOFF,R1                ;                               3381
000146  006301                          ASL     R1
000150  063701  000000G                 ADD     CST.ADDR,R1
000154  013716  000000G                 MOV     DUPROUND,(SP)
000160  012746  000031                  MOV     #31,-(SP)
000164  004737  000000G                 JSR     PC,BL$MUL
000170  010061  000010                  MOV     R0,10(R1)
000174  052737  000002  000000G         BIS     #2,DUP.FLAGS            ;                               3386
000202  004737  000212'                 JSR     PC,INIT.TEST            ;                               3387
000206  042737  000002  000000G         BIC     #2,DUP.FLAGS            ;                               3388
000214  012737  177777  000162'         MOV     #-1,MX2                 ;                               3391
000222  013716  000000G                 MOV     CCTLR,(SP)              ;                               3392
000226  004737  000000G                 JSR     PC,GET.PKT
000232  010037  000160'                 MOV     R0,MX1
000236  002426                          BLT     2$
000240  132737  000001  000000G         BITB    #1,EOP.FLAG             ;                               3393
000246  001022                          BNE     2$                      ;                               3319
000250  013716  000000G                 MOV     CCTLR,(SP)              ;                               3395
000254  004737  000000G                 JSR     PC,GET.PKT
000260  010037  000162'                 MOV     R0,MX2
000264  002404                          BLT     1$
000266  132737  000001  000000G         BITB    #1,EOP.FLAG             ;                               3396
000274  001411                          BEQ     3$
000276  013716  000160'         1$:     MOV     MX1,(SP)                ;                               3398
000302  004737  000000G                 JSR     PC,PUT.PKT
000306  012737  177777  000160'         MOV     #-1,MX1                 ;                               3399
000314  022626                  2$:     CMP     (SP)+,(SP)+             ;                               3395
000316  000207                          RTS     PC                      ;                               3397
000320  013716  000160'         3$:     MOV     MX1,(SP)                ;                               3403
000324  012746  000104                  MOV     #104,-(SP)
000330  004737  000000G                 JSR     PC,BL$MUL
000334  062700  000000G                 ADD     #MSCP.PKT,R0
```

```
000340  010037  000164'                        MOV     R0,MAD1
000344  013716  000162'                        MOV     MX2,(SP)                ;                                       3404
000350  012746  000104                          MOV     #104,-(SP)
000354  004737  000000G                         JSR     PC,BL$MUL
000360  062700  000000G                         ADD     #MSCP.PKT,R0
000364  010037  000166'                        MOV     R0,MAD2
000370  004737  007076'                        JSR     PC,GET.RANDOM           ;                                       3405
000374  004737  007206'                        JSR     PC,QIO.UNIT             ;                                       3406
000400  062706  000010                          ADD     #10,SP                  ;                                       3373
000404  013700  000000G               4$:      MOV     CUOFF,R0                ;                                       3414
000410  006300                                  ASL     R0
000412  063700  000000G                         ADD     CST.ADDR,R0
000416  005360  000010                          DEC     10(R0)
000422  013700  000166'                        MOV     MAD2,R0                 ;                                       3417
000426  012703  000022                          MOV     #22,R3
000432  060003                                  ADD     R0,R3
000434  105013                                  CLRB    (R3)
000436  013700  000000G                         MOV     CUOFF,R0                ;                                       3419
000442  006300                                  ASL     R0
000444  063700  000000G                         ADD     CST.ADDR,R0
000450  032710  100000                          BIT     #100000,(R0)
000454  001404                                  BEQ     5$                      ;                                       3421
000456  032737  000001  000120'                 BIT     #1,RANDOM+2             ;                                       3424
000464  001403                                  BEQ     6$
000466  012701  000041               5$:      MOV     #41,R1                  ; *,FUNC                                3426
000472  000402                                  BR      7$                      ;                                       3424
000474  012701  000042               6$:      MOV     #42,R1                  ; *,FUNC                                3428
000500  013700  000164'               7$:      MOV     MAD1,R0                 ;                                       3430
000504  013702  000000G                         MOV     SWP.FLAGS,R2            ;                                       3435
000510  110160  000022                          MOVB    R1,22(R0)               ; FUNC,*                                3430
000514  020127  000042                          CMP     R1,#42                  ; FUNC,*
000520  001024                                  BNE     9$
000522  012760  000002  000004                 MOV     #2,4(R0)                ;                                       3433
000530  032702  000020                          BIT     #20,R2                  ;                                       3435
000534  001403                                  BEQ     8$
000536  012760  040000  000024                 MOV     #40000,24(R0)           ;                                       3437
000544  032702  000040               8$:      BIT     #40,R2                  ;                                       3439
000550  001421                                  BEQ     10$
000552  112713  000041                          MOVB    #41,(R3)                ;                                       3442
000556  013701  000166'                        MOV     MAD2,R1                 ;                                       3443
000562  012761  000002  000004                 MOV     #2,4(R1)
000570  000411                                  BR      10$                     ;                                       3430
000572  012760  000002  000004               9$:      MOV     #2,4(R0)                ;                               3449
000600  032702  000004                          BIT     #4,R2                   ;                                       3451
000604  001403                                  BEQ     10$
000606  012760  040000  000024                 MOV     #40000,24(R0)           ;                                       3453
000614  105713                              10$:      TSTB    (R3)                    ;                               3457
000616  001010                                  BNE     11$
000620  013746  000162'                        MOV     MX2,-(SP)               ;                                       3460
000624  004737  000000G                         JSR     PC,PUT.PKT
000630  012737  177777  000162'                 MOV     #-1,MX2                 ;                                       3461
000636  005726                                  TST     (SP)+                   ;                                       3459
000640  000207                              11$:      RTS     PC                      ;                               3319
```

```
; Routine Size:  209 words,     Routine Base:  $CODE$ + 10476
; Maximum stack depth per invocation:  9 words
```

```
;       3465  !^
;       3466  ROUTINE DUP : NOVALUE =
;       3467
;       3468
;       3469  !+
;       3470  !       THIS ROUTINE IS CALLED BY QIO_FUNC AFTER 25 * "DUPROUND" RD/WTS .
;       3471  !       THIS EXERCISER WAS PLACED IN THE MIDDLE OF THE MSCP EXERCISER SO A
;       3472  !       COMMON INITIALIZATION AND OTHER ROUTINES COULD BE USED.
;       3473  !               THE DUP EXCERSIZER WILL RUN A READ ONLY OR A WRITE/READ COMPARE
;       3474  !       TO THE DIAGONOSTIC BLOCKS OR DBN'S.  IT WILL RECORD
;       3475  !       THE STATICS IN THE TALLY TABLES.
;       3476  !
;       3477  !       THE PROGRAM USES CONTROLLER LOCAL PROGRAMS TO WRITE AND READ THE DBN'S.
;       3478  !       WHEN WRITTING TO THE DBN'S A ONE WORD PATTERN WILL BE SELECTED
;       3479  !       AND COPY TO A 256 WORD BLOCK. THE ROUTINE WILL WRITE TO "DUPROUND" AMOUNT
;       3480  !       OF DBN SEQUENTIAL BLOCKS.  IF A THERE ARE CONTROLLER LOCAL  PROGRAMS AND
;       3481  !       IF THE USER SO DESIRES A WRITE AND  READ OF EACH BLOCK AND A COMPARISON
;       3482  !       TO THE DATA PATTERN WILL BE GIVEN TO TEST FOR FAULTY DBN'S.  IF THE USER
;       3483  !       DOES NOT WANT TO WRITE TO THE DBN'S ONLY A READ WILL BE GIVEN AND NO
;       3484  !       COMPARISON WILL BE DONE.  THE BAD BLOCKS FOUND IN THE COMPARISON TEST
;       3485  !       WILL NOT BE LISTED IN THE RCT TABLES.
;       3486  !
;       3487  !       AFTER THE EXERCISER HAS EXAMINE "DUPROUND" AMOUNT OF  BLOCKS IT WILL
;       3488  !       REINITIATE THE ENVOLOPES SO THAT THE MSCP EXERCISER MAY CONTINUE AS
;       3489  !       BEFORE.
;       3490  !
;       3491  !       IMPLICIT INPUTS:
;       3492  !               CCTLR - CURRENT CONTROLLER NUMBER
;       3493  !               CST_ADDRS  - CONTAINS THE CURRENT CONTROLLER STATUS TABLE
;       3494  !               CUOFF      - CURRENT OFFSET IN CST TABLE FOR PARTICULAR DRIVE
;       3495  !
;       3496  !       IMPLICIT OUTPUTS:
;       3497  !               S_PATTERN - PATTERN BEING WRITTEN TO AND READ FROM DBN'S
;       3498  !-
;       3499  BEGIN
;       3500  OWN
;       3501          TEMP : WORD;
;       3502  !PRINTX (DBM110);
;       3503  !PRINTX (DER10);
;       3504
;       3505  S_PATTERN = .RANDOM [1];
;       3506  IF (.CST_ADDR [.CUOFF + 3, D_DBN] + .duproun) GEQ 144 THEN (CST_ADDR [.CUOFF + 3, D_DBN] = 0);
;       3507                                                  ! TEST TO SEE IF NEXT DBN'S TO LARGE
;       3508                                                  ! CIRCLE ARROUND IF DBN TO LARGE
;       3509  DO
;       3510          BEGIN
;       3511          MSCP_PKT [.MX1, MSGLEN] = SZ_GDS;               ! PACKET SIZE                    GET DUST STATUS
;       3512          MSCP_PKT [.MX1, OPCODE] = OP_GDS;               ! OPCODE = GET DUST STATUS
;       3513          MSCP_PKT [.MX1, MODIFY] = 0;                    !
;       3514          DUPCOMMAND ();                                  ! SENDS AND RECEIVES THE COMMAND
;       3515          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1
;       3516                  THEN
;       3517                   RETURN;                                !IF DUP ERROR THEN CLR FLAG & RETURN
```

```
;         3518          IF .CST_ADDR [.CUOFF + 3, NODUPMEDIA] EQL 1 THEN RETURN; ! IF DUP LOCAL MEDIA NOT THERE THEN RETURN
;         3519          IF (MX1 = GET_PKT (.CCTLR)) LSS 0                        ! TRY TO GET AN ENVELOPE. IF FAILURE
;         3520                OR (.EOP_FLAG)
;         3521                THEN RETURN;                                       ! NO POINT IN CONTINUING
;         3522          END
;         3523    UNTIL .CST_ADDR [.CUOFF + 3, D_ACTIVE] EQL IDLE;
;         3524
;         3525
;         3526    TEMP = .CST_ADDR [.CUOFF + 3, D_DBN];
;         3527    INCR DBNCNT FROM (.TEMP + 1) TO (.TEMP + .dupround) DO              ! INCREMENT FROM RELATIVE DBN TO DBN + dupro
;         3528          BEGIN
;         3529
;         3530          IF .CST_ADDR [.CUOFF + 3, DUPWRITE]                        ! IF WRITE FLAG SET IN CST TABLE THEN WRITE DBN'S
;         3531           THEN
;         3532                BEGIN
;         3533                T_ADDR [T_DBN_WT] = .T_ADDR [T_DBN_WT] + 1;
;         3534                DUPWRTDBN();                                         ! CALL ROUTINE TO HANDLE WRITTING ROUTINES
;         3535                IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1
;         3536                     THEN
;         3537                      RETURN;                                        !IF DUP ERROR THEN CLR FLAG & RETURN
;         3538                END;
;         3539          T_ADDR [T_DBN_RD] = .T_ADDR [T_DBN_RD] + 1;
;         3540          DUPREDDBN();                                             ! CALL ROUTINE TO HANDLE READING DBN'S
;         3541          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1
;         3542                THEN
;         3543                 RETURN;                                           !IF DUP ERROR THEN CLR FLAG & RETURN
;         3544
;         3545          END;
;         3546
;         3547    END;


001250                                          .PSECT  $GGGG$,  RO
001250                                  TEMP:   .BLKW   1


                                                .SBTTL  DUP MULTI-DRIVE TEST ROUTINES
011340                                          .PSECT  $CODE$,  RO

000000  004137  000000G                 DUP:    JSR     R1,$SAVE3                ;                                  3466
000004  013737  000120'  000000G               MOV     RANDOM+2,S.PATTERN       ;                                  3505
000012  013700  000000G                         MOV     CUOFF,R0                 ;                                  3506
000016  006300                                  ASL     R0
000020  063700  000000G                         ADD     CST.ADDR,R0
000024  005001                                  CLR     R1
000026  156001  000006                          BISB    6(R0),R1
000032  063701  000000G                         ADD     DUPROUND,R1
000036  020127  000220                          CMP     R1,#220
000042  002402                                  BLT     1$
000044  105060  000006                          CLRB    6(R0)
000050  013746  000160'                 1$:     MOV     MX1,-(SP)                ;                                  3511
000054  012746  000104                         MOV     #104,-(SP)
```

```
000060  004737  000000G                        JSR     PC,BL$MUL
000064  012760  000014  000006G                MOV     #14,MSCP.PKT+6(R0)
000072  112760  000001  000022G                MOVB    #1,MSCP.PKT+22(R0)          ;3512
000100  005060  000024G                        CLR     MSCP.PKT+24(R0)             ;3513
000104  004737  000000V                        JSR     PC,DUPCOMMAND               ;3514
000110  013700  000000G                        MOV     CUOFF,R0                    ;3515
000114  006300                                ASL     R0
000116  063700  000000G                        ADD     CST.ADDR,R0
000122  032760  040000  000006                BIT     #40000,6(R0)
000130  001016                                BNE     2$                          ;3466
000132  005760  000006                        TST     6(R0)                       ;3518
000136  100413                                BMI     2$                          ;3466
000140  013716  000000G                        MOV     CCTLR,(SP)                  ;3519
000144  004737  000000G                        JSR     PC,GET.PKT
000150  010037  000160'                        MOV     R0,MX1
000154  002404                                BLT     2$
000156  132737  000001  000000G                BITB    #1,EOP.FLAG                 ;3520
000164  001402                                BEQ     3$
000166  022626                        2$:     CMP     (SP)+,(SP)+                 ;3466
000170  000207                                RTS     PC                          ;3521
000172  022626                        3$:     CMP     (SP)+,(SP)+                 ;3510
000174  013700  000000G                        MOV     CUOFF,R0                    ;3523
000200  006300                                ASL     R0
000202  063700  000000G                        ADD     CST.ADDR,R0
000206  062700  000006                        ADD     #6,R0
000212  032710  020000                        BIT     #20000,(R0)
000216  001314                                BNE     1$
000220  111037  001250'                        MOVB    (R0),TEMP                   ;3526
000224  105037  001251'                        CLRB    TEMP+1
000230  013702  001250'                        MOV     TEMP,R2                     ;3527
000234  063702  000000G                        ADD     DUPROUND,R2
000240  013700  000000G                        MOV     CUOFF,R0                    ;3530
000244  006300                                ASL     R0
000246  063700  000000G                        ADD     CST.ADDR,R0
000252  010003                                MOV     R0,R3
000254  062703  000006                        ADD     #6,R3
000260  013701  001250'                        MOV     TEMP,R1                     ;*,DBNCNT 3527
000264  000443                                BR      6$
000266  032713  010000                4$:     BIT     #10000,(R3)                 ;3530
000272  001417                                BEQ     5$
000274  013700  000000G                        MOV     T.ADDR,R0                   ;3533
000300  005260  000052                        INC     52(R0)
000304  004737  000000V                        JSR     PC,DUPWRTDBN                ;3534
000310  013700  000000G                        MOV     CUOFF,R0                    ;3535
000314  006300                                ASL     R0
000316  063700  000000G                        ADD     CST.ADDR,R0
000322  032760  040000  000006                BIT     #40000,6(R0)
000330  001024                                BNE     7$                          ;3537
000332  013700  000000G                5$:     MOV     T.ADDR,R0                   ;3539
000336  005260  000056                        INC     56(R0)
000342  004737  000000V                        JSR     PC,DUPREDDBN                ;3540
000346  013700  000000G                        MOV     CUOFF,R0                    ;3541
000352  006300                                ASL     R0
```

```
000354  063700  000000G                        ADD     CST.ADDR,R0
000360  010003                                 MOV     R0,R3
000362  062703  000006                         ADD     #6,R3
000366  032713  040000                         BIT     #40000,(R3)
000372  001003                                 BNE     7$                      ;                               3543
000374  005201                         6$:     INC     R1                      ; DBNCNT                        3527
000376  020102                                 CMP     R1,R2                   ; DBNCNT,*
000400  003732                                 BLE     4$
000402  000207                         7$:     RTS     PC                      ;                               3466

; Routine Size:  130 words,     Routine Base:  $CODE$ + 11340
; Maximum stack depth per invocation:  7 words


;        3548
;        3549
```

```
;       3550  ROUTINE DUPWRTDBN : NOVALUE =
;       3551
;       3552  !+
;       3553  ! THIS ROUTINE IS CALLED BY DUP ROUTINE TO USE THE CONTROLLER LOCAL PROGRAM
;       3554  ! "WRTDBN".  TO USE THE PROGRAM THE OPTIONAL DUP SUB-PROTOCAL IS USED TO
;       3555  ! COMMUNICATE WITH THE CONTROLLER.  THE PROGRAM WRITES TO A DIAGONOSTIC BLOCK (DBN)
;       3556  ! THE WORD IN "S_PATTERN" IS WRITTEN TO THE 256 WORDS IN THE DBN.  IF AN ERROR OCCURS
;       3557  ! WHILE RUNNING THE CONTROLLER LOCAL PROGRAM THE ERROR IS USSUALLY REPORTED IN THE
;       3558  ! DUP BUFFER. (EX. ILLEGAL UNIT NUMBER, ILLEGAL BLK #, DEVICE ERROR, ZERO LENGHT MSG)
;       3559  !
;       3560  !       IMPLICIT INPUTS:
;       3561  !               CST_ADDRS  - CONTAINS THE CURRENT CONTROLLER STATUS TABLE
;       3562  !               CUOFF      - CURRENT OFFSET IN CST TABLE FOR PARTICULAR DRIVE
;       3563  !               S_PATTERN  - CONTAINS PATTERN WORD!-
;       3564  BEGIN
;       3565
;       3566  !PRINTX (DER11);
;       3567
;       3568
;       3569  MSCP_PKT [.MX1, MSGLEN] = SZ_ELP;                  ! PACKET SIZE                          EXECUTE LOCAL PROGRAM WRT DB
;       3570  MSCP_PKT [.MX1, OPCODE] = OP_ELP;                  ! OPCODE = EXECUTE LOCAL PROGRAM
;       3571  MSCP_PKT [.MX1, MODIFY] = 1;                       ! STANDALONE MODIFIER
;       3572  MSCP_PKT [.MX1, L1] = %ascii'W';                   ! FILL IN PROGRAM NAME WITH ASCII LETTERS
;       3573  MSCP_PKT [.MX1, L2] = %ascii'R';
;       3574  MSCP_PKT [.MX1, L3] = %ascii'T';
;       3575  MSCP_PKT [.MX1, L4] = %ascii'D';
;       3576  MSCP_PKT [.MX1, L5] = %ascii'B';
;       3577  MSCP_PKT [.MX1, L6] = %ascii'N';
;       3578  DUPCOMMAND ();                                     ! SENDS AND RECEIVES THE COMMAND
;       3579  IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN    RETURN; !IF DUP ERROR THEN  RETURN
;       3580  IF (MX1 = GET_PKT (.CCTLR)) LSS 0                 ! TRY TO GET AN ENVELOPE. IF FAILURE
;       3581          OR (.EOP_FLAG)
;       3582          THEN
;       3583                  (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;       3584                   RETURN;);                                 ! NO POINT IN CONTINUING
;       3585
;       3586
;       3587
;       3588          MSCP_PKT [.MX1, MSGLEN] = SZ_REC;          ! PACKET SIZE                          RECIEVE DATA
;       3589          MSCP_PKT [.MX1, OPCODE] = OP_RCD;          ! OPCODE = RECEIVE DATA
;       3590          MSCP_PKT [.MX1, BC_LO] = 2;                ! BYTE COUNT TO BE TRANSFERED EQUALS 2
;       3591          MSCP_PKT [.MX1, BUF_0] = DUPPKT;           ! LOAD DESCRIBTOR BUFFER
;       3592          MSCP_PKT [.MX1, MODIFY] = 0;               !
;       3593          DUPCOMMAND ();                             ! SENDS AND RECEIVES THE COMMAND
;       3594          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
;       3595          IF .DUPPKT [DUPTYPE]  NEQ 1
;       3596             OR .DUPPKT [DUPMESG] NEQ 6              ! IF CORRECT RESPONSE
;       3597             OR (MX1 = GET_PKT (.CCTLR)) LSS 0       ! TRY TO GET AN ENVELOPE. IF FAILURE
;       3598             OR (.EOP_FLAG)
;       3599             THEN
;       3600                  (HARD_ERROR ();
;       3601                   CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;       3602                   RETURN;);                                 ! NO POINT IN CONTINUING
```

```
;          3603
;          3604
;          3605          MSCP_PKT [.MX1, MSGLEN] = SZ_SEN;                ! PACKET SIZE                                    SEND DATA
;          3606          MSCP_PKT [.MX1, OPCODE] = OP_SDD;                ! OPCODE = SEND DATA
;          3607          MSCP_PKT [.MX1, BC_LO] = 6;                      ! BYTE COUNT TO BE TRANSFERED EQUALS 6
;          3608          MSCP_PKT [.MX1, BUF_0] = DUPPKT;                 ! LOAD DESCRIBTOR BUFFER
;          3609          DUPPKT [DUPBF0] = .Cdisk;                        !LOAD UNIT NUMBER (RDRX)
;          3610          DUPPKT [DUPBF1] = .CST_ADDR [.CUOFF + 3, D_DBN]; ! LOAD DBN NUMBER
;          3611          DUPPKT [DUPBF2] = .S_PATTERN;                    ! LOAD PATTERN
;          3612          MSCP_PKT [.MX1, MODIFY] = 0;                     !
;          3613          DUPCOMMAND ();                                   ! SENDS AND RECEIVES THE COMMAND
;          3614          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
;          3615          IF (MX1 = GET_PKT (.CCTLR)) LSS 0                ! TRY TO GET AN ENVELOPE. IF FAILURE
;          3616                OR (.EOP_FLAG)
;          3617                THEN
;          3618                (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;          3619                 RETURN;);                                                ! NO POINT IN CONTINUING
;          3620
;          3621
;          3622          MSCP_PKT [.MX1, MSGLEN] = SZ_REC;                ! PACKET SIZE                                    RECEIVE DATA
;          3623          MSCP_PKT [.MX1, OPCODE] = OP_RCD;                ! OPCODE = RECEIVE DATA
;          3624          MSCP_PKT [.MX1, BC_LO] = 4;                      ! BYTE COUNT TO BE TRANSFERED EQUALS 4
;          3625          MSCP_PKT [.MX1, BUF_0] = DUPPKT;                 ! LOAD DESCRIBTOR BUFFER
;          3626          MSCP_PKT [.MX1, MODIFY] = 0;                     !
;          3627          DUPCOMMAND ();                                   ! SENDS AND RECEIVES THE COMMAND
;          3628          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
;          3629          IF (.DUPPKT [DUPTYPE]  NEQ 3)
;          3630             OR (.DUPPKT [DUPMESG] NEQ 3)                  ! IF CORRECT RESPONSE
;          3631              OR (.DUPPKT [DUPBF1] NEQ 0)
;          3632              THEN
;          3633                  (HARD_ERROR ();
;          3634                   CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;          3635                   RETURN;);                                              ! NO POINT IN CONTINUING
;          3636
;          3637          IF ((MX1 = GET_PKT (.CCTLR)) LSS 0)              ! TRY TO GET AN ENVELOPE. IF FAILURE
;          3638             OR (.EOP_FLAG)
;          3639             THEN
;          3640                 (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;          3641                  RETURN;);                                              ! NO POINT IN CONTINUING
;          3642
;          3643  T_ADDR [T_BLK_WT] = .T_ADDR [T_BLK_WT] + 1;
;          3644  END;


                                        .SBTTL  DUPWRTDBN MULTI-DRIVE TEST ROUTINES
000000  010146                  DUPWRTDBN:
                                        MOV     R1,-(SP)                         ;                                3550
000002  013746  000160'                 MOV     MX1,-(SP)                        ;                                3569
000006  012746  000104                  MOV     #104,-(SP)
000012  004737  000000G                 JSR     PC,BL$MUL
000016  012760  000022  000006G         MOV     #22,MSCP.PKT+6(R0)
000024  112760  000003  000022G         MOVB    #3,MSCP.PKT+22(R0)               ;                                3570
000032  012760  000001  000024G         MOV     #1,MSCP.PKT+24(R0)               ;                                3571
```

```
000040  112760  000127  000026G          MOVB    #127,MSCP.PKT+26(R0)            ;                       3572
000046  112760  000122  000027G          MOVB    #122,MSCP.PKT+27(R0)            ;                       3573
000054  112760  000124  000030G          MOVB    #124,MSCP.PKT+30(R0)            ;                       3574
000062  112760  000104  000031G          MOVB    #104,MSCP.PKT+31(R0)            ;                       3575
000070  112760  000102  000032G          MOVB    #102,MSCP.PKT+32(R0)            ;                       3576
000076  112760  000116  000033G          MOVB    #116,MSCP.PKT+33(R0)            ;                       3577
000104  004737  000000V                  JSR     PC,DUPCOMMAND                   ;                       3578
000110  013700  000000G                  MOV     CUOFF,R0                        ;                       3579
000114  006300                          ASL     R0
000116  063700  000000G                  ADD     CST.ADDR,R0
000122  032760  040000  000006          BIT     #40000,6(R0)
000130  001023                          BNE     2$                              ;                       3550
000132  013716  000000G                  MOV     CCTLR,(SP)                      ;                       3580
000136  004737  000000G                  JSR     PC,GET.PKT
000142  010037  000160'                  MOV     R0,MX1
000146  002404                          BLT     1$
000150  132737  000001  000000G          BITB    #1,EOP.FLAG                     ;                       3581
000156  001412                          BEQ     3$
000160  013700  000000G          1$:     MOV     CUOFF,R0                        ;                       3583
000164  006300                          ASL     R0
000166  063700  000000G                  ADD     CST.ADDR,R0
000172  052760  040000  000006          BIS     #40000,6(R0)
000200  022626                  2$:     CMP     (SP)+,(SP)+                     ;                       3580
000202  000504                          BR      6$                              ;                       3583
000204  013716  000160'          3$:     MOV     MX1,(SP)                        ;                       3588
000210  012746  000104                  MOV     #104,-(SP)
000214  004737  000000G                  JSR     PC,BL$MUL
000220  012760  000034  000006G          MOV     #34,MSCP.PKT+6(R0)
000226  112760  000005  000022G          MOVB    #5,MSCP.PKT+22(R0)              ;                       3589
000234  012760  000002  000026G          MOV     #2,MSCP.PKT+26(R0)              ;                       3590
000242  012760  000000G  000032G          MOV     #DUPPKT,MSCP.PKT+32(R0)         ;                       3591
000250  005060  000024G                  CLR     MSCP.PKT+24(R0)                 ;                       3592
000254  004737  000000V                  JSR     PC,DUPCOMMAND                   ;                       3593
000260  013700  000000G                  MOV     CUOFF,R0                        ;                       3594
000264  006300                          ASL     R0
000266  063700  000000G                  ADD     CST.ADDR,R0
000272  032760  040000  000006          BIT     #40000,6(R0)
000300  001043                          BNE     5$                              ;                       3550
000302  013700  000000G                  MOV     DUPPKT,R0                       ;                       3595
000306  042700  007777                  BIC     #7777,R0
000312  020027  010000                  CMP     R0,#10000
000316  001022                          BNE     4$
000320  013700  000000G                  MOV     DUPPKT,R0                       ;                       3596
000324  042700  170000                  BIC     #170000,R0
000330  020027  000006                  CMP     R0,#6
000334  001013                          BNE     4$
000336  013716  000000G                  MOV     CCTLR,(SP)                      ;                       3597
000342  004737  000000G                  JSR     PC,GET.PKT
000346  010037  000160'                  MOV     R0,MX1
000352  002404                          BLT     4$
000354  132737  000001  000000G          BITB    #1,EOP.FLAG                     ;                       3598
000362  001415                          BEQ     7$
000364  004737  000000V          4$:     JSR     PC,HARD.ERROR                   ;                       3600
```

```
000370  013700  000000G                        MOV     CUOFF,R0                        ;                       3601
000374  006300                                 ASL     R0
000376  063700  000000G                        ADD     CST.ADDR,R0
000402  052760  040000  000006                 BIS     #40000,6(R0)
000410  062706  000006                 5$:     ADD     #6,SP                           ;                       3595
000414  000504                         6$:     BR      10$                             ;                       3600
000416  013716  000160'                 7$:     MOV     MX1,(SP)                        ;                       3605
000422  012746  000104                         MOV     #104,-(SP)
000426  004737  000000G                        JSR     PC,BL$MUL
000432  012760  000034  000006G                MOV     #34,MSCP.PKT+6(R0)
000440  112760  000004  000022G                MOVB    #4,MSCP.PKT+22(R0)              ;                       3606
000446  012760  000006  000026G                MOV     #6,MSCP.PKT+26(R0)              ;                       3607
000454  012760  000000G 000032G                MOV     #DUPPKT,MSCP.PKT+32(R0)         ;                       3608
000462  013737  000000G 000000G                MOV     CDISK,DUPPKT                    ;                       3609
000470  013701  000000G                        MOV     CUOFF,R1                        ;                       3610
000474  006301                                 ASL     R1
000476  063701  000000G                        ADD     CST.ADDR,R1
000502  116137  000006  000002G                MOVB    6(R1),DUPPKT+2
000510  105037  000003G                        CLRB    DUPPKT+3
000514  013737  000000G 000004G                MOV     S.PATTERN,DUPPKT+4              ;                       3611
000522  005060  000024G                        CLR     MSCP.PKT+24(R0)                 ;                       3612
000526  004737  000000V                        JSR     PC,DUPCOMMAND                   ;                       3613
000532  013700  000000G                        MOV     CUOFF,R0                        ;                       3614
000536  006300                                 ASL     R0
000540  063700  000000G                        ADD     CST.ADDR,R0
000544  032760  040000  000006                 BIT     #40000,6(R0)
000552  001023                                 BNE     9$                              ;                       3550
000554  013716  000000G                        MOV     CCTLR,(SP)                      ;                       3615
000560  004737  000000G                        JSR     PC,GET.PKT
000564  010037  000160'                        MOV     R0,MX1
000570  002404                                 BLT     8$
000572  132737  000001  000000G                BITB    #1,EOP.FLAG                     ;                       3616
000600  001413                                 BEQ     11$
000602  013700  000000G                 8$:     MOV     CUOFF,R0                        ;                       3618
000606  006300                                 ASL     R0
000610  063700  000000G                        ADD     CST.ADDR,R0
000614  052760  040000  000006                 BIS     #40000,6(R0)
000622  062706  000010                 9$:     ADD     #10,SP                          ;                       3615
000626  000515                         10$:    BR      17$                             ;                       3618
000630  013716  000160'                 11$:    MOV     MX1,(SP)                        ;                       3622
000634  012746  000104                         MOV     #104,-(SP)
000640  004737  000000G                        JSR     PC,BL$MUL
000644  012760  000034  000006G                MOV     #34,MSCP.PKT+6(R0)
000652  112760  000005  000022G                MOVB    #5,MSCP.PKT+22(R0)              ;                       3623
000660  012760  000004  000026G                MOV     #4,MSCP.PKT+26(R0)              ;                       3624
000666  012760  000000G 000032G                MOV     #DUPPKT,MSCP.PKT+32(R0)         ;                       3625
000674  005060  000024G                        CLR     MSCP.PKT+24(R0)                 ;                       3626
000700  004737  000000V                        JSR     PC,DUPCOMMAND                   ;                       3627
000704  013700  000000G                        MOV     CUOFF,R0                        ;                       3628
000710  006300                                 ASL     R0
000712  063700  000000G                        ADD     CST.ADDR,R0
000716  032760  040000  000006                 BIT     #40000,6(R0)
000724  001054                                 BNE     16$                             ;                       3550
```

```
000726  013700  000000G                        MOV     DUPPKT,R0               ;                       3629
000732  042700  007777                         BIC     #7777,R0
000736  020027  030000                         CMP     R0,#30000
000742  001012                                 BNE     12$
000744  013700  000000G                        MOV     DUPPKT,R0               ;                       3630
000750  042700  170000                         BIC     #170000,R0
000754  020027  000003                         CMP     R0,#3
000760  001003                                 BNE     12$
000762  005737  000002G                        TST     DUPPKT+2                ;                       3631
000766  001403                                 BEQ     13$
000770  004737  000000V              12$:      JSR     PC,HARD.ERROR           ;                       3633
000774  000413                                 BR      14$                     ;                       3634
000776  013716  000000G              13$:      MOV     CCTLR,(SP)              ;                       3637
001002  004737  000000G                        JSR     PC,GET.PKT
001006  010037  000160'                        MOV     R0,MX1
001012  002404                                 BLT     14$
001014  132737  000001  000000G               BITB    #1,EOP.FLAG             ;                       3638
001022  001411                                 BEQ     15$
001024  013700  000000G              14$:      MOV     CUOFF,R0                ;                       3640
001030  006300                                 ASL     R0
001032  063700  000000G                        ADD     CST.ADDR,R0
001036  052760  040000  000006               BIS     #40000,6(R0)
001044  000404                                 BR      16$                     ;                       3637
001046  013700  000000G              15$:      MOV     T.ADDR,R0               ;                       3643
001052  005260  000050                         INC     50(R0)
001056  062706  000012              16$:      ADD     #12,SP                  ;                       3564
001062  012601                      17$:      MOV     (SP)+,R1                ;                       3550
001064  000207                                 RTS     PC

; Routine Size:  283 words,     Routine Base:  $CODE$ + 11744
; Maximum stack depth per invocation:  7 words
```

```
3645  ROUTINE DUPREDDBN : NOVALUE =
3646
3647  !+
3648  ! THIS ROUTINE IS CALLED BY DUP ROUTINE TO USE THE CONTROLLER LOCAL PROGRAM
3649  ! "REDDBN".  TO USE THE PROGRAM THE OPTIONAL DUP SUB-PROTOCAL IS USED TO
3650  ! COMMUNICATE WITH THE CONTROLLER.  THE PROGRAM READS A DIAGONOSTIC BLOCK (DBN)
3651  ! AND PLACES IT IN THE DUP BUFFER CALLED "DUPPKT".  IF AN ERROR OCCURS WHILE
3652  ! RUNNING THE CONTROLLER LOCAL PROGRAM THE ERROR IS USSUALLY REPORTED IN THE
3653  ! DUP BUFFER. (EX. ILLEGAL UNIT NUMBER, ILLEGAL BLK #, DEVICE ERROR, ZERO LENGHT MSG)
3654  !
3655  !
3656  !       IMPLICIT INPUTS:
3657  !               CST_ADDR   - CONTAINS THE CURRENT CONTROLLER STATUS TABLE
3658  !               CUOFF      - CURRENT OFFSET IN CST TABLE FOR PARTICULAR DRIVE
3659  !
3660  BEGIN
3661
3662  !PRINTX (DER12);
3663
3664  MSCP_PKT [.MX1, MSGLEN] = SZ_ELP;                     ! PACKET SIZE                                   EXECUTE REDDBN PROGRAM
3665  MSCP_PKT [.MX1, OPCODE] = OP_ELP;                     ! OPCODE = EXECUTE LOCAL PROGRAM
3666  MSCP_PKT [.MX1, MODIFY] = 1;                          ! STANDALONE MODIFIER
3667  MSCP_PKT [.MX1, L1] = %ascii'R';                      ! FILL IN PROGRAM NAME WITH ASCII LETTERS
3668  MSCP_PKT [.MX1, L2] = %ascii'E';
3669  MSCP_PKT [.MX1, L3] = %ascii'D';
3670  MSCP_PKT [.MX1, L4] = %ascii'D';
3671  MSCP_PKT [.MX1, L5] = %ascii'B';
3672  MSCP_PKT [.MX1, L6] = %ascii'N';
3673  DUPCOMMAND ();                                        ! SENDS AND RECEIVES THE COMMAND
3674  IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN        RETURN; !IF DUP ERROR THEN  RETURN
3675  IF (MX1 = GET_PKT (.CCTLR)) LSS 0                     ! TRY TO GET AN ENVELOPE. IF FAILURE
3676          OR (.EOP_FLAG)
3677          THEN
3678                  (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
3679                   RETURN;);                                    ! NO POINT IN CONTINUING
3680
3681
3682          MSCP_PKT [.MX1, MSGLEN] = SZ_REC;             ! PACKET SIZE                                   RECIEVE DATA
3683          MSCP_PKT [.MX1, OPCODE] = OP_RCD;             ! OPCODE = RECEIVE DATA
3684          MSCP_PKT [.MX1, BC_LO] = 2;                   ! BYTE COUNT TO BE TRANSFERED EQUALS 2
3685          MSCP_PKT [.MX1, BUF_0] = DUPPKT;              ! LOAD DESCRIBTOR BUFFER
3686          MSCP_PKT [.MX1, MODIFY] = 0;                  !
3687          DUPCOMMAND ();                                ! SENDS AND RECEIVES THE COMMAND
3688          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
3689          IF .DUPPKT [DUPTYPE]  NEQ 1
3690             OR .DUPPKT [DUPMESG] NEQ 5                 ! IF CORRECT RESPONSE
3691             OR (MX1 = GET_PKT (.CCTLR)) LSS 0          ! TRY TO GET AN ENVELOPE. IF FAILURE
3692             OR (.EOP_FLAG)
3693             THEN
3694                  (HARD_ERROR ();
3695                   CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
3696                   RETURN;);                                    ! NO POINT IN CONTINUING
3697
```

```
;        3698
;        3699          MSCP_PKT [.MX1, MSGLEN] = SZ_SEN;                ! PACKET SIZE                                   SEND DATA
;        3700          MSCP_PKT [.MX1, OPCODE] = OP_SDD;                ! OPCODE = SEND DATA
;        3701          MSCP_PKT [.MX1, BC_LO] = 4;                      ! BYTE COUNT TO BE TRANSFERED EQUALS 4
;        3702          MSCP_PKT [.MX1, BUF_0] = DUPPKT;                 ! LOAD DESCRIBTOR BUFFER
;        3703          DUPPKT [DUPBF0] = .CDISK;                        ! LOAD UNIT NUMBER (RDRX)
;        3704          DUPPKT [DUPBF1] = .CST_ADDR [.CUOFF + 3, D_DBN]; ! LOAD DBN NUMBER
;        3705          MSCP_PKT [.MX1, MODIFY] = 0;                     !
;        3706          DUPCOMMAND ();                                   ! SENDS AND RECEIVES THE COMMAND
;        3707          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
;        3708          IF (MX1 = GET_PKT (.CCTLR)) LSS 0                ! TRY TO GET AN ENVELOPE. IF FAILURE
;        3709                  OR (.EOP_FLAG)
;        3710                  THEN
;        3711                  (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;        3712                   RETURN;);                                        ! NO POINT IN CONTINUING
;        3713
;        3714
;        3715
;        3716          MSCP_PKT [.MX1, MSGLEN] = SZ_REC;                ! PACKET SIZE                                   RECEIVE DATA
;        3717          MSCP_PKT [.MX1, OPCODE] = OP_RCD;                ! OPCODE = GET DUST STATUS
;        3718          MSCP_PKT [.MX1, BC_LO] = 514;                    ! BYTE COUNT TO BE TRANSFERED EQUALS 512
;        3719          MSCP_PKT [.MX1, BUF_0] = DUPPKT;                 ! LOAD DESCRIBTOR BUFFER
;        3720          MSCP_PKT [.MX1, MODIFY] = 0;                     !
;        3721          DUPCOMMAND ();                                   ! SENDS AND RECEIVES THE COMMAND
;        3722          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
;        3723          IF .DUPPKT [DUPTYPE]  NEQ 6
;        3724             OR .DUPPKT [DUPMESG] NEQ 2                    ! IF CORRECT RESPONSE
;        3725             OR (MX1 = GET_PKT (.CCTLR)) LSS 0             ! TRY TO GET AN ENVELOPE. IF FAILURE
;        3726             OR (.EOP_FLAG)
;        3727             THEN
;        3728                  (HARD_ERROR ();
;        3729                   CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;        3730                   RETURN;);                                        ! NO POINT IN CONTINUING
;        3731
;        3732          CST_ADDR [.CUOFF + 3, D_DBN] = .CST_ADDR [.CUOFF + 3, D_DBN] + 1;       ! INCREMENT RELATIVE DBN COUNTER
;        3733          T_ADDR [T_BLK_RD] = .T_ADDR [T_BLK_RD] + 1;
;        3734
;        3735          END;


                                                .SBTTL  DUPREDDBN MULTI-DRIVE TEST ROUTINES
000000  010146                          DUPREDDBN:
                                                MOV     R1,-(SP)                        ;                               3645
000002  013746  000160'                         MOV     MX1,-(SP)                       ;                               3664
000006  012746  000104                          MOV     #104,-(SP)
000012  004737  000000G                         JSR     PC,BL$MUL
000016  012760  000022  000006G                 MOV     #22,MSCP.PKT+6(R0)
000024  112760  000003  000022G                 MOVB    #3,MSCP.PKT+22(R0)              ;                               3665
000032  012760  000001  000024G                 MOV     #1,MSCP.PKT+24(R0)              ;                               3666
000040  112760  000122  000026G                 MOVB    #122,MSCP.PKT+26(R0)            ;                               3667
000046  112760  000105  000027G                 MOVB    #105,MSCP.PKT+27(R0)            ;                               3668
000054  112760  000104  000030G                 MOVB    #104,MSCP.PKT+30(R0)            ;                               3669
000062  112760  000104  000031G                 MOVB    #104,MSCP.PKT+31(R0)            ;                               3670
```

```
000070  112760  000102  000032G          MOVB    #102,MSCP.PKT+32(R0)            ;                       3671
000076  112760  000116  000033G          MOVB    #116,MSCP.PKT+33(R0)            ;                       3672
000104  004737  000000V                  JSR     PC,DUPCOMMAND                   ;                       3673
000110  013700  000000G                  MOV     CUOFF,R0                        ;                       3674
000114  006300                           ASL     R0
000116  063700  000000G                  ADD     CST.ADDR,R0
000122  032760  040000  000006          BIT     #40000,6(R0)
000130  001023                           BNE     2$                              ;                       3645
000132  013716  000000G                  MOV     CCTLR,(SP)                      ;                       3675
000136  004737  000000G                  JSR     PC,GET.PKT
000142  010037  000160'                  MOV     R0,MX1
000146  002404                           BLT     1$
000150  132737  000001  000000G          BITB    #1,EOP.FLAG                     ;                       3676
000156  001412                           BEQ     3$
000160  013700  000000G          1$:     MOV     CUOFF,R0                        ;                       3678
000164  006300                           ASL     R0
000166  063700  000000G                  ADD     CST.ADDR,R0
000172  052760  040000  000006          BIS     #40000,6(R0)
000200  022626                   2$:     CMP     (SP)+,(SP)+                     ;                       3675
000202  000504                           BR      6$                              ;                       3678
000204  013716  000160'          3$:     MOV     MX1,(SP)                        ;                       3682
000210  012746  000104                   MOV     #104,-(SP)
000214  004737  000000G                  JSR     PC,BL$MUL
000220  012760  000034  000006G          MOV     #34,MSCP.PKT+6(R0)
000226  112760  000005  000022G          MOVB    #5,MSCP.PKT+22(R0)              ;                       3683
000234  012760  000002  000026G          MOV     #2,MSCP.PKT+26(R0)              ;                       3684
000242  012760  000000G  000032G          MOV     #DUPPKT,MSCP.PKT+32(R0)         ;                       3685
000250  005060  000024G                  CLR     MSCP.PKT+24(R0)                 ;                       3686
000254  004737  000000V                  JSR     PC,DUPCOMMAND                   ;                       3687
000260  013700  000000G                  MOV     CUOFF,R0                        ;                       3688
000264  006300                           ASL     R0
000266  063700  000000G                  ADD     CST.ADDR,R0
000272  032760  040000  000006          BIT     #40000,6(R0)
000300  001043                           BNE     5$                              ;                       3645
000302  013700  000000G                  MOV     DUPPKT,R0                       ;                       3689
000306  042700  007777                   BIC     #7777,R0
000312  020027  010000                   CMP     R0,#10000
000316  001022                           BNE     4$
000320  013700  000000G                  MOV     DUPPKT,R0                       ;                       3690
000324  042700  170000                   BIC     #170000,R0
000330  020027  000005                   CMP     R0,#5
000334  001013                           BNE     4$
000336  013716  000000G                  MOV     CCTLR,(SP)                      ;                       3691
000342  004737  000000G                  JSR     PC,GET.PKT
000346  010037  000160'                  MOV     R0,MX1
000352  002404                           BLT     4$
000354  132737  000001  000000G          BITB    #1,EOP.FLAG                     ;                       3692
000362  001415                           BEQ     7$
000364  004737  000000V          4$:     JSR     PC,HARD.ERROR                   ;                       3694
000370  013700  000000G                  MOV     CUOFF,R0                        ;                       3695
000374  006300                           ASL     R0
000376  063700  000000G                  ADD     CST.ADDR,R0
000402  052760  040000  000006          BIS     #40000,6(R0)
```

```
000410  062706  000006                  5$:     ADD     #6,SP                           ;                       3689
000414  000501                          6$:     BR      10$                             ;                       3694
000416  013716  000160'                 7$:     MOV     MX1,(SP)                        ;                       3699
000422  012746  000104                          MOV     #104,-(SP)
000426  004737  000000G                         JSR     PC,BL$MUL
000432  012760  000034  000006G                 MOV     #34,MSCP.PKT+6(R0)
000440  112760  000004  000022G                 MOVB    #4,MSCP.PKT+22(R0)              ;                       3700
000446  012760  000004  000026G                 MOV     #4,MSCP.PKT+26(R0)              ;                       3701
000454  012760  000000G 000032G                 MOV     #DUPPKT,MSCP.PKT+32(R0)         ;                       3702
000462  013737  000000G 000000G                 MOV     CDISK,DUPPKT                    ;                       3703
000470  013701  000000G                         MOV     CUOFF,R1                        ;                       3704
000474  006301                                  ASL     R1
000476  063701  000000G                         ADD     CST.ADDR,R1
000502  116137  000006  000002G                 MOVB    6(R1),DUPPKT+2
000510  105037  000003G                         CLRB    DUPPKT+3
000514  005060  000024G                         CLR     MSCP.PKT+24(R0)                 ;                       3705
000520  004737  000000V                         JSR     PC,DUPCOMMAND                   ;                       3706
000524  013700  000000G                         MOV     CUOFF,R0                        ;                       3707
000530  006300                                  ASL     R0
000532  063700  000000G                         ADD     CST.ADDR,R0
000536  032760  040000  000006                  BIT     #40000,6(R0)
000544  001023                                  BNE     9$                              ;                       3645
000546  013716  000000G                         MOV     CCTLR,(SP)                      ;                       3708
000552  004737  000000G                         JSR     PC,GET.PKT
000556  010037  000160'                         MOV     R0,MX1
000562  002404                                  BLT     8$
000564  132737  000001  000000G                 BITB    #1,EOP.FLAG                     ;                       3709
000572  001413                                  BEQ     11$
000574  013700  000000G                 8$:     MOV     CUOFF,R0                        ;                       3711
000600  006300                                  ASL     R0
000602  063700  000000G                         ADD     CST.ADDR,R0
000606  052760  040000  000006                  BIS     #40000,6(R0)
000614  062706  000010                  9$:     ADD     #10,SP                          ;                       3708
000620  000520                          10$:    BR      15$                             ;                       3711
000622  013716  000160'                 11$:    MOV     MX1,(SP)                        ;                       3716
000626  012746  000104                          MOV     #104,-(SP)
000632  004737  000000G                         JSR     PC,BL$MUL
000636  012760  000034  000006G                 MOV     #34,MSCP.PKT+6(R0)
000644  112760  000005  000022G                 MOVB    #5,MSCP.PKT+22(R0)              ;                       3717
000652  012760  001002  000026G                 MOV     #1002,MSCP.PKT+26(R0)           ;                       3718
000660  012760  000000G 000032G                 MOV     #DUPPKT,MSCP.PKT+32(R0)         ;                       3719
000666  005060  000024G                         CLR     MSCP.PKT+24(R0)                 ;                       3720
000672  004737  000000V                         JSR     PC,DUPCOMMAND                   ;                       3721
000676  013700  000000G                         MOV     CUOFF,R0                        ;                       3722
000702  006300                                  ASL     R0
000704  063700  000000G                         ADD     CST.ADDR,R0
000710  032760  040000  000006                  BIT     #40000,6(R0)
000716  001057                                  BNE     14$                             ;                       3645
000720  013700  000000G                         MOV     DUPPKT,R0                       ;                       3723
000724  042700  007777                          BIC     #7777,R0
000730  020027  060000                          CMP     R0,#60000
000734  001022                                  BNE     12$
000736  013700  000000G                         MOV     DUPPKT,R0                       ;                       3724
```

```
000742  042700  170000                          BIC     #170000,R0
000746  020027  000002                          CMP     R0,#2
000752  001013                                  BNE     12$
000754  013716  000000G                         MOV     CCTLR,(SP)              ;                       3725
000760  004737  000000G                         JSR     PC,GET.PKT
000764  010037  000160'                         MOV     R0,MX1
000770  002404                                  BLT     12$
000772  132737  000001  000000G                 BITB    #1,EOP.FLAG             ;                       3726
001000  001413                                  BEQ     13$
001002  004737  000000V                 12$:    JSR     PC,HARD.ERROR           ;                       3728
001006  013700  000000G                         MOV     CUOFF,R0                ;                       3729
001012  006300                                  ASL     R0
001014  063700  000000G                         ADD     CST.ADDR,R0
001020  052760  040000  000006                 BIS     #40000,6(R0)
001026  000413                                  BR      14$                     ;                       3723
001030  013700  000000G                 13$:    MOV     CUOFF,R0                ;                       3732
001034  006300                                  ASL     R0
001036  063700  000000G                         ADD     CST.ADDR,R0
001042  105260  000006                          INCB    6(R0)
001046  013700  000000G                         MOV     T.ADDR,R0               ;                       3733
001052  005260  000054                          INC     54(R0)
001056  062706  000012                  14$:    ADD     #12,SP                  ;                       3660
001062  012601                          15$:    MOV     (SP)+,R1                ;                       3645
001064  000207                                  RTS     PC

; Routine Size:  283 words,     Routine Base:  $CODE$ + 13032
; Maximum stack depth per invocation:  7 words


;               3736
```

```
;         3737
;         3738  ROUTINE DUPCOMMAND : NOVALUE =
;         3739
;         3740  !+
;         3741  ! THIS ROUTINE IS CALLED BY DUP TO PROCESS COMMANDS.
;         3742  ! THE COMMAND ENVOLOPES ARE FILLED IN DUP ROUTINES IN THE "MX1" INDEX.
;         3743  ! WITH THE INDEX THIS ROUTINE SENDS THE COMMAND, WAITS FOR A
;         3744  ! RESPONSES AND THEN PROCESSES THE RETURN PACKET.
;         3745  !-
;         3746  BEGIN
;         3747  !PRINTX (DER13);
;         3748  IF .EOP_FLAG
;         3749  THEN RETURN
;         3750  ELSE
;         3751          BEGIN
;         3752          MSCP_PKT [.MX1, CREDITS] = 1;                   ! CREDITS EQUALS 1
;         3753          MSCP_PKT [.MX1, MSGTYP] = 0;                    !
;         3754          MSCP_PKT [.MX1, CONNID] = 2;                    ! MAKE PACKAGE EQUAL A DUP COMMAND
;         3755          MSCP_PKT [.MX1, DK_NUM] = 0;            ! DISK NUMBER
;         3756
;         3757
;         3758          IF SEND (.MX1) EQLU FAILURE                     ! ATTEMPT SEND; IF CTLR IS OFFLINE
;         3759          THEN
;         3760                  BEGIN
;         3761                  PUT_PKT (.MX1);
;         3762                  MX1 = -1;                               ! RETURN ENVELOPE TO POOL
;         3763                  CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;         3764                  END
;         3765
;         3766          ELSE    BEGIN                                   ! IF SEND WAS SUCCESSFUL
;         3767                  QIO [.CCTLR] = .QIO [.CCTLR] + 1;               ! INCR OUTSTANDING QIO COUNT
;         3768                  BREAK;                                  ! BREAK FOR SUPERVISOR TO CATCH USER REQUESTS
;         3769                  WAIT ();                                ! WAIT FOR RETPKT RESPONSE
;         3770                  PROC_RETPKT();                          ! PROCESS RETURN PACKET TO SEE IF OK FOR DUP
;         3771                  END;
;         3772          END;
;         3773  END;


                                        .SBTTL  DUPCOMMAND MULTI-DRIVE TEST ROUTINES
000000  010146                  DUPCOMMAND:
                                        MOV     R1,-(SP)                        ;                       3738
000002  132737  000001  000000G         BITB    #1,EOP.FLAG                     ;                       3748
000010  001060                          BNE     3$                              ;                       3749
000012  013746  000160'                 MOV     MX1,-(SP)                       ;                       3752
000016  012746  000104                  MOV     #104,-(SP)
000022  004737  000000G                 JSR     PC,BL$MUL
000026  012701  000010G                 MOV     #MSCP.PKT+10,R1
000032  060001                          ADD     R0,R1
000034  112711  000001                  MOVB    #1,(R1)                         ;                       3753
000040  112761  000002  000001          MOVB    #2,1(R1)                        ;                       3754
000046  005060  000016G                 CLR     MSCP.PKT+16(R0)                 ;                       3755
000052  013716  000160'                 MOV     MX1,(SP)                        ;                       3758
```

```
000056  004737  000000G                        JSR     PC,SEND
000062  005700                                  TST     R0
000064  001020                                  BNE     1$
000066  013716  000160'                        MOV     MX1,(SP)                ;                       3761
000072  004737  000000G                        JSR     PC,PUT.PKT
000076  012737  177777  000160'                MOV     #-1,MX1                 ;                       3762
000104  013700  000000G                        MOV     CUOFF,R0                ;                       3763
000110  006300                                  ASL     R0
000112  063700  000000G                        ADD     CST.ADDR,R0
000116  052760  040000  000006                BIS     #40000,6(R0)
000124  000411                                  BR      2$                      ;                       3758
000126  013700  000000G                1$:     MOV     CCTLR,R0                ;                       3767
000132  105260  000000G                        INCB    QIO(R0)
000136  104422                                  TRAP    22
000140  004737  000000G                        JSR     PC,WAIT                 ;                       3769
000144  004737  000000V                        JSR     PC,PROC.RETPKT          ;                       3770
000150  022626                        2$:     CMP     (SP)+,(SP)+             ;                       3751
000152  012601                        3$:     MOV     (SP)+,R1                ;                       3738
000154  000207                                RTS     PC

; Routine Size:  55 words,      Routine Base:  $CODE$ + 14120
; Maximum stack depth per invocation:  5 words

;        3774
```

```
;        3775  routine QIO_LBN : novalue =
;        3776
;        3777  !+
;        3778  !       THIS ROUTINE IS CALLED BY QIO_GEN TO SELECT THE LOGICAL BLOCK NUMBER TO
;        3779  !       BE USED FOR THE CURRENT QIO OR QIO PAIR.
;        3780  !
;        3781  !       IF THE OPERATOR CHOSE THE RANDOM SEEK MODE OPTION, THEN THE LBN IS
;        3782  !       RANDOMLY CHOSEN WITHIN THE SPECIFIED LIMITS FOR THE LBN.
;        3783  !       OTHERWISE, THE NEXT SEQUENTIAL LBN IS DERIVED FROM THE BLOCK SEQUENCE
;        3784  !       TABLE (BST).
;        3785  !
;        3786  !       IMPLICIT INPUTS:
;        3787  !               L$LUN - CURRENT (DIAGNOSTIC SUPERVIOR) UNIT NUMBER
;        3788  !
;        3789  !       IMPLICIT OUTPUTS:
;        3790  !               THE LBN IS LOADED INTO ONE OR BOTH MSCP PACKETS.
;        3791  !-
;        3792
;        3793      begin
;        3794
;        3795      own
;        3796          LBN_SAVE : word initial (0);            ! LBN SELECTED IN LAST PASS
;        3797
;        3798      local
;        3799          S_TEMP : word,                          ! TEMPORARY STORAGE FOR START LBN
;        3800          E_TEMP : word,                          ! TEMPORARY STORAGE FOR END LBN
;        3801          LBN : word,                             ! LOGICAL BLOCK NUMBER
;        3802          RD51_DISK : byte initial (byte (TRUE)); ! FLAG TO INDICATE WINCHESTER DISK SELECTED
;        3803
;        3804      S_TEMP = .CST_ADDR [.CUOFF + 1, D_BEG];     ! STARTING LBN
;        3805      E_TEMP = .CST_ADDR [.CUOFF + 2, D_END];     ! ENDING LBN
;        3806
;        3807      if .CST_ADDR [.CUOFF, D_TYPE] eql RX_50
;        3808      then
;        3809          RD51_DISK = FALSE;
;        3810
;        3811      if BIT_TST (SWP_FLAGS, SWF_RDM)             ! IF RANDOM SEEK MODE
;        3812      then
;        3813          begin
;        3814
;        3815          if (.RD51_DISK) and
;        3816             (((.RANDOM [0] and %o'077777') mod (99)) lequ 33)
;        3817          then
;        3818              LBN = .LBN_SAVE                     ! REDUCE SEEKS ON RDs by 33%
;        3819          else
;        3820              LBN = .S_TEMP + ((.RANDOM [3] and %o'077777') mod (.E_TEMP - .S_TEMP + 1));
;        3821
;        3822          end
;        3823      else
;        3824          begin                                   ! ELSE - SEQUENTIAL LBN MODE
;        3825          LBN = .BST [.L$LUN];                    ! GET LBN FROM BST
;        3826          BST [.L$LUN] = .BST [.L$LUN] + .TRK_SGN [.L$LUN];! MODIFY LBN (INC OR DEC FOR NEXT PASS)
;        3827
```

```
;         3828          if .TRK_SGN [.L$LUN] lss 0
;         3829          then
;         3830              begin
;         3831
;         3832              if .BST [.L$LUN] lssu .S_TEMP       ! IF SECTOR IS LESS THAN LOWER LIMIT
;         3833              then
;         3834                  begin
;         3835                  BST [.L$LUN] = .S_TEMP;
;         3836                  TRK_SGN [.L$LUN] = 1;                   ! CHANGE TRACK DIRECTION
;         3837                  end;
;         3838
;         3839              end
;         3840          else
;         3841              begin
;         3842
;         3843              if .BST [.L$LUN] gtru .E_TEMP       ! IF SECTOR IS BEYOND HIGH LIMIT
;         3844              then
;         3845                  begin
;         3846                  BST [.L$LUN] = .E_TEMP;
;         3847                  TRK_SGN [.L$LUN] = -1;                  ! CHANGE TRACK DIRECTION
;         3848                  end;
;         3849
;         3850              end;
;         3851
;         3852          end;
;         3853
;         3854      if .LBN lssu .S_TEMP                        ! MAKE SURE LBN WITHIN LIMITS
;         3855      then
;         3856          LBN = .S_TEMP;
;         3857
;         3858      if .LBN gtru .E_TEMP                        !
;         3859      then
;         3860          LBN = .E_TEMP;
;         3861
;         3862      MAD1 [LBN_L] = .LBN;                        ! LOAD LBN INTO 1ST PACKET
;         3863
;         3864      if .MX2 geq 0                               ! IF 2 QIOS
;         3865      then
;         3866          MAD2 [LBN_L] = .LBN;                    ! LOAD LBN INTO 2ND PACKET
;         3867
;         3868      LBN_SAVE = .LBN;                            ! SAVE FOR USE NEXT CYCLE
;         3869
;         3870      end;                                        ! ROUTINE QIO_LBN


001252                                  .PSECT  $GGG$,  RO
001252  000000                  LBN.SAVE:
                                        .WORD   0


                                        .SBTTL  QIO.LBN MULTI-DRIVE TEST ROUTINES
014276                                  .PSECT  $CODE$,  RO
```

```
000000 004137 000000G          QIO.LBN:JSR     R1,$SAVE5                       ;                                       3775
000004 112705 000001                   MOVB    #1,R5                           ; *,RD51.DISK                           3793
000010 013702 000000G                  MOV     CST.ADDR,R2                     ;                                       3804
000014 013701 000000G                  MOV     CUOFF,R1
000020 010100                          MOV     R1,R0
000022 006300                          ASL     R0
000024 060200                          ADD     R2,R0
000026 016003 000002                   MOV     2(R0),R3                        ; *,S.TEMP
000032 010100                          MOV     R1,R0                           ;                                       3805
000034 006300                          ASL     R0
000036 060200                          ADD     R2,R0
000040 016004 000004                   MOV     4(R0),R4                        ; *,E.TEMP
000044 006301                          ASL     R1                              ;                                       3807
000046 060201                          ADD     R2,R1
000050 132711 000004                   BITB    #4,(R1)
000054 001001                          BNE     1$
000056 105005                          CLRB    R5                              ; RD51.DISK                             3809
000060 032737 000002 000000G   1$:     BIT     #2,SWP.FLAGS                    ;                                       3811
000066 001437                          BEQ     3$
000070 006005                          ROR     R5                              ; RD51.DISK                             3815
000072 103017                          BCC     2$
000074 013746 000116'                  MOV     RANDOM,-(SP)                    ;                                       3816
000100 042716 100000                   BIC     #100000,(SP)
000104 012746 000143                   MOV     #143,-(SP)
000110 004737 000000G                  JSR     PC,BL$MOD
000114 022626                          CMP     (SP)+,(SP)+
000116 020027 000041                   CMP     R0,#41
000122 101003                          BHI     2$
000124 013701 001252'                  MOV     LBN.SAVE,R1                     ; *,LBN                                 3818
000130 000445                          BR      5$                              ;                                       3815
000132 013746 000124'          2$:     MOV     RANDOM+6,-(SP)                  ;                                       3820
000136 042716 100000                   BIC     #100000,(SP)
000142 010400                          MOV     R4,R0                           ; E.TEMP,*
000144 160300                          SUB     R3,R0                           ; S.TEMP,*
000146 010046                          MOV     R0,-(SP)
000150 005216                          INC     (SP)
000152 004737 000000G                  JSR     PC,BL$MOD
000156 060300                          ADD     R3,R0                           ; S.TEMP,*
000160 010001                          MOV     R0,R1                           ; *,LBN
000162 022626                          CMP     (SP)+,(SP)+
000164 000427                          BR      5$                              ;                                       3811
000166 013700 000000G          3$:     MOV     L$LUN,R0                        ;                                       3825
000172 006300                          ASL     R0
000174 012702 000050'                  MOV     #BST,R2
000200 060002                          ADD     R0,R2
000202 011201                          MOV     (R2),R1                         ; *,LBN
000204 062700 001220'                  ADD     #TRK.SGN,R0                     ;                                       3826
000210 061012                          ADD     (R0),(R2)
000212 005710                          TST     (R0)                            ;                                       3828
000214 002006                          BGE     4$
000216 021203                          CMP     (R2),R3                         ; *,S.TEMP                              3832
000220 103011                          BHIS    5$
```

```
000222  010312                          MOV     R3,(R2)         ; S.TEMP,*                      3835
000224  012710  000001                   MOV     #1,(R0)         ;                               3836
000230  000405                          BR      5$              ;                               3828
000232  021204                  4$:     CMP     (R2),R4         ; *,E.TEMP                      3843
000234  101403                          BLOS    5$                                              3844
000236  010412                          MOV     R4,(R2)         ; E.TEMP,*                      3846
000240  012710  177777                   MOV     #-1,(R0)        ;                               3847
000244  020103                  5$:     CMP     R1,R3           ; LBN,S.TEMP                    3854
000246  103001                          BHIS    6$                                              3855
000250  010301                          MOV     R3,R1           ; S.TEMP,LBN                    3856
000252  020104                  6$:     CMP     R1,R4           ; LBN,E.TEMP                    3858
000254  101401                          BLOS    7$                                              3859
000256  010401                          MOV     R4,R1           ; E.TEMP,LBN                    3860
000260  013700  000164'          7$:     MOV     MAD1,R0         ;                               3862
000264  010160  000046                   MOV     R1,46(R0)       ; LBN,*
000270  005737  000162'                   TST     MX2             ;                               3864
000274  002404                          BLT     8$                                              3865
000276  013700  000166'                   MOV     MAD2,R0         ;                               3866
000302  010160  000046                   MOV     R1,46(R0)       ; LBN,*
000306  010137  001252'          8$:     MOV     R1,LBN.SAVE     ; LBN,*                         3868
000312  000207                          RTS     PC              ;                               3775

; Routine Size:  102 words,    Routine Base:  $CODE$ + 14276
; Maximum stack depth per invocation:  9 words
```

```
;         3871  routine QIO_SIZE : novalue =
;         3872
;         3873  !+
;         3874  !       THIS ROUTINE IS CALLED BY QIO_GEN TO SELECT THE I/O TRANSFER BYTE COUNT
;         3875  !       TO BE USED FOR THE CURRENT QIO OR QIO PAIR. THE BYTE COUNT IS
;         3876  !       DETERMINED BY A RANDOM NUMBER, AND WILL ALWAYS FALL BETWEEN 1 AND THE
;         3877  !       I/O BUFFER SIZE (BYTS_PER_QIO).
;         3878  !
;         3879  !       IMPLICIT OUTPUTS:
;         3880  !               THE BYTE COUNT IS LOADED INTO ONE OR BOTH MSCP PACKETS.
;         3881  !-
;         3882
;         3883      begin
;         3884
;         3885      local
;         3886          SIZE : word,                                                    ! BYTE COUNT
;         3887          BLOCKS_LEFT : word;                                             ! REMAINING BLOCKS LEFT
;         3888
;         3889      SIZE = ((.RANDOM [4] and %o'077777') mod (.BYTS_PER_QIO + 1)) and %o'177760';!GET BYTE COUNT FROM RANDOM NUMBER
;         3890
;         3891      if .SIZE eql 0
;         3892      then
;         3893          SIZE = 16;
;         3894
;         3895      BLOCKS_LEFT = .CST_ADDR [.CUOFF + 2, D_END] - .MAD1 [LBN_L] + 1;    ! REMAINING BLOCK COUNT
;         3896
;         3897      if ((.SIZE + BYTES_PER_SECT - 1) / BYTES_PER_SECT) gtru .BLOCKS_LEFT ! IF BLOCK COUNT NOT ENOUGH
;         3898      then
;         3899          SIZE = .BLOCKS_LEFT * BYTES_PER_SECT;                           ! ADJUST BYTE COUNT DOWN
;         3900
;         3901      MAD1 [BC_LO] = .SIZE;                                               ! LOAD SIZE INTO 1ST MSCP PACKET
;         3902
;         3903      if .MX2 geq 0                                                       ! IF 2 QIOS
;         3904      then
;         3905          MAD2 [BC_LO] = .SIZE;                                           ! LOAD SIZE INTO 2ND MSCP PACKET
;         3906
;         3907      end;                                                                ! ROUTINE QIO_SIZE

                                        .SBTTL  QIO.SIZE MULTI-DRIVE TEST ROUTINES
000000  004137  000000G       QIO.SIZE:
                                        JSR     R1,$SAVE2                       ;                       3871
000004  013746  000126'                 MOV     RANDOM+10,-(SP)                 ;                       3889
000010  042716  100000                  BIC     #100000,(SP)
000014  013746  000000G                 MOV     BYTS.PER.QIO,-(SP)
000020  005216                          INC     (SP)
000022  004737  000000G                 JSR     PC,BL$MOD
000026  010002                          MOV     R0,R2                           ; *,SIZE
000030  042702  000017                  BIC     #17,R2      -                   ; *,SIZE
000034  001002                          BNE     1$                              ;                       3891
000036  012702  000020                  MOV     #20,R2                          ; *,SIZE                3893
000042  013701  000000G       1$:       MOV     CUOFF,R1                        ;                       3895
000046  006301                          ASL     R1
```

```
000050  063701  000000G                        ADD     CST.ADDR,R1
000054  013700  000164'                        MOV     MAD1,R0
000060  016101  000004                        MOV     4(R1),R1
000064  166001  000046                        SUB     46(R0),R1
000070  005201                                INC     R1
000072  010216                                MOV     R2,(SP)                 ; SIZE,*                        3897
000074  062716  000777                        ADD     #777,(SP)
000100  012746  001000                        MOV     #1000,-(SP)
000104  004737  000000G                        JSR     PC,BL$DIV
000110  005726                                TST     (SP)+
000112  020001                                CMP     R0,R1                   ; *,BLOCKS.LEFT
000114  101405                                BLOS    2$
000116  010100                                MOV     R1,R0                   ; BLOCKS.LEFT,*                 3899
000120  000300                                SWAB    R0
000122  105000                                CLRB    R0
000124  006300                                ASL     R0
000126  010002                                MOV     R0,R2                   ; *,SIZE
000130  013700  000164'                2$:     MOV     MAD1,R0                 ;                               3901
000134  010260  000026                        MOV     R2,26(R0)               ; SIZE,*
000140  005737  000162'                        TST     MX2                     ;                               3903
000144  002404                                BLT     3$
000146  013700  000166'                        MOV     MAD2,R0                 ;                               3905
000152  010260  000026                        MOV     R2,26(R0)               ; SIZE,*
000156  022626                        3$:     CMP     (SP)+,(SP)+             ;                               3883
000160  000207                                RTS     PC                      ;                               3871

; Routine Size:  57 words,      Routine Base:  $CODE$ + 14612
; Maximum stack depth per invocation:  7 words
```

```
;       3908  routine FILL_BUFF : novalue =
;       3909
;       3910  !+
;       3911  !       THIS ROUTINE IS CALLED BY QIO_GEN TO LOAD THE I/O BUFFER DESCRIBED IN
;       3912  !       THE FIRST MSCP PACKET WITH THE APPROPRIATE DATA PATTERN.
;       3913  !
;       3914  !       THE DATA PATTERN TO BE SELECTED IS BASED ON THE FOLLOWING ALGORITHM:
;       3915  !
;       3916  !               IF THE OPERATOR DEFINED A DATA PATTERN
;       3917  !               THEN
;       3918  !                       SELECT IT
;       3919  !               ELSE
;       3920  !                       GET DATA PATTERN NUMBER FROM SW P-TABLE
;       3921  !                       IF DATA PATTERN NUMBER = 0
;       3922  !                       THEN
;       3923  !                               GET DATA PATTERN NUMBER FROM THE UNIT'S ENTRY
;       3924  !                               IN THE DATA PATTERN SEQUENCE TABLE (DPST)
;       3925  !
;       3926  !       NOTE THAT PATTERN # 1 CONSISTS OF RANDOM NUMBERS, AND PATTERNS # 17 -
;       3927  !       21 USE THE ACTUAL LBN OF THE WRITE REQUEST.
;       3928  !
;       3929  !       IMPLICIT INPUTS:
;       3930  !               L$LUN - CURRENT (DRS) UNIT NUMBER
;       3931  !-
;       3932
;       3933      begin
;       3934
;       3935      local
;       3936          DP_NUM : word,                          ! DATA PATTERN NUMBER SELECTED
;       3937          DP_ADDR,                                ! ADDR OF DATA PATTERN (LENGTH)
;       3938          IOB_ADDR,                               ! I/O BUFFER ADDRESS (DESTINATION)
;       3939          SRC_ADDR,                               ! WORKING SOURCE ADDRESS
;       3940          COUNT : word;                           ! NO. OF WORDS IN DATA PATTERN
;       3941
;       3942      if BIT_TST (SWP_FLAGS, SWF_UDP)             ! IF USER DEFINED A DATA PATTERN
;       3943      then
;       3944          DP_ADDR = SWP_UCNT                      ! SELECT IT
;       3945      else
;       3946          begin
;       3947
;       3948          if .SWP_DPAT neq 0                      ! IF USER SELECTED A PRE-DEFINED DATA PATTERN
;       3949          then
;       3950              DP_NUM = .SWP_DPAT                  ! SELECT IT
;       3951          else
;       3952              begin
;       3953              DP_NUM = .DPST [.L$LUN];            ! GET PATTERN NUMBER FROM SEQUENCE TABLE
;       3954              DPST [.L$LUN] = .DPST [.L$LUN] + 1; ! ADVANCE TO NEXT PATTERN NUMBER
;       3955
;       3956              if .DPST [.L$LUN] gtru DP_CNT       ! CHECK FOR HIGH LIMIT
;       3957              then
;       3958                  DPST [.L$LUN] = 1;
;       3959
;       3960              end;
```

```
;         3961
;         3962          DP_ADDR = .DPA_TBL [.DP_NUM - 1];       ! ADDRESS OF DATA PATTERN (COUNT)
;         3963
;         3964          if .DP_NUM gequ 17
;         3965          then
;         3966
;         3967              if .DP_NUM                          ! CHECK MACRO          ! IF PATTERN 17, 19, OR 21
;         3968              then
;         3969                  (.DP_ADDR + 2) = .MAD1 [LBN_L]  ! LOAD LBN INTO FIRST WORD OF PATTERN
;         3970              else
;         3971                  (.DP_ADDR + 4) = .MAD1 [LBN_L]; ! LOAD LBN INTO SECOND WORD OF PATTERN
;         3972
;         3973              end;
;         3974
;         3975          IOB_ADDR = .MAD1 [BUF_0];               ! I/O BUFFER ADDRESS
;         3976          COUNT = ..DP_ADDR;                      ! NO. OF WORDS IN DATA PATTERN
;         3977          SRC_ADDR = .DP_ADDR + 2;                ! START OF THE ACTUAL DATA PATTERN
;         3978
;         3979          incr N from 1 to ((.MAD1 [BC_LO] + 1) / 2) do       ! FOR EACH WORD IN THIS WRITE REQUEST
;         3980              begin
;         3981              .IOB_ADDR = ..SRC_ADDR;             ! MOVE 1 WORD
;         3982              IOB_ADDR = .IOB_ADDR + 2;           ! ADVANCE DESTINATION ADDRESS
;         3983              SRC_ADDR = .SRC_ADDR + 2;           ! ADVANCE SOURCE ADDRESS
;         3984              COUNT = .COUNT - 1;                 ! DECREMENT COUNT
;         3985
;         3986              if .COUNT eql 0                     ! IF END OF DATA PATTERN
;         3987              then
;         3988                  begin
;         3989                  COUNT = ..DP_ADDR;              ! REPEAT DATA PATTERN
;         3990                  SRC_ADDR = .DP_ADDR + 2;
;         3991                  end;
;         3992
;         3993              end;                                ! WORD TRANSFER LOOP
;         3994
;         3995          end;                                    ! ROUTINE FILL_BUFF


                                        .SBTTL  FILL.BUFF MULTI-DRIVE TEST ROUTINES
000000  004137  000000G          FILL.BUFF:
                                        JSR     R1,$SAVE5                       ;                               3908
000004  005746                          TST     -(SP)
000006  032737  000100  000000G         BIT     #100,SWP.FLAGS                  ;                               3942
000014  001403                          BEQ     1$
000016  012703  000000G                 MOV     #SWP.UCNT,R3                    ; *,DP.ADDR                     3944
000022  000443                          BR      5$                              ;                               3942
000024  013700  000000G          1$:    MOV     SWP.DPAT,R0                     ;                               3948
000030  001402                          BEQ     2$
000032  010001                          MOV     R0,R1                           ; *,DP.NUM                      3950
000034  000414                          BR      3$                              ;                               3948
000036  013700  000000G          2$:    MOV     L$LUN,R0                        ;                               3953
000042  062700  000060'                 ADD     #DPST,R0
000046  005001                          CLR     R1                              ; DP.NUM
000050  151001                          BISB    (R0),R1                         ; *,DP.NUM
```

```
000052  105210                          INCB    (R0)                    ;                               3954
000054  121027  000025                  CMPB    (R0),#25                ;                               3956
000060  101402                          BLOS    3$
000062  112710  000001                  MOVB    #1,(R0)                 ;                               3958
000066  010100                  3$:     MOV     R1,R0                   ; DP.NUM,*                      3962
000070  006300                          ASL     R0
000072  016003  001144'                  MOV     DPA.TBL-2(R0),R3        ; *,DP.ADDR
000076  020127  000021                  CMP     R1,#21                  ; DP.NUM,*                      3964
000102  103413                          BLO     5$
000104  013700  000164'                  MOV     MAD1,R0                 ;                               3969
000110  006001                          ROR     R1                      ; DP.NUM                        3967
000112  103004                          BCC     4$
000114  016063  000046  000002          MOV     46(R0),2(R3)            ; *,*(DP.ADDR)                  3969
000122  000403                          BR      5$                      ;                               3967
000124  016063  000046  000004  4$:     MOV     46(R0),4(R3)            ; *,*(DP.ADDR)                  3971
000132  013700  000164'          5$:     MOV     MAD1,R0                 ;                               3975
000136  016005  000032                  MOV     32(R0),R5               ; *,IOB.ADDR
000142  011302                          MOV     (R3),R2                 ; DP.ADDR,COUNT                 3976
000144  012704  000002                  MOV     #2,R4                   ;                               3977
000150  060304                          ADD     R3,R4                   ; DP.ADDR,*
000152  010416                          MOV     R4,(SP)                 ; *,SRC.ADDR
000154  016046  000026                  MOV     26(R0),-(SP)            ;                               3979
000160  005216                          INC     (SP)
000162  012746  000002                  MOV     #2,-(SP)
000166  004737  000000G                 JSR     PC,BL$DIV
000172  005001                          CLR     R1                      ; N
000174  000412                          BR      7$
000176  017625  000004          6$:     MOV     @4(SP),(R5)+            ; SRC.ADDR,IOB.ADDR             3981
000202  062766  000002  000004          ADD     #2,4(SP)                ; *,SRC.ADDR                    3983
000210  005302                          DEC     R2                      ; COUNT                         3984
000212  001003                          BNE     7$                      ;                               3986
000214  011302                          MOV     (R3),R2                 ; DP.ADDR,COUNT                 3989
000216  010466  000004                  MOV     R4,4(SP)                ; *,SRC.ADDR                    3990
000222  005201                  7$:     INC     R1                      ; N                             3979
000224  020100                          CMP     R1,R0                   ; N,*
000226  003763                          BLE     6$
000230  062706  000006                  ADD     #6,SP                   ;                               3908
000234  000207                          RTS     PC

; Routine Size:  79 words,      Routine Base:  $CODE$ + 14774
; Maximum stack depth per invocation:  10 words
```

```
;        3996  routine PROC_RETPKT : novalue =
;        3997
;        3998  !+
;        3999  !       THIS ROUTINE IS CALLED FROM THE MULTI_DRIVE "EXECUTIVE" AND DUP_COMMAND TO CHECK FOR
;        4000  !       AND PROCESS ANY RETURN PACKETS THAT HAVE BEEN "SENT" BY THE "DRIVER"
;        4001  !       PORTION OF THE PROGRAM. THE I/O DONE QUEUE (IODQ) ACTS AS THE LINK
;        4002  !       BETWEEN THE TWO PROGRAM PARTS; IT HOLDS INDECES OF RETURN PACKETS WHICH
;        4003  !       REQUIRE PROCESSING.
;        4004  !
;        4005  !       UNDER THE MULTI-DRIVE TEST, RETURN PACKETS ORIGINATE FROM TWO SOURCES:
;        4006  !               1. MSCP         - THE MORE COMMON, DESCRIBING A COMPLETED I/O
;        4007  !                                 OPERATION.
;        4008  !               2. DUP          - THE LESS COMMON, DESCRIBING A PORTION OF I/O
;        4009  !                                 COMMUNICATIONS WITH THE CONTROLLER PROGRAM.
;        4010  !               3. THE PROGRAM "DRIVER" - DESCRIBING A CONTROLLER ERROR OR
;        4011  !                                 COMMAND TIMEOUT.
;        4012  !-
;        4013
;        4014      while .IODQ_IN neq .IODQ_OUT do                   ! DO UNTIL I/O DONE QUEUE IS EMPTY
;        4015          begin
;        4016          RP_INDX = OUT_IODQ ();                        ! GET INDEX OF NEXT RETPKT AND ADVANCE OUT POINTER
;        4017          RP_ADDR = RETPKT + (.RP_INDX * RP_LEN * 2);   ! CALCULATE RETPKT ADDRESS
;        4018          SET_CPAR (.RP_ADDR [CTLR]);                   ! SET UP CURRENT CONTROLLER PARAMETERS
;        4019
;        4020          selectoneu .RP_ADDR [CONID] of                ! CONNECTION ID INDICATES PACKET SOURCE
;        4021              set
;        4022
;        4023              [CID_MSCP] :      IO_RETPKT ();           ! DISK MSCP (I/O TRANSFER DONE)
;        4024              [CID_DUP] :       DIO_RETPKT ();          ! DUP (I/O TRANSFER DONE)
;        4025              [CID_DRIVER] :    DR_RETPKT ();           ! MESSAGE FROM "DRIVER"
;        4026
;        4027              [otherwise] :     PRINTF (DBM12, .RP_ADDR [CONID]);!"CONN ID = XXXXX RECEIVED"
;        4028              tes;
;        4029
;        4030          end;                                          ! UNITL I/O DONE QUEUE IS EMPTY


                                        .SBTTL  PROC.RETPKT MULTI-DRIVE TEST ROUTINES
000000  010146                  PROC.RETPKT:
                                MOV     R1,-(SP)                        ;                       3996
000002  023737  000000G 000000G 1$:     CMP     IODQ.IN,IODQ.OUT        ;                       4014
000010  001463                  BEQ     6$
000012  004737  000000G                 JSR     PC,OUT.IODQ             ;                       4016
000016  010037  000000G                 MOV     R0,RP.INDX
000022  010046                  MOV     R0,-(SP)                        ; RP.INDX,*             4017
000024  012746  000060          MOV     #60,-(SP)
000030  004737  000000G                 JSR     PC,BL$MUL
000034  062700  000000G                 ADD     #RETPKT,R0
000040  010037  000000G                 MOV     R0,RP.ADDR
000044  116016  000002          MOVB    2(R0),(SP)                      ;                       4018
000050  042716  177760          BIC     #177760,(SP)
000054  004737  000000G                 JSR     PC,SET.CPAR
000060  013700  000000G                 MOV     RP.ADDR,R0              ;                       4020
```

```
000064  005001                          CLR     R1
000066  156001  000003                  BISB    3(R0),R1
000072  005701                          TST     R1
000074  001003                          BNE     2$
000076  004737  000000V                 JSR     PC,IO.RETPKT            ;                       4023
000102  000424                          BR      5$                      ;                       4020
000104  020127  000002          2$:     CMP     R1,#2
000110  001003                          BNE     3$
000112  004737  000000V                 JSR     PC,DIO.RETPKT           ;                       4024
000116  000416                          BR      5$                      ;                       4020
000120  020127  000003          3$:     CMP     R1,#3
000124  001003                          BNE     4$
000126  004737  000000V                 JSR     PC,DR.RETPKT            ;                       4025
000132  000410                          BR      5$                      ;                       4020
000134  010116                  4$:     MOV     R1,(SP)                 ;                       4027
000136  012746  000000G                 MOV     #DBM12,-(SP)
000142  012746  000002                  MOV     #2,-(SP)
000146  010600                          MOV     SP,R0                   ; SP,*
000150  104417                          TRAP    17
000152  022626                          CMP     (SP)+,(SP)+
000154  022626                  5$:     CMP     (SP)+,(SP)+             ;                       4015
000156  000711                          BR      1$                      ;                       4014
000160  012601                  6$:     MOV     (SP)+,R1                ;                       3996
000162  000207                          RTS     PC

; Routine Size:  58 words,      Routine Base:  $CODE$ + 15232
; Maximum stack depth per invocation:  7 words
```

```
;        4031  !^
;        4032  ROUTINE DIO_RETPKT : NOVALUE =
;        4033
;        4034  !+
;        4035  !       THIS ROUTINE IS CALLED BY PROC_RETPKT TO HANDLE ALL DUP I/O TRANSFER
;        4036  !       RETURN PACKETS. PROCESSING OF THESE PACKETS INCLUDES DECLARING ANY
;        4037  !       HARD ERRORS THAT MAY HAVE OCCURRED, UPDATING THE STATISTICS.
;        4038  !
;        4039  !       IMPLICIT INPUTS:
;        4040  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;        4041  !               T_ADDR - ADDRESS OF THE CURRENT UNIT'S STATISTICS BLOCK (TALLY)
;        4042  !               CST_ADDR - ADDRESS OF THE CURRENT CONTROLLER'S CST
;        4043  !               CUOFF - CST OFFSET FOR THE CURRENT UNIT
;        4044  !               L$LUN - CURRENT UNIT NUMBER
;        4045  !               CCTLR - CURRENT CONTROLLER NUMBER
;        4046  !
;        4047  !       IMPLICIT OUTPUTS
;        4048  !               CST_ADDR [.CUOFF + 3, NODUPMEDIA] - IF THIS BIT SET NO DUP EXERCISER
;        4049  !
;        4050  !-
;        4051  BEGIN
;        4052
;        4053  LOCAL   FLAG : BYTE INITIAL(BYTE(TRUE)),
;        4054          SUM2 : WORD,
;        4055          SUM : WORD;                             ! TOTAL NUMBER OF BYTES TRANSFERRED TO/FROM A UNIT
;        4056  !PRINTX (DER18);
;        4057  FSET_UPAR ();                                  ! FIND UNIT'S ENTRY IS CST AND SET UP UNIT-RELATED DATA
;        4058
;        4059  IF .RP_ADDR [STATUS] NEQU ST_SUC                        ! IF STATUS CODE INDICATES ERROR
;        4060  THEN    BEGIN
;        4061          CST_ADDR [.CUOFF + 3, DUPERROR] = 1;    ! SET DUP ERROR FLAG
;        4062          HARD_ERROR ();
;        4063          IF .RP_ADDR [ENDCOD] EQLU (OP_ELP + OP_END) OR  ! IF ENDCODE IS EXECUTE LOCAL PROGRAM
;        4064             .RP_ADDR [ENDCOD] EQLU (OP_GDS + OP_END)     !             OR GET DUST STATUS
;        4065          THEN    BEGIN
;        4066                  CST_ADDR [.CUOFF + 3, NODUPMEDIA] = 1;          ! TURN OFF DUP EXERCISR
;        4067                  END;
;        4068          END
;        4069
;        4070  ELSE                                            ! ELSE - I/O WAS SUCCESSFUL
;        4071
;        4072  BEGIN
;        4073
;        4074  IF .RP_ADDR [ENDCOD] EQLU (OP_GDS + OP_END)     ! IF ENDCODE IS GET DUST STATUS
;        4075  THEN    BEGIN
;        4076          IF .RP_ADDR [9,11,1,0] EQL 1
;        4077                  THEN CST_ADDR [.CUOFF + 3, D_ACTIVE] = ACTIVE   ! CONTROLLER IN AN ACTIVE STAE
;        4078                  ELSE CST_ADDR [.CUOFF + 3, D_ACTIVE] = IDLE;    ! CONTROLLER IN AN IDLE STATE
;        4079          IF .RP_ADDR [9,9,1,0] NEQ 1 THEN        ! TEST TO SEE IF CONTROLLER LOCAL PROGRAMS(PG 18 OF DUP DOC)
;        4080                  BEGIN
;        4081                  HARD_ERROR ();
;        4082                  CST_ADDR [.CUOFF + 3, NODUPMEDIA] = 1;          ! TURN OFF DUP EXERCISR
;        4083                  END;
```

```
;         4084          END;
;         4085
;         4086
;         4087  IF (.RP_ADDR [ENDCOD] EQL (OP_RCD + OP_END)) AND
;         4088     (.DUPPKT [DUPTYPE] EQL 6) AND
;         4089     (.DUPPKT [DUPMESG] EQL 2) AND                    !IF IT IS A RECEIVE DBN COMMAND WITH TYPE 6 AND MESSAGE 2 THEN
;         4090     (.CST_ADDR [.CUOFF+3, DUPWRITE] EQLU 1)          ! IF WRITE FLAG SET IN CST TABLE THEN COMPARE BLOCKS
;         4091          THEN DUP_COMPARE ();
;         4092
;         4093  END;                                                ! COMPARE THE FOLLOWING 512 BYTES
;         4094
;         4095  PUT_RETPKT (.RP_INDX);
;         4096  QIO [.CCTLR] = .QIO [.CCTLR] - 1;                   ! DECREMENT NO. OF OUTSTANDING QIOS
;         4097
;         4098  END;                                                ! ROUTINE DIO_RETPKT


                                        .SBTTL  DIO.RETPKT MULTI-DRIVE TEST ROUTINES
000000  010146                  DIO.RETPKT:
                                        MOV     R1,-(SP)                        ;                               4032
000002  112700  000001                  MOVB    #1,R0                           ; *,FLAG                        4051
000006  004737  000000V                 JSR     PC,FSET.UPAR                    ;                               4057
000012  013701  000000G                 MOV     RP.ADDR,R1                      ;                               4059
000016  005761  000016                  TST     16(R1)
000022  001435                          BEQ     2$
000024  013700  000000G                 MOV     CUOFF,R0                        ;                               4061
000030  006300                          ASL     R0
000032  063700  000000G                 ADD     CST.ADDR,R0
000036  052760  040000  000006         BIS     #40000,6(R0)
000044  004737  000000V                 JSR     PC,HARD.ERROR                   ;                               4062
000050  013700  000000G                 MOV     RP.ADDR,R0                      ;                               4063
000054  126027  000014  000203         CMPB    14(R0),#203
000062  001404                          BEQ     1$
000064  126027  000014  000201         CMPB    14(R0),#201                     ;                               4064
000072  001112                          BNE     6$
000074  013700  000000G         1$:     MOV     CUOFF,R0                        ;                               4066
000100  006300                          ASL     R0
000102  063700  000000G                 ADD     CST.ADDR,R0
000106  052760  100000  000006         BIS     #100000,6(R0)
000114  000501                          BR      6$                              ;                               4059
000116  126127  000014  000201 2$:     CMPB    14(R1),#201                     ;                               4074
000124  001036                          BNE     5$
000126  013700  000000G                 MOV     CUOFF,R0                        ;                               4077
000132  006300                          ASL     R0
000134  063700  000000G                 ADD     CST.ADDR,R0
000140  032761  004000  000022         BIT     #4000,22(R1)                    ;                               4076
000146  001404                          BEQ     3$
000150  052760  020000  000006         BIS     #20000,6(R0)                    ;                               4077
000156  000403                          BR      4$                              ;                               4076
000160  042760  020000  000006 3$:     BIC     #20000,6(R0)                    ;                               4078
000166  032761  001000  000022 4$:     BIT     #1000,22(R1)                    ;                               4079
000174  001012                          BNE     5$
000176  004737  000000V                 JSR     PC,HARD.ERROR                   ;                               4081
```

```
000202  013700  000000G                        MOV     CUOFF,R0                        ;                       4082
000206  006300                                  ASL     R0
000210  063700  000000G                        ADD     CST.ADDR,R0
000214  052760  100000  000006                 BIS     #100000,6(R0)
000222  013700  000000G                5$:     MOV     RP.ADDR,R0                      ;                       4087
000226  126027  000014  000205                 CMPB    14(R0),#205
000234  001031                                  BNE     6$
000236  013700  000000G                        MOV     DUPPKT,R0                       ;                       4088
000242  042700  007777                         BIC     #7777,R0
000246  020027  060000                         CMP     R0,#60000
000252  001022                                  BNE     6$
000254  013700  000000G                        MOV     DUPPKT,R0                       ;                       4089
000260  042700  170000                         BIC     #170000,R0
000264  020027  000002                         CMP     R0,#2
000270  001013                                  BNE     6$
000272  013700  000000G                        MOV     CUOFF,R0                        ;                       4090
000276  006300                                  ASL     R0
000300  063700  000000G                        ADD     CST.ADDR,R0
000304  032760  010000  000006                 BIT     #10000,6(R0)
000312  001402                                  BEQ     6$
000314  004737  000000V                        JSR     PC,DUP.COMPARE                  ;                       4091
000320  013746  000000G                6$:     MOV     RP.INDX,-(SP)                   ;                       4095
000324  004737  000000G                        JSR     PC,PUT.RETPKT
000330  013700  000000G                        MOV     CCTLR,R0                        ;                       4096
000334  105360  000000G                        DECB    QIO(R0)
000340  005726                                  TST     (SP)+                           ;                       4051
000342  012601                                  MOV     (SP)+,R1                        ;                       4032
000344  000207                                  RTS     PC

; Routine Size:  115 words,    Routine Base:  $CODE$ + 15416
; Maximum stack depth per invocation:  3 words

;       4099
```

```
;       4100  ROUTINE DUP_COMPARE : NOVALUE =
;       4101
;       4102
;       4103  !+
;       4104  !       THIS ROUTINE IS CALLED BY DIO_RETPKT WHEN THE RECEIVE DATA COMMAND
;       4105  !       IS BEING PROCESSED. THIS COMMAND COMPARES THE WRITTEN BUFFER WITH
;       4106  !       THE PATERN WORD GIVEN IN SEND DATA COMMAND.  FOR EVERY WORD COMPARED
;       4107  !       THE ROUTINE INCREMENTS THE TALLY TABLE.  IF THE COMPARE SHOWS AN
;       4108  !       ERROR. THE DBN HARD ERROR COUNTER WILL BE INCREMENTED AND THE
;       4109  !       THE DBN NUMBER AND BYTE COUNT WILL BE PRINTED.
;       4110  !
;       4111  !       IMPLICIT INPUTS:
;       4112  !               S_PATTERN       ! THE SAVED PATTERN WRITTEN TO THE DBN'S
;       4113  !               S_DUPPKT        ! THE POINTER FOR DUP BUFFER
;       4114  !               T_ADDR          ! THE ADDRESS OF THE TALLY TABLE FOR THIS UNIT
;       4115  !               CST_ADDR        ! THE ADDRESS OF PRESENT CONTROLLER STATUS TABLE
;       4116  !-
;       4117  BEGIN
;       4118
;       4119  OWN
;       4120          COUNT : WORD;
;       4121
;       4122  !PRINTX (DER19);
;       4123  S_DUPPKT = 0;
;       4124  INCR COUNT FROM 1 TO 256 DO     !INDEX PIONTER FOR DATA STORED IN MSCP ENV PACKET
;       4125          BEGIN
;       4126          S_DUPPKT = .S_DUPPKT + 2;               ! INITIALLY THIS SKIPS THE FIRST WORD OF DUPPKT
;       4127          IF .(DUPPKT + .S_DUPPKT) NEQ .S_PATTERN THEN    !IF THE CONTENTS OF DBN DOESN'T EQUAL PATTERN
;       4128                  BEGIN
;       4129                  ERRHRD (46, EH_10, EMS_22);     !LIST ERROR
;       4130                  EXITLOOP;
;       4131                  END;
;       4132          END;                                    !GO THROUGH ALL DBN WORDS
;       4133  END;                                            !END ROUTINE DUP-COMPARE


001254                                  .PSECT  $GGG$,  RO
001254                          COUNT:  .BLKW   1


                                        .SBTTL  DUP.COMPARE MULTI-DRIVE TEST ROUTINES
015764                                  .PSECT  $CODE$,  RO

000000  010146                  DUP.COMPARE:
                                        MOV     R1,-(SP)                        ;                               4100
000002  005037  000000G                 CLR     S.DUPPKT                        ;                               4123
000006  012701  000400                  MOV     #400.R1                         ; *,COUNT                       4124
000012  062737  000002  000000G 1$:     ADD     #2,S.DUPPKT                     ;                               4126
000020  013700  000000G                 MOV     S.DUPPKT,R0                     ;                               4127
000024  026037  000000G 000000G         CMP     DUPPKT(R0),S.PATTERN
000032  001405                          BEQ     2$
000034  104456                          TRAP    56                              ;                               4129
```

```
000036  000056                          .WORD   56
000040  000000G                         .WORD   EH.10
000042  000000G                         .WORD   EMS.22
000044  000402                          BR      3$                      ;                       4128
000046  005301                  2$:     DEC     R1                      ; COUNT                 4124
000050  001360                          BNE     1$
000052  012601                  3$:     MOV     (SP)+,R1                ;                       4100
000054  000207                          RTS     PC

; Routine Size:  23 words,      Routine Base:  $CODE$ + 15764
; Maximum stack depth per invocation:  3 words


;         4134
;         4135
;         4136
```

```
4137  routine IO_RETPKT : novalue =
4138
4139  !+
4140  !       THIS ROUTINE IS CALLED BY PROC_RETPKT TO HANDLE ALL I/O TRANSFER
4141  !       RETURN PACKETS. PROCESSING OF THESE PACKETS INCLUDES DECLARING ANY
4142  !       HARD ERRORS THAT MAY HAVE OCCURRED, UPDATING THE STATISTICS, AND
4143  !       PERFORMING HOST WRITE-COMPARES IF REQUIRED.
4144  !
4145  !       IMPLICIT INPUTS:
4146  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
4147  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
4148  !               T_ADDR - ADDRESS OF CURRENT UNIT'S STATISTICS BLOCK (TALLY)
4149  !               CCTLR - CURRENT CONTROLLER NUMBER
4150  !               L$LUN - CURRENT UNIT NUMBER
4151  !-
4152
4153      begin
4154
4155      local
4156          FLAG : byte initial (byte (TRUE));
4157
4158      FSET_UPAR ();                               ! FIND UNIT'S ENTRY IS CST AND SET UP UNIT-RELATED DATA
4159      ST_CODE = .RP_ADDR [STSCOD];                ! GET STATUS CODE FROM RETPKT
4160      SB_CODE = .RP_ADDR [SUBCOD];                ! GET SUB-CODE, IF ANY
4161
4162      if (.ST_CODE neq ST_SUC)                ! IF STATUS CODE INDICATES ERROR
4163      then
4164          begin
4165          HARD_ERROR ();                          ! UPDATE ERROR COUNT
4166          COMPARE_DATA = FALSE;                   ! NO POINT IN DOING HOST COMPARES ON ERROR
4167
4168          if (.ST_CODE neq ST_OFL) and            ! DROP UNIT IF ERROR COUNTS EXCEEDS LIMIT
4169             (.st_code neq ST_AVL) and
4170             (.T_ADDR [ERR_HARD] gequ .SWP_ERROR)
4171          then
4172              begin
4173              DUR [.L$LUN] = DU_HERR;             ! LOAD REASON FOR DROPPING UNIT
4174              DODU (.L$LUN);                      ! DROP UNIT
4175              end;
4176
4177          end;
4178
4179      if .ST_CODE eql ST_SUC                      ! IF I/O WAS SUCCESSFUL
4180      then
4181          begin
4182          UPD_IO_TALLY ();                        ! UPDATE I/O TALLY (STATISTICS)
4183
4184          if .RP_ADDR [ENDCOD] eql (OP_WRT or OP_END)
4185          then COMPARE_DATA = TRUE;               ! HOST COMPARES MAY BE ALLOWED IF NO FURTHER ERRORS
4186
4187          if (BIT_TST (SWP_FLAGS, SWF_HWC)) and   ! IF HOST IS DOING WRITE-COMPARES
4188             (.COMPARE_DATA)
4189          then
```

```
;         4190              FLAG = HOST_WRT_CHK ();          ! SAVE I/O PACKET OR DO WRITE-CHECK
;         4191
;         4192          end;
;         4193
;         4194      if .FLAG                                 ! IF FLAG IS STILL TRUE
;         4195      then
;         4196          SWEEP ();                            ! DEALLOCATE BUFFER(S) AND RETPKT(S)
;         4197
;         4198      QIO [.CCTLR] = .QIO [.CCTLR] - 1;        ! DECREMENT NO. OF OUTSTANDING QIOS
;         4199      end;                                     ! ROUTINE IO_RETPKT


                                        .SBTTL  IO.RETPKT MULTI-DRIVE TEST ROUTINES
000000  004137  000000G          IO.RETPKT:
                                        JSR     R1,$SAVE2                       ;                       4137
000004  112701  000001                  MOVB    #1,R1                           ; *,FLAG                4153
000010  004737  000000V                 JSR     PC,FSET.UPAR                    ;                       4158
000014  013700  000000G                 MOV     RP.ADDR,R0                      ;                       4159
000020  116037  000016  000000G         MOVB    16(R0),ST.CODE
000026  042737  177740  000000G         BIC     #177740,ST.CODE
000034  016002  000016                  MOV     16(R0),R2                       ;                       4160
000040  006202                          ASR     R2
000042  006202                          ASR     R2
000044  006202                          ASR     R2
000046  006202                          ASR     R2
000050  006202                          ASR     R2
000052  042702  174000                  BIC     #174000,R2
000056  010237  000000G                 MOV     R2,SB.CODE
000062  005737  000000G                 TST     ST.CODE                         ;                       4162
000066  001433                          BEQ     2$
000070  004737  000000V                 JSR     PC,HARD.ERROR                   ;                       4165
000074  105037  001240'                 CLRB    COMPARE.DATA                    ;                       4166
000100  023727  000000G 000003         CMP     ST.CODE,#3                      ;                       4168
000106  001420                          BEQ     1$
000110  023727  000000G 000004         CMP     ST.CODE,#4                      ;                       4169
000116  001414                          BEQ     1$
000120  013700  000000G                 MOV     T.ADDR,R0                       ;                       4170
000124  026037  000014  000000G         CMP     14(R0),SWP.ERROR
000132  103406                          BLO     1$
000134  013700  000000G                 MOV     L$LUN,R0                        ;                       4173
000140  112760  000004  000000G         MOVB    #4,DUR(R0)
000146  104451                          TRAP    51                              ;                       4174
000150  005737  000000G          1$:    TST     ST.CODE                         ;                       4179
000154  001026                          BNE     4$
000156  004737  000000V          2$:    JSR     PC,UPD.IO.TALLY                 ;                       4182
000162  013700  000000G                 MOV     RP.ADDR,R0                      ;                       4184
000166  126027  000014  000242         CMPB    14(R0),#242
000174  001003                          BNE     3$
000176  112737  000001  001240'         MOVB    #1,COMPARE.DATA                 ;                       4185
000204  032737  000040  000000G  3$:    BIT     #40,SWP.FLAGS                   ;                       4187
000212  001407                          BEQ     4$
000214  032737  000001  001240'         BIT     #1,COMPARE.DATA                 ;                       4188
000222  001403                          BEQ     4$
```

```
000224  004737  000000V                        JSR     PC,HOST.WRT.CHK                 ;                                       4190
000230  110001                                  MOVB    R0,R1                           ; *,FLAG
000232  006001                          4$:     ROR     R1                              ; FLAG                                  4194
000234  103002                                  BCC     5$
000236  004737  000000V                        JSR     PC,SWEEP                        ;                                       4196
000242  013700  000000G                5$:     MOV     CCTLR,R0                        ;                                       4198
000246  105360  000000G                        DECB    QIO(R0)
000252  000207                                  RTS     PC                              ;                                       4137

; Routine Size:  86 words,      Routine Base:  $CODE$ + 16042
; Maximum stack depth per invocation:  5 words
```

```
;        4200  routine FSET_UPAR : novalue =
;        4201
;        4202  !+
;        4203  !       THIS ROUTINE IS CALLED BY IO_RETPKT AND OTHERS TO SEARCH THE CURRENT
;        4204  !       CONTROLLER STATUS TABLE (CST) FOR THE DISK ADDRESS WHICH IS
;        4205  !       CONTAINED IN THE CURRENT RETURN PACKET. WHEN FOUND, THE OFFSET INTO THE
;        4206  !       CST IS USED AS INPUT TO SET_UPAR, WHICH SETS UP CURRENT UNIT-RELATED
;        4207  !       DATA PARAMETERS.
;        4208  !
;        4209  !       IMPLICIT INPUTS:
;        4210  !               RP_ADDR - ADDRESS OF CURRENT RETURN PACKET
;        4211  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;        4212  !-
;        4213
;        4214      begin
;        4215
;        4216      incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do     ! FOR EACH UNIT IN CST
;        4217
;        4218          if .CST_ADDR [.OFFSET, D_DISK_NUM] eql .RP_ADDR [DISK]
;        4219                                                  ! IF RETPKT UNIT NUMBER MATCHES CST ENTRY
;        4220          then
;        4221              begin
;        4222              SET_UPAR (.OFFSET);                 ! SET UP UNIT-RELATED DATA
;        4223              return;                             ! DONE
;        4224              end;
;        4225
;        4226      PRINTF (DBM19, .RP_ADDR [DISK], .CCTLR);    ! "CAN'T FIND DISK XXX IN CST X"
;        4227      end;                                        ! ROUTINE FSET_UPAR


                        .SBTTL  FSET.UPAR MULTI-DRIVE TEST ROUTINES
000000  004137  000000G FSET.UPAR:
                        JSR     R1,$SAVE4                       ;                       4200
000004  012702  000003          MOV     #3,R2                   ; *,OFFSET              4216
000010  010201          1$:     MOV     R2,R1                   ; OFFSET,*              4218
000012  006301                  ASL     R1
000014  063701  000000G         ADD     CST.ADDR,R1
000020  013700  000000G         MOV     RP.ADDR,R0
000024  016004  000010          MOV     10(R0),R4
000030  111103                  MOVB    (R1),R3
000032  042703  177774          BIC     #177774,R3
000036  020304                  CMP     R3,R4
000040  001005                  BNE     2$
000042  010246                  MOV     R2,-(SP)                ; OFFSET,*              4222
000044  004737  000000G         JSR     PC,SET.UPAR
000050  005726                  TST     (SP)+                   ;                       4218
000052  000207                  RTS     PC                      ;                       4221
000054  062702  000005  2$:     ADD     #5,R2                   ; *,OFFSET              4216
000060  020227  000022          CMP     R2,#22                  ; OFFSET,*
000064  003751                  BLE     1$
000066  013746  000000G         MOV     CCTLR,-(SP)             ;                       4226
000072  013700  000000G         MOV     RP.ADDR,R0
000076  016046  000010          MOV     10(R0),-(SP)
```

```
000102  012746  000000G                        MOV     #DBM19,-(SP)
000106  012746  000003                         MOV     #3,-(SP)
000112  010600                                 MOV     SP,R0                   ; SP,*
000114  104417                                 TRAP    17
000116  062706  000010                         ADD     #10,SP                  ;                                       4214
000122  000207                                 RTS     PC                      ;                                       4200

; Routine Size:  42 words,      Routine Base:  $CODE$ + 16316
; Maximum stack depth per invocation:  11 words
```

```
;        4228  routine HARD_ERROR : novalue =
;        4229
;        4230  !+
;        4231  !       THIS ROUTINE IS CALLED BY IO_RETPKT, DIO_RETPKT, AND OTHERS TO INCREMENT THE HARD
;        4232  !       ERROR STATISTIC FIELD FOR THE CURRENT UNIT. IF THE HARD ERROR COUNT
;        4233  !       HAS EXCEEDED THE OPERATOR-SPECIFIED LIMIT, THEN THE UNIT IS DROPPED
;        4234  !       FROM TESTING.
;        4235  !
;        4236  !       IMPLICIT INPUTS:
;        4237  !               L$LUN - CURRENT UNIT NUMBER
;        4238  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;        4239  !               CUOFF - CST OFFSET FOR CURRENT UNIT
;        4240  !               T_ADDR - ADDRESS OF CURRENT UNIT'S STATISTICS BLOCK (TALLY)
;        4241  !-
;        4242
;        4243      begin
;        4244      T_ADDR [ERR_HARD] = .T_ADDR [ERR_HARD] + 1;                     ! INCREMENT UNIT'S HARD ERROR COUNT
;        4245  if .RP_ADDR [CONID] EQL CID_MSCP
;        4246  THEN
;        4247      selectoneu .ST_CODE of
;        4248          set
;        4249
;        4250          [ST_SUC]:       if .SB_CODE neq 0                           ! SUCCESS WITH NON-ZERO SUB-CODE
;        4251                          then
;        4252                              begin
;        4253
;        4254                              if .SB_CODE eql 4
;        4255                              then
;        4256                                  begin
;        4257                                  T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1
;        4258                                  end
;        4259                              else
;        4260                                  T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;        4261
;        4262                              ERRHRD (44, EGH_30, EMS_30);
;        4263                              end;
;        4264
;        4265          [ST_CMD]:       begin                                       ! INVALID COMMAND
;        4266                          T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;        4267                          ERRHRD (31, EGH_30, EMS_30);
;        4268                          end;
;        4269
;        4270          [ST_ABO]:       begin                                       ! COMMMAND ABORTED
;        4271                          T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4272                          ERRHRD (32, EGH_30, EMS_30);
;        4273                          end;
;        4274
;        4275          [ST_OFL] :      begin                                       ! OFFLINE
;        4276                          T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4277                          ERRDF (18, EGD_T8, EMS_18);
;        4278                          DUR [.L$LUN] = DU_DFATAL;                   ! DEVICE FATAL ERROR
;        4279                          DODU (.L$LUN);                              ! DROP UNIT
;        4280                          end;
```

```
;        4281
;        4282          [ST_AVL] :      begin
;        4283                          T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4284                          ERRDF (24, EGD_T8, EMS_18);
;        4285                          DUR [.L$LUN] = DU_AV;                           ! DEVICE WENT AVAILABLE STATE
;        4286                          DODU (.L$LUN);                                  ! DROP UNIT
;        4287                          end;
;        4288
;        4289          [ST_MFE]:       begin                                           ! MEDIA FORMAT ERROR
;        4290                          T_ADDR [ERR_HRD_SEK] = .T_ADDR [ERR_HRD_SEK] + 1;
;        4291                          ERRHRD (35, EGH_30, EMS_30);
;        4292                          end;
;        4293
;        4294          [ST_WPT]:       begin                                           ! DEVICE WRITE PROTECTED
;        4295
;        4296                          if .SB_CODE eql 128
;        4297                          then
;        4298                              T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1
;        4299                          else
;        4300                              T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4301
;        4302                          ERRHRD (36, EGH_30, EMS_30);
;        4303                          end;
;        4304
;        4305          [ST_CMP]:       begin                                           ! COMPARE ERROR
;        4306                          T_ADDR [ERR_HRD_DAT] = .T_ADDR [ERR_HRD_DAT] + 1;
;        4307                          ERRHRD (37, EGH_30, EMS_30);
;        4308                          end;
;        4309
;        4310          [ST_DAT]:       begin                                           ! DATA ERROR
;        4311
;        4312                          if .SB_CODE eql 2
;        4313                          then
;        4314                              T_ADDR [ERR_HRD_SEK] = .T_ADDR [ERR_HRD_SEK] + 1
;        4315                          else
;        4316                              T_ADDR [ERR_HRD_DAT] = .T_ADDR [ERR_HRD_DAT] + 1;
;        4317
;        4318                          ERRHRD (38, EGH_30, EMS_30);
;        4319                          end;
;        4320
;        4321          [ST_HST]:       begin                                           ! HOST ACCESS ERROR
;        4322                          T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;        4323                          ERRHRD (39, EGH_30, EMS_30);
;        4324                          end;
;        4325
;        4326          [ST_CNT]:       begin                                           ! CONTROLLER ERROR
;        4327                          T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4328                          ERRHRD (40, EGH_30, EMS_30);
;        4329                          end;
;        4330
;        4331          [ST_DRV]:       begin                                           ! DRIVE ERROR
;        4332
;        4333                          if .SB_CODE eql 3
```

```
;       4334                            then
;       4335                                T_ADDR [ERR_HRD_SEK] = .T_ADDR [ERR_HRD_SEK] + 1
;       4336                            else
;       4337                                T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;       4338
;       4339                            ERRHRD (41, EGH_30, EMS_30);
;       4340                            end;
;       4341
;       4342                [ST_DIA]:   begin                                   ! MESSAGE FROM INTERNAL DIAGNOSTICS
;       4343                            T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;       4344                            ERRHRD (43, EGH_30, EMS_30);
;       4345                            end;
;       4346
;       4347                [otherwise]: begin                                  ! PRINT STATUS CODE IF NO MATCH
;       4348                            C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;       4349                            ERRHRD (45, EGH_30, EMS_30);
;       4350                            end;
;       4351
;       4352                tes;
;       4353        if .RP_ADDR [CONID] EQL CID_DUP
;       4354        THEN
;       4355
;       4356            selectoneu .RP_ADDR [STSCOD] of
;       4357                    SET
;       4358                    [%o'0']         : begin                     ! if status code succesful
;       4359                        IF .RP_ADDR [ENDCOD] EQLU (OP_GDS + OP_END)     ! IF ENDCODE IS GET DUST STATUS
;       4360                        THEN
;       4361                            IF .RP_ADDR [9,9,1,0] NEQ 1 THEN        ! TEST TO SEE IF CONTROLLER LOCAL PROGRAMS(PG 18 OF
;       4362                                BEGIN
;       4363                                ERRHRD (47, EH_12, EMS_30);     ! UNABLE TO LOAD LOCAL CONTROLLER DUP MEDIA(PROGRAMS
;       4364                                T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;       4365                                END;
;       4366                    if .duppkt [dupmesg] eql 3
;       4367                    then
;       4368                            if (.DUPPKT [DUPTYPE] eql 5)
;       4369                            then
;       4370                                    begin
;       4371                                    DUR [.L$LUN] = DU_DFATAL;
;       4372                                    DODU (.L$LUN);                  ! FATAL DEVICE ERROR DROP UNIT);
;       4373                                                                    ! SET REASON FOR DROPPING UNIT
;       4374                                    selectoneu .DUPPKT [DUPMESG] of
;       4375                                            SET
;       4376                                            [%o'1'] : begin
;       4377                                                    errhrd (48, eh_13, ems_30); ! illegal unit number
;       4378                                                    T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;       4379                                                    end;
;       4380                                            [%o'2'] : begin
;       4381                                                    errhrd (49, eh_13, ems_30); ! illegal relative or physical block #
;       4382                                                    T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;       4383                                                    end;
;       4384                                            [%o'3'] : begin
;       4385                                                    errhrd (50, eh_13, ems_30); ! device error
;       4386                                                    T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
```

```
;        4387                                                    end;
;        4388                                            [%o'4'] : begin
;        4389                                                    errhrd (51, eh_13, ems_30); ! zero lenght message
;        4390                                                    T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;        4391                                                    end;
;        4392                                            tes;
;        4393                                        end;
;        4394                                end;
;        4395                        [%o'1']     : begin
;        4396                                ERRHRD (52, EH_7, EMS_30);      ! INVALID COMMAND
;        4397                                T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4398                                end;
;        4399                        [%o'2']     : begin
;        4400                                ERRHRD (53, EH_7, EMS_30);      ! NO REGION AVAILABLE
;        4401                                T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4402                                end;
;        4403                        [%o'3']     : begin
;        4404                                ERRHRD (54, EH_7, EMS_30);      ! NO REGION SUITABLE
;        4405                                T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;        4406                                end;
;        4407                        [%o'4']     : begin
;        4408                                ERRHRD (55, EH_7, EMS_30);      ! PROGRAM NOT KNOWN
;        4409                                T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;        4410                                end;
;        4411                        [%o'5']     : begin
;        4412                                ERRHRD (56, EH_7, EMS_30);      ! LOAD FAILURE
;        4413                                T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4414                                end;
;        4415                        [%o'6']     : begin
;        4416                                ERRHRD (57, EH_7, EMS_30);      ! STANDALONE
;        4417                                T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;        4418                                end;
;        4419                        [OTHERWISE] : begin
;        4420                                ERRHRD (58, EH_8, EMS_30);      ! DUP UNKNOWN STATUS CODE
;        4421                                C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;        4422                                end;
;        4423                        TES;
;        4424
;        4425        end;                                                    ! ROUTINE HARD_ERROR


                                            .SBTTL  HARD.ERROR MULTI-DRIVE TEST ROUTINES
000000  004137  000000G                    HARD.ERROR:
                                                JSR     R1,$SAVE3                       ;                       4228
000004  013701  000000G                             MOV     T.ADDR,R1                       ;                       4244
000010  005261  000014                              INC     14(R1)
000014  013700  000000G                             MOV     RP.ADDR,R0                      ;                       4245
000020  105760  000003                              TSTB    3(R0)
000024  001157                                      BNE     12$
000026  013700  000000G                             MOV     ST.CODE,R0                      ;                       4247
000032  001022                                      BNE     3$
000034  013702  000000G                             MOV     SB.CODE,R2                      ;                       4250
000040  001574                                      BEQ     16$
```

```
000042  012703  000062                      MOV     #62,R3                          ;                       4257
000046  060103                              ADD     R1,R3
000050  020227  000004                      CMP     R2,#4                           ;                       4254
000054  001002                              BNE     1$
000056  105213                              INCB    (R3)                            ;                       4257
000060  000402                              BR      2$                              ;                       4254
000062  105263  000001              1$:     INCB    1(R3)                           ;                       4260
000066  104456                      2$:     TRAP    56                              ;                       4262
000070  000054                              .WORD   54
000072  000000G                             .WORD   EGH.30
000074  000000G                             .WORD   EMS.30
000076  000567                              BR      18$                             ;                       4247
000100  020027  000001              3$:     CMP     R0,#1
000104  001007                              BNE     4$
000106  105261  000063                      INCB    63(R1)                          ;                       4266
000112  104456                              TRAP    56                              ;                       4267
000114  000037                              .WORD   37
000116  000000G                             .WORD   EGH.30
000120  000000G                             .WORD   EMS.30
000122  000567                              BR      20$                             ;                       4247
000124  020027  000002              4$:     CMP     R0,#2
000130  001007                              BNE     5$
000132  105261  000062                      INCB    62(R1)                          ;                       4271
000136  104456                              TRAP    56                              ;                       4272
000140  000040                              .WORD   40
000142  000000G                             .WORD   EGH.30
000144  000000G                             .WORD   EMS.30
000146  000576                              BR      24$                             ;                       4247
000150  020027  000003              5$:     CMP     R0,#3
000154  001015                              BNE     6$
000156  105261  000062                      INCB    62(R1)                          ;                       4276
000162  104455                              TRAP    55                              ;                       4277
000164  000022                              .WORD   22
000166  000000G                             .WORD   EGD.18
000170  000000G                             .WORD   EMS.18
000172  013700  000000G                     MOV     L$LUN,R0                        ;                       4278
000176  112760  000005  000000G             MOVB    #5,DUR(R0)
000204  104451                              TRAP    51                              ;                       4279
000206  000570                              BR      26$                             ;                       4247
000210  020027  000004              6$:     CMP     R0,#4
000214  001015                              BNE     7$
000216  105261  000062                      INCB    62(R1)                          ;                       4283
000222  104455                              TRAP    55                              ;                       4284
000224  000030                              .WORD   30
000226  000000G                             .WORD   EGD.18
000230  000000G                             .WORD   EMS.18
000232  013700  000000G                     MOV     L$LUN,R0                        ;                       4285
000236  112760  000013  000000G             MOVB    #13,DUR(R0)
000244  104451                              TRAP    51                              ;                       4286
000246  000562                              BR      28$                             ;                       4247
000250  020027  000005              7$:     CMP     R0,#5
000254  001007                              BNE     8$
000256  105261  000060                      INCB    60(R1)                          ;                       4290
```

```
000262  104456                          TRAP    56                              ;                               4291
000264  000043                          .WORD   43
000266  000000G                         .WORD   EGH.30
000270  000000G                         .WORD   EMS.30
000272  000550                          BR      28$                             ;                               4247
000274  020027  000006          8$:     CMP     R0,#6
000300  001020                          BNE     11$
000302  012702  000062                  MOV     #62,R2                          ;                               4298
000306  060102                          ADD     R1,R2
000310  023727  000000G 000200          CMP     SB.CODE,#200                    ;                               4296
000316  001003                          BNE     9$
000320  105262  000001                  INCB    1(R2)                           ;                               4298
000324  000401                          BR      10$                             ;                               4296
000326  105212                  9$:     INCB    (R2)                            ;                               4300
000330  104456                  10$:    TRAP    56                              ;                               4302
000332  000044                          .WORD   44
000334  000000G                         .WORD   EGH.30
000336  000000G                         .WORD   EMS.30
000340  000525                          BR      28$                             ;                               4247
000342  020027  000007          11$:    CMP     R0,#7
000346  001007                          BNE     13$
000350  105261  000061                  INCB    61(R1)                          ;                               4306
000354  104456                          TRAP    56                              ;                               4307
000356  000045                          .WORD   45
000360  000000G                         .WORD   EGH.30
000362  000000G                         .WORD   EMS.30
000364  000513                  12$:    BR      28$                             ;                               4247
000366  020027  000010          13$:    CMP     R0,#10
000372  001020                          BNE     17$
000374  012702  000060                  MOV     #60,R2                          ;                               4314
000400  060102                          ADD     R1,R2
000402  023727  000000G 000002          CMP     SB.CODE,#2                      ;                               4312
000410  001002                          BNE     14$
000412  105212                          INCB    (R2)                            ;                               4314
000414  000402                          BR      15$                             ;                               4312
000416  105262  000001          14$:    INCB    1(R2)                           ;                               4316
000422  104456                  15$:    TRAP    56                              ;                               4318
000424  000046                          .WORD   46
000426  000000G                         .WORD   EGH.30
000430  000000G                         .WORD   EMS.30
000432  000470                  16$:    BR      28$                             ;                               4247
000434  020027  000011          17$:    CMP     R0,#11
000440  001007                          BNE     19$
000442  105261  000063                  INCB    63(R1)                          ;                               4322
000446  104456                          TRAP    56                              ;                               4323
000450  000047                          .WORD   47
000452  000000G                         .WORD   EGH.30
000454  000000G                         .WORD   EMS.30
000456  000456                  18$:    BR      28$                             ;                               4247
000460  020027  000012          19$:    CMP     R0,#12
000464  001007                          BNE     21$
000466  105261  000062                  INCB    62(R1)                          ;                               4327
000472  104456                          TRAP    56                              ;                               4328
```

```
000474  000050                          .WORD   50
000476  000000G                         .WORD   EGH.30
000500  000000G                         .WORD   EMS.30
000502  000444                  20$:    BR      28$                             ;                               4247
000504  020027  000013          21$:    CMP     R0,#13
000510  001016                          BNE     25$
000512  023727  000000G 000003          CMP     SB.CODE,#3                      ;                               4333
000520  001003                          BNE     22$
000522  105261  000060                  INCB    60(R1)                          ;                               4335
000526  000402                          BR      23$                             ;                               4333
000530  105261  000062          22$:    INCB    62(R1)                          ;                               4337
000534  104456                  23$:    TRAP    56                              ;                               4339
000536  000051                          .WORD   51
000540  000000G                         .WORD   EGH.30
000542  000000G                         .WORD   EMS.30
000544  000423                  24$:    BR      28$                             ;                               4247
000546  020027  000037          25$:    CMP     R0,#37
000552  001007                          BNE     27$
000554  105261  000062                  INCB    62(R1)                          ;                               4343
000560  104456                          TRAP    56                              ;                               4344
000562  000053                          .WORD   53
000564  000000G                         .WORD   EGH.30
000566  000000G                         .WORD   EMS.30
000570  000411                  26$:    BR      28$                             ;                               4247
000572  013700  000000G         27$:    MOV     CCTLR,R0                        ;                               4348
000576  006300                          ASL     R0
000600  105260  000000G                 INCB    C.ERR.TBL(R0)
000604  104456                          TRAP    56                              ;                               4349
000606  000055                          .WORD   55
000610  000000G                         .WORD   EGH.30
000612  000000G                         .WORD   EMS.30
000614  013700  000000G         28$:    MOV     RP.ADDR,R0                      ;                               4353
000620  126027  000003  000002          CMPB    3(R0),#2
000626  001107                          BNE     33$
000630  116001  000016                  MOVB    16(R0),R1                       ;                               4356
000634  042701  177740                  BIC     #177740,R1
000640  001110                          BNE     34$
000642  126027  000014  000201          CMPB    14(R0),#201                     ;                               4359
000650  001014                          BNE     29$
000652  032760  001000  000022          BIT     #1000,22(R0)                    ;                               4361
000660  001010                          BNE     29$
000662  104456                          TRAP    56                              ;                               4363
000664  000057                          .WORD   57
000666  000000G                         .WORD   EH.12
000670  000000G                         .WORD   EMS.30
000672  013700  000000G                 MOV     T.ADDR,R0                       ;                               4364
000676  105260  000062                  INCB    62(R0)
000702  013700  000000G         29$:    MOV     DUPPKT,R0                       ;                               4366
000706  042700  170000                  BIC     #170000,R0
000712  020027  000003                  CMP     R0,#3
000716  001162                          BNE     43$
000720  013700  000000G                 MOV     DUPPKT,R0                       ;                               4368
000724  042700  007777                  BIC     #7777,R0
```

```
000730  020027  050000                      CMP     R0,#50000
000734  001153                              BNE     43$
000736  013700  000000G                     MOV     L$LUN,R0                ;                       4371
000742  112760  000005  000000G             MOVB    #5,DUR(R0)
000750  104451                              TRAP    51                      ;                       4372
000752  013700  000000G                     MOV     DUPPKT,R0               ;                       4374
000756  042700  170000                      BIC     #170000,R0
000762  020027  000001                      CMP     R0,#1
000766  001005                              BNE     30$
000770  104456                              TRAP    56                      ;                       4377
000772  000060                              .WORD   60
000774  000000G                             .WORD   EH.13
000776  000000G                             .WORD   EMS.30
001000  000513                              BR      41$                     ;                       4378
001002  020027  000002          30$:        CMP     R0,#2                   ;                       4374
001006  001005                              BNE     31$
001010  104456                              TRAP    56                      ;                       4381
001012  000061                              .WORD   61
001014  000000G                             .WORD   EH.13
001016  000000G                             .WORD   EMS.30
001020  000503                              BR      41$                     ;                       4382
001022  020027  000003          31$:        CMP     R0,#3                   ;                       4374
001026  001005                              BNE     32$
001030  104456                              TRAP    56                      ;                       4385
001032  000062                              .WORD   62
001034  000000G                             .WORD   EH.13
001036  000000G                             .WORD   EMS.30
001040  000457                              BR      39$                     ;                       4386
001042  020027  000004          32$:        CMP     R0,#4                   ;                       4374
001046  001106                  33$:        BNE     43$
001050  104456                              TRAP    56                      ;                       4389
001052  000063                              .WORD   63
001054  000000G                             .WORD   EH.13
001056  000000G                             .WORD   EMS.30
001060  000463                              BR      41$                     ;                       4390
001062  020127  000001          34$:        CMP     R1,#1                   ;                       4356
001066  001005                              BNE     35$
001070  104456                              TRAP    56                      ;                       4396
001072  000064                              .WORD   64
001074  000000G                             .WORD   EH.7
001076  000000G                             .WORD   EMS.30
001100  000437                              BR      39$                     ;                       4397
001102  020127  000002          35$:        CMP     R1,#2                   ;                       4356
001106  001005                              BNE     36$
001110  104456                              TRAP    56                      ;                       4400
001112  000065                              .WORD   65
001114  000000G                             .WORD   EH.7
001116  000000G                             .WORD   EMS.30
001120  000427                              BR      39$                     ;                       4401
001122  020127  000003          36$:        CMP     R1,#3                   ;                       4356
001126  001005                              BNE     37$
001130  104456                              TRAP    56                      ;                       4404
001132  000066                              .WORD   66
```

```
001134  000000G                         .WORD   EH.7
001136  000000G                         .WORD   EMS.30
001140  000433                          BR      41$                             ;4405
001142  020127  000004          37$:    CMP     R1,#4                           ;4356
001146  001005                          BNE     38$
001150  104456                          TRAP    56                              ;4408
001152  000067                          .WORD   67
001154  000000G                         .WORD   EH.7
001156  000000G                         .WORD   EMS.30
001160  000423                          BR      41$                             ;4409
001162  020127  000005          38$:    CMP     R1,#5                           ;4356
001166  001011                          BNE     40$
001170  104456                          TRAP    56                              ;4412
001172  000070                          .WORD   70
001174  000000G                         .WORD   EH.7
001176  000000G                         .WORD   EMS.30
001200  013700  000000G         39$:    MOV     T.ADDR,R0                       ;4413
001204  105260  000062                  INCB    62(R0)
001210  000207                          RTS     PC                              ;4356
001212  020127  000006          40$:    CMP     R1,#6
001216  001011                          BNE     42$
001220  104456                          TRAP    56                              ;4416
001222  000071                          .WORD   71
001224  000000G                         .WORD   EH.7
001226  000000G                         .WORD   EMS.30
001230  013700  000000G         41$:    MOV     T.ADDR,R0                       ;4417
001234  105260  000063                  INCB    63(R0)
001240  000207                          RTS     PC                              ;4356
001242  104456                  42$:    TRAP    56                              ;4420
001244  000072                          .WORD   72
001246  000000G                         .WORD   EH.8
001250  000000G                         .WORD   EMS.30
001252  013700  000000G                 MOV     CCTLR,R0                        ;4421
001256  006300                          ASL     R0
001260  105260  000000G                 INCB    C.ERR.TBL(R0)
001264  000207                  43$:    RTS     PC                              ;4228

; Routine Size:  347 words,    Routine Base:  $CODE$ + 16442
; Maximum stack depth per invocation:  6 words
```

```
;        4426  routine UPD_IO_TALLY : novalue =
;        4427
;        4428  !+
;        4429  !       THIS ROUTINE IS CALLED FROM IO_RETPKT FOR ALL I/O TRANSFER RETURN
;        4430  !       PACKETS WITH "SUCCESS" STATUS CODES. ITS PURPOSE IS TO UPDATE ALL THE
;        4431  !       APPROPRIATE STATISTICAL FIELDS FOR THE CURRENT UNIT. A CHECK IS ALSO
;        4432  !       MADE ON THE TOTAL NUMBER OF BYTES TRANSFERRED THUS FAR; IF THE
;        4433  !       OPERATOR-SPECIFIED LIMIT HAS BEEN REACHED, THEN THE UNIT IS DROPPED.
;        4434  !
;        4435  !       IMPLICIT INPUTS:
;        4436  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;        4437  !               T_ADDR - ADDRESS OF THE CURRENT UNIT'S STATISTICS BLOCK (TALLY)
;        4438  !               CST_ADDR - ADDRESS OF THE CURRENT CONTROLLER'S CST
;        4439  !               CUOFF - CST OFFSET FOR THE CURRENT UNIT
;        4440  !               LSLUN - CURRENT UNIT NUMBER
;        4441  !-
;        4442
;        4443      begin
;        4444
;        4445      local
;        4446          THOUSANDS : word,
;        4447          MILLIONS : word;                                    ! TOTAL NUMBER OF BYTES TRANSFERRED TO/FROM A UNIT
;        4448
;        4449      if .RP_ADDR [ENDCOD] eql (OP_RD or OP_END)              ! IF ENDCODE IS READ
;        4450      then
;        4451          begin
;        4452          T_ADDR [TOT_READS_LO] = .T_ADDR [TOT_READS_LO] + 1;     ! INCREMENT NO. OF READS AND ADD BYTE COUNT
;        4453          T_ADDR [BYTES_READ_LO] = .T_ADDR [BYTES_READ_LO] + .RP_ADDR [BCNT_LO];
;        4454          T_ADDR [TOT_BYT_RED_LO] = .T_ADDR [TOT_BYT_RED_LO] + .RP_ADDR [BCNT_LO];
;        4455          OVF_CHK (T_ADDR [TOT_READS_LO]);                    ! CHECK FOR FIELD OVERFLOW
;        4456          OVF_CHK (T_ADDR [BYTES_READ_LO]);                   !
;        4457          OVF_CHK (T_ADDR [TOT_BYT_RED_LO]);                  !
;        4458          end
;        4459      else
;        4460
;        4461          if .RP_ADDR [ENDCOD] eql (OP_WRT or OP_END)         ! IF ENDCODE IS WRITE
;        4462          then
;        4463              begin
;        4464              T_ADDR [TOT_WRITES_LO] = .T_ADDR [TOT_WRITES_LO] + 1;       ! INCREMENT NO. OF WRITES AND ADD BYTE COUNT
;        4465              T_ADDR [BYTES_WRIT_LO] = .T_ADDR [BYTES_WRIT_LO] + .RP_ADDR [BCNT_LO];
;        4466              T_ADDR [TOT_BYT_WRT_LO] = .T_ADDR [TOT_BYT_WRT_LO] + .RP_ADDR [BCNT_LO];
;        4467              OVF_CHK (T_ADDR [TOT_WRITES_LO]);               ! CHECK FOR FIELD OVERFLOW
;        4468              OVF_CHK (T_ADDR [BYTES_WRIT_LO]);               !
;        4469              OVF_CHK (T_ADDR [TOT_BYT_WRT_LO]);              !
;        4470              end;
;        4471
;        4472      if (.RP_ADDR [ENDCOD] eql (OP_RD or OP_END)) or
;        4473         (.RP_ADDR [ENDCOD] eql (OP_WRT or OP_END))
;        4474      then
;        4475          begin
;        4476          MILLIONS = .T_ADDR [MBYTES_READ] + .T_ADDR [MBYTES_WRIT];       ! TOTAL BYTES TRANSFERRED
;        4477          THOUSANDS = .T_ADDR [BYTES_READ_HI] + .T_ADDR [BYTES_WRIT_HI];
;        4478
```

```
;        4479          if .THOUSANDS gequ 1000
;        4480          then
;        4481              begin
;        4482              MILLIONS = .MILLIONS + 1;                          ! COUNT THE LOWER OVERFLOW TOO!
;        4483              THOUSANDS = .THOUSANDS - 1000;
;        4484              end;
;        4485
;        4486      !
;        4487      ! THIS ADDED BECAUSE IT WILL TAKE FOREVER TO TRANSFER ON THE ORDER OF A MEGABYTE TO A FLOPPY
;        4488      ! BUT IT IS A MUCH MORE REASONABLE MEASURE FOR THE RD51 WINCHESTER.  THE QUESTION NOW REFERS TO
;        4489      ! THE TOTAL DATA TRANSFER TO THE CONTROLLER AND THIS IS PRETTY CLOSE SINCE THE FLOPPIES GET
;        4490      ! ABOUT 1/1000 THE DATA THE HARD DISK(S) GET.
;        4491      !
;        4492
;        4493          if .SWP_XFER eql 0                                     ! IF THERE IS A TRANSFER LIMIT
;        4494          then
;        4495              begin
;        4496
;        4497              if .THOUSANDS gtru 100
;        4498              then
;        4499                  begin
;        4500                  EOP_FLAG = TRUE;                               ! SET END-OF-PASS FLAG
;        4501                  DCT_ADDR [IG_INT] = TRUE;                      ! IGNORE FURTHER INTERRUPTS
;        4502                  end;
;        4503
;        4504              end
;        4505          else
;        4506
;        4507              if .MILLIONS gequ .SWP_XFER                        ! IF TRANSFER LIMIT IS REACHED
;        4508              then
;        4509                  begin
;        4510                  EOP_FLAG = TRUE;                               ! SET END-OF-PASS FLAG
;        4511                  DCT_ADDR [IG_INT] = TRUE;                      ! IGNORE FURTHER INTERRUPTS
;        4512                  end;
;        4513
;        4514              end;                                               ! IF UNIT IS STILL ALIVE
;        4515
;        4516      end;                                                       ! ROUTINE UPD_IO_TALLY


                                        .SBTTL  UPD.IO.TALLY MULTI-DRIVE TEST ROUTINES
000000  004137  000000G           UPD.IO.TALLY:
                                        JSR     R1,$SAVE2                       ;                       4426
000004  013701  000000G                 MOV     RP.ADDR,R1                      ;                       4449
000010  126127  000014  000241          CMPB    14(R1),#241
000016  001027                          BNE     1$
000020  013700  000000G                 MOV     T.ADDR,R0                       ;                       4452
000024  005260  000016                  INC     16(R0)
000030  066110  000020                  ADD     20(R1),(R0)                     ;                       4453
000034  066160  000020  000032          ADD     20(R1),32(R0)                   ;                       4454
000042  012746  000016                  MOV     #16,-(SP)                       ;                       4455
000046  060016                          ADD     R0,(SP)
000050  004737  000000V                 JSR     PC,OVF.CHK
```

```
000054  013716  000000G                        MOV     T.ADDR,(SP)             ;                               4456
000060  004737  000000V                        JSR     PC,OVF.CHK
000064  013716  000000G                        MOV     T.ADDR,(SP)             ;                               4457
000070  062716  000032                        ADD     #32,(SP)
000074  000435                                BR      2$
000076  126127  000014  000242        1$:     CMPB    14(R1),#242             ;                               4461
000104  001034                                BNE     3$
000106  013700  000000G                        MOV     T.ADDR,R0               ;                               4464
000112  005260  000024                        INC     24(R0)
000116  066160  000020  000006                ADD     20(R1),6(R0)            ;                               4465
000124  066160  000020  000040                ADD     20(R1),40(R0)           ;                               4466
000132  012746  000024                        MOV     #24,-(SP)               ;                               4467
000136  060016                                ADD     R0,(SP)
000140  004737  000000V                        JSR     PC,OVF.CHK
000144  013716  000000G                        MOV     T.ADDR,(SP)             ;                               4468
000150  062716  000006                        ADD     #6,(SP)
000154  004737  000000V                        JSR     PC,OVF.CHK
000160  013716  000000G                        MOV     T.ADDR,(SP)             ;                               4469
000164  062716  000040                        ADD     #40,(SP)
000170  004737  000000V        2$:             JSR     PC,OVF.CHK
000174  005726                                TST     (SP)+                   ;                               4463
000176  013700  000000G        3$:             MOV     RP.ADDR,R0              ;                               4472
000202  126027  000014  000241                CMPB    14(R0),#241
000210  001404                                BEQ     4$
000212  126027  000014  000242                CMPB    14(R0),#242             ;                               4473
000220  001037                                BNE     8$
000222  013700  000000G        4$:             MOV     T.ADDR,R0               ;                               4476
000226  016002  000004                        MOV     4(R0),R2                ; *,MILLIONS
000232  066002  000012                        ADD     12(R0),R2               ; *,MILLIONS
000236  016001  000002                        MOV     2(R0),R1                ; *,THOUSANDS                   4477
000242  066001  000010                        ADD     10(R0),R1               ; *,THOUSANDS
000246  020127  001750                        CMP     R1,#1750                ; THOUSANDS,*                   4479
000252  103403                                BLO     5$
000254  005202                                INC     R2                      ; MILLIONS                      4482
000256  162701  001750                        SUB     #1750,R1                ; *,THOUSANDS                   4483
000262  013700  000000G        5$:             MOV     SWP.XFER,R0             ;                               4493
000266  001004                                BNE     6$
000270  020127  000144                        CMP     R1,#144                 ; THOUSANDS,*                   4497
000274  101411                                BLOS    8$
000276  000402                                BR      7$                      ;                               4500
000300  020200        6$:             CMP     R2,R0                   ; MILLIONS,*                    4507
000302  103406                                BLO     8$
000304  112737  000001  000000G        7$:     MOVB    #1,EOP.FLAG             ;                               4510
000312  052777  040000  000000G                BIS     #40000,@DCT.ADDR        ;                               4511
000320  000207                        8$:     RTS     PC                      ;                               4426

; Routine Size:  105 words,     Routine Base:  $CODE$ + 17730
; Maximum stack depth per invocation:  5 words
```

```
;       4517  routine OVF_CHK (ADDR) : novalue =
;       4518
;       4519  !+
;       4520  !       THIS ROUTINE IS CALLED FROM UPD_IO_TALLY TO CHECK FOR OVERFLOW IN
;       4521  !       CERTAIN STATISTICAL FIELDS OF THE CURRENT UNIT. SPECIFICALLY, THE
;       4522  !       LOW-ORDER FIELD OF THE NUMBER OF BYTES READ OR WRITTEN IS CHECKED FOR
;       4523  !       EXCEEDING 1000. IF TRUE, THEN THE HIGH-ORDER COUNT IS INCREMENTED. IF
;       4524  !       THAT EXCEEDS 1000, THEN THE MEGABYTE COUNT IN INCREMENTED.
;       4525  !
;       4526  !       INPUTS:
;       4527  !               ADDR - ADDRESS OF THE BYTES_READ_LO OR BYTES_WRIT_LO FIELD FOR
;       4528  !                       THE CURRENT UNIT (SEE STATISTIC TABLE (TALLY) LAYOUT)
;       4529  !-
;       4530
;       4531      begin
;       4532
;       4533      while ..ADDR gequ 1000 do                 ! IF LO-ORDER OVERFLOW
;       4534          begin
;       4535          .ADDR = ..ADDR - 1000;                ! SUBTRACT 1000
;       4536          (.ADDR + 2) = .(.ADDR + 2) + 1;       ! INCR HI-ORDER
;       4537          end;
;       4538
;       4539      if .(.ADDR + 2) gequ 1000                 ! IF HI-ORDER OVERFLOW
;       4540      then
;       4541          begin
;       4542          (.ADDR + 2) = .(.ADDR + 2) - 1000;    ! SUBTRACT 1000
;       4543          (.ADDR + 4) = .(.ADDR + 4) + 1;       ! INCREMENT MBYTES
;       4544          end;
;       4545
;       4546      end;                                      ! ROUTINE OVF_CHK


                                        .SBTTL  OVF.CHK MULTI-DRIVE TEST ROUTINES
000000  010146                  OVF.CHK:MOV     R1,-(SP)                        ;                       4517
000002  016600  000004                  MOV     4(SP),R0                        ; ADDR,*                4533
000006  012701  000002                  MOV     #2,R1                           ;                       4536
000012  060001                          ADD     R0,R1
000014  021027  001750          1$:     CMP     (R0),#1750                      ;                       4533
000020  103404                          BLO     2$
000022  162710  001750                  SUB     #1750,(R0)                      ;                       4535
000026  005211                          INC     (R1)                            ;                       4536
000030  000771                          BR      1$                              ;                       4533
000032  021127  001750          2$:     CMP     (R1),#1750                      ;                       4539
000036  103404                          BLO     3$
000040  162711  001750                  SUB     #1750,(R1)                      ;                       4542
000044  005260  000004                  INC     4(R0)                           ;                       4543
000050  012601                  3$:     MOV     (SP)+,R1                        ;                       4517
000052  000207                          RTS     PC

; Routine Size:  22 words,      Routine Base:  $CODE$ + 20252
; Maximum stack depth per invocation:  2 words
```

```
;       4547  routine HOST_WRT_CHK =
;       4548
;       4549  !+
;       4550  !       THIS ROUTINE IS CALLED FROM IO_RETPKT FOR ALL I/O TRANSFER RETURN
;       4551  !       PACKETS WITH "SUCCESS" STATUS CODES, BUT ONLY IF THE HOST WRITE-COMPARE
;       4552  !       OPTION WAS SELECTED BY THE OPERATOR.
;       4553  !
;       4554  !       IF THE CURRENT RETPKT BEING PROCESSED IS A WRITE FUNCTION, THEN THE
;       4555  !       PACKET INDEX (RP_INDX) IS SAVED IN THE CONTROLLER'S RETURN PACKET SAVE
;       4556  !       AREA (RP_SAVE). OTHERWISE, THE PACKET IS A READ, SO ITS ASSOCIATED
;       4557  !       WRITE PACKET IS REMOVED FROM THE SAVE AREA, AND A BYTE-BY-BYTE
;       4558  !       COMPARISON IS PERFORMED ON THE TWO I/O BUFFERS. ANY DIFFERENCES
;       4559  !       ENCOUNTERED RESULTS IN THE DECLARATION OF A HARD ERROR.
;       4560  !
;       4561  !       IMPLICIT INPUTS:
;       4562  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;       4563  !               RP_INDX - INDEX OF THE CURRENT RETURN PACKET
;       4564  !-
;       4565
;       4566      begin
;       4567
;       4568      local
;       4569          BUFF1 : ref block [MAX_XFER * 2, byte], ! I/O BUFFER ADDRESS
;       4570          BUFF2 : ref block [MAX_XFER * 2, byte], ! I/O BUFFER ADDRESS
;       4571          BUFFW,                                  ! I/O BUFFER ADDRESS
;       4572          COUNT : word,                           ! BYTE COUNT
;       4573          FLAG : byte initial (byte (TRUE)),
;       4574          index : signed word;
;       4575
;       4576      if .RP_ADDR [ENDCOD] eql (OP_WRT or OP_END) ! IF WRITE OPERATION
;       4577      then
;       4578          FLAG = FALSE                            ! DON'T CALL SWEEP FROM IO_RETPKT
;       4579      else
;       4580          if (.RP_ADDR [ENDCOD] eql (OP_RD or OP_END)) and
;       4581             ((index = RPS_REM ()) geq 0)         ! IF ASSOCIATED WRITE PACKET IS FOUND ELSE ENDCODE IS READ
;       4582          then
;       4583              begin
;       4584              BUFFW = RETPKT [.index, BUFF_O];    ! ADDR OF ADDR OF WRITE I/O BUFFER
;       4585              BUFF1 = ..BUFFW;                    ! ADDR OF WRITE I/O BUFFER
;       4586              BUFF2 = .RP_ADDR [BU... O];         ! ADDR OF READ I/O BUFFER
;       4587              COUNT = .RP_ADDR [BU..._LO];        ! BYTE COUNT
;       4588
;       4589              incr I from 1 to .COUNT do          ! FOR EACH BYTE IN BUFFERS
;       4590
;       4591                  if .(.BUFF1)<0, 8, 0> eql .(.BUFF2)<0, 8, 0>    ! IF BYTES COMPARE O.K.
;       4592                  then
;       4593                      begin
;       4594                      BUFF1 = .BUFF1 + 1;         ! ADVANCE WRITE BUFFER ADDR
;       4595                      BUFF2 = .BUFF2 + 1;         ! ADVANCE READ BUFFER ADDR
;       4596                      end
;       4597                  else
;       4598                      begin                       ! ELSE - COMPARE ERROR
;       4599                      T_ADDR [ERR_HARD] = .T_ADDR [ERR_HARD] + 1;
```

```
;          4600                    T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;          4601                    ERRHRD (42, EGH_30, 0);     ! I/O REQUEST FAILED
;          4602                    EMS_CMP (RETPKT + (.index * RP_LEN * 2));
;          4603
;          4604                    if .T_ADDR [ERR_HARD] gequ .SWP_ERROR
;          4605                    then
;          4606                        begin
;          4607                        DUR [.L$LUN] = DU_HERR; ! IF ERROR COUNT EXCEEDED
;          4608                        DODU (.L$LUN);          ! DROP UNIT
;          4609                        end;
;          4610
;          4611                    exitloop;                   ! NO NEED TO CONTINUE
;          4612                    end;                        ! IF COMPARE ERROR
;          4613
;          4614                end;                            ! IF ASSOCIATED WRITE RETPKT WAS FOUND
;          4615
;          4616        return (.FLAG);
;          4617        end;                                    ! ROUTINE HOST_WRT_CHK


                                    .SBTTL  HOST.WRT.CHK ULTI-DRIVE TEST ROUTINES
000000  004137  000000G          HOST.WRT.CHK:
                                    JSR     R1,$SAVE5                       ;                       4547
000004  005746                      TST     -(SP)
000006  112705  000001              MOVB    #1,R5                           ; *,FLAG                4566
000012  013700  000000G             MOV     RP.ADDR,R0                      ;                       4576
000016  126027  000014  000242      CMPB    14(R0),#242
000024  001002                      BNE     1$
000026  105005                      CLRB    R5                              ; FLAG                  4578
000030  000500                      BR      6$                              ;                       4576
000032  126027  000014  000241  1$: CMPB    14(R0),#241                     ;                       4580
000040  001074                      BNE     6$
000042  004737  000000V             JSR     PC,RPS.REM                      ;                       4581
000046  005700                      TST     R0                              ; INDEX
000050  002470                      BLT     6$
000052  010046                      MOV     R0,-(SP)                        ; INDEX,*               4584
000054  012746  000060              MOV     #60,-(SP)
000060  004737  000000G             JSR     PC,BL$MUL
000064  010066  000004              MOV     R0,4(SP)
000070  062700  000024G             ADD     #RETPKT+24,R0                   ; *,BUFFW
000074  011002                      MOV     (R0),R2                         ; BUFFW,BUFF1           4585
000076  013700  000000G             MOV     RP.ADDR,R0                      ;                       4586
000102  016003  000024              MOV     24(R0),R3                       ; *,BUFF2
000106  016004  000020              MOV     20(R0),R4                       ; *,COUNT               4587
000112  005001                      CLR     R1                              ; I                     4589
000114  000442                      BR      4$
000116  121213              2$:     CMPB    (R2),(R3)                       ; BUFF1,BUFF2           4591
000120  001003                      BNE     3$
000122  005202                      INC     R2                              ; BUFF1                 4594
000124  005203                      INC     R3                              ; BUFF2                 4595
000126  000435                      BR      4$                              ;                       4591
000130  013700  000000G         3$: MOV     T.ADDR,R0                       ;                       4599
000134  005260  000014              INC     14(R0)
```

```
000140  105260  000063                        INCB    63(R0)                          ;                               4600
000144  104456                                TRAP    56                              ;                               4601
000146  000052                                .WORD   52
000150  000000G                               .WORD   EGH.30
000152  000000                                .WORD   0
000154  016616  000004                        MOV     4(SP),(SP)                      ;                               4602
000160  062716  000000G                       ADD     #RETPKT,(SP)
000164  004737  000000G                       JSR     PC,EMS.CMP
000170  013700  000000G                       MOV     T.ADDR,R0                       ;                               4604
000174  026037  000014  000000G               CMP     14(R0),SWP.ERROR
000202  103412                                BLO     5$
000204  013700  000000G                       MOV     L$LUN,R0                        ;                               4607
000210  112760  000004  000000G               MOVB    #4,DUR(R0)
000216  104451                                TRAP    51                              ;                               4608
000220  000403                                BR      5$                              ;                               4598
000222  005201                        4$:     INC     R1                              ; I                             4589
000224  020104                                CMP     R1,R4                           ; I,COUNT
000226  003733                                BLE     2$
000230  022626                        5$:     CMP     (SP)+,(SP)+                     ;                               4583
000232  005000                        6$:     CLR     R0                              ;                               4566
000234  150500                                BISB    R5,R0                           ; FLAG,*
000236  005726                                TST     (SP)+                           ;                               4547
000240  000207                                RTS     PC

; Routine Size:  81 words,      Routine Base:  $CODE$ + 20326
; Maximum stack depth per invocation:  11 words
```

```
;        4618  routine SWEEP : novalue =
;        4619
;        4620  !+
;        4621  !       THIS ROUTINE IS CALLED FROM IO_RETPKT AND OTHERS TO DEALLOCATE THE
;        4622  !       RESOURCES ASSOCIATED WITH THE CURRENT RETURN PACKET. THIS INCLUDES THE
;        4623  !       PACKET ITSELF AND THE I/O BUFFER. IN ADDITION, IF THE HOST IS
;        4624  !       PERFORMING WRITE-COMPARES, AND IF THE CURRENT RETURN PACKET IS A READ
;        4625  !       FUNCTION, THEN THE CURRENT CONTROLLER'S RP SAVE AREA IS SEARCHED FOR
;        4626  !       THE ASSOCIATED WRITE RETPKT SO THAT ITS RESOURCES CAN ALSO BE
;        4627  !       DEALLOCATED.
;        4628  !
;        4629  !       IMPLICIT INPUTS:
;        4630  !               RP_ADDR - ADDRESS OF CURRENT RETURN PACKET
;        4631  !               RP_INDX - INDEX OF CURRENT RETURN PACKET
;        4632  !-
;        4633
;        4634      begin
;        4635
;        4636      local
;        4637          index : signed word;
;        4638
;        4639      if (.RP_ADDR [ENDCOD] and OP_MSK) eql OP_RD ! IF READ OPCODE OR ENDCODE
;        4640      then
;        4641
;        4642          if BIT_TST (SWP_FLAGS, SWF_HWC)         ! IF HOST IS DOING WRITE-COMPARES
;        4643          then
;        4644
;        4645              if (index = RPS_REM ()) geq 0       ! IF ASSOCIATED WRITE RETPKT IS FOUND
;        4646              then
;        4647                  begin
;        4648                  PUT_IO_BUFF (RETPKT [.index, BUFF_O]);  ! RETURN WRITE I/O BUFFER TO POOL
;        4649                  PUT_RETPKT (.index);            ! RETURN WRITE PACKET TO POOL
;        4650                  end;
;        4651
;        4652      PUT_IO_BUFF (RP_ADDR [BUFF_O]);             ! RETURN CURRENT I/O BUFFER TO POOL
;        4653      PUT_RETPKT (.RP_INDX);                      ! RETURN CURRENT RETPKT TO POOL
;        4654      end;                                        ! ROUTINE SWEEP


                                        .SBTTL  SWEEP MULTI-DRIVE TEST ROUTINES
000000  010146                  SWEEP:  MOV     R1,-(SP)                        ;                       4618
000002  013700  000000G                 MOV     RP.ADDR,R0                      ;                       4639
000006  116000  000014                  MOVB    14(R0),R0
000012  042700  177600                  BIC     #177600,R0
000016  020027  000041                  CMP     R0,#41
000022  001026                          BNE     1$
000024  032737  000040  000000G         BIT     #40,SWP.FLAGS                   ;                       4642
000032  001422                          BEQ     1$
000034  004737  000000V                 JSR     PC,RPS.REM                      ;                       4645
000040  010001                          MOV     R0,R1                           ; *,INDEX
000042  002416                          BLT     1$
000044  010146                          MOV     R1,-(SP)                        ; INDEX,*               4648
000046  012746  000060                  MOV     #60,-(SP)
```

```
000052  004737  000000G                        JSR     PC,BL$MUL
000056  062700  000024G                        ADD     #RETPKT+24,R0
000062  010016                                 MOV     R0,(SP)
000064  004737  000000G                        JSR     PC,PUT.IO.BUFF
000070  010116                                 MOV     R1,(SP)          ; INDEX,*                4649
000072  004737  000000G                        JSR     PC,PUT.RETPKT
000076  022626                                 CMP     (SP)+,(SP)+      ;                        4647
000100  013746  000000G             1$:        MOV     RP.ADDR,-(SP)    ;                        4652
000104  062716  000024                         ADD     #24,(SP)
000110  004737  000000G                        JSR     PC,PUT.IO.BUFF
000114  013716  000000G                        MOV     RP.INDX,(SP)     ;                        4653
000120  004737  000000G                        JSR     PC,PUT.RETPKT
000124  005726                                 TST     (SP)+            ;                        4634
000126  012601                                 MOV     (SP)+,R1         ;                        4618
000130  000207                                 RTS     PC

; Routine Size:  45 words,      Routine Base:  $CODE$ + 20570
; Maximum stack depth per invocation:  4 words
```

```
;          4655  routine RPS_REM =
;          4656
;          4657  !+
;          4658  !       THIS ROUTINE SEARCHES THE CURRENT CONTROLLER'S RP_SAVE AREA FOR A
;          4659  !       RETURN PACKET WHOSE COMMAND REFERENCE NUMBER (CRN) IS ONE LESS THAN THE
;          4660  !       CRN OF THE CURRENT RETURN PACKET (I.E., SEARCHING FOR THE SAVED WRITE
;          4661  !       OPERATION ASSOCIATED WITH THE CURRENT READ OPERATION). IF FOUND, THE
;          4662  !       RP_SAVE ENTRY IS CLEARED (TO -1) AND THE RETPKT INDEX OF THE WRITE
;          4663  !       OPERATION IS RETURNED TO THE CALLER.
;          4664  !
;          4665  !       IMPLICIT INPUTS:
;          4666  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;          4667  !
;          4668  !       OUTPUTS:
;          4669  !               INDEX (VALUE OF THIS ROUTINE) - INDEX OF THE RETPKT CONTAINING
;          4670  !                       A CRN WHICH IS ONE LESS THAN THE CURRENT
;          4671  !-
;          4672
;          4673      begin
;          4674
;          4675      local
;          4676          index : signed word initial (-1);                          ! ASSUME NOT FOUND
;          4677
;          4678      incr COUNT from 0 to RP_CNT - 1 do                             ! FOR EACH ENTRY IN RP_SAVE
;          4679
;          4680          if (.RP_USE [.COUNT] eql .CCTLR) and                               ! IF THIS IS A VALID RETPKT
;          4681             (.RETPKT [.COUNT, ENDCOD] eql (OP_WRT or OP_END))
;          4682          then
;          4683
;          4684              if ((.RETPKT [.COUNT, CRF_LO] eql (.RP_ADDR [CRF_LO] - 1)) and     ! IF CORRECT CRN
;          4685                  (.RETPKT [.COUNT, CRF_HI] eql .RP_ADDR [CRF_HI])) or
;          4686                 ((.RETPKT [.COUNT, CRF_HI] eql (.RP_ADDR [CRF_HI] - 1)) and
;          4687                  (.RETPKT [.COUNT, CRF_LO] eql %o'177777') and
;          4688                  (.RP_ADDR [CRF_LO] eql 0))
;          4689              then
;          4690                  begin
;          4691                  index = .COUNT;                                    ! INDEX TO BE RETURNED
;          4692                  exitloop;                                          ! DONE
;          4693                  end;
;          4694
;          4695      return .index;
;          4696      end;                                                           ! ROUTINE RPS_REM


                                        .SBTTL  RPS.REM MULTI-DRIVE TEST ROUTINES
000000  004137  000000G         RPS.REM:JSR     R1,$SAVE4                       ;                       4655
000004  012704  177777                  MOV     #-1,R4                          ; *,INDEX               4673
000010  005003                          CLR     R3                              ; COUNT                 4678
000012  116300  000000G         1$:     MOVB    RP.USE(R3),R0                   ; *(COUNT),*            4680
000016  020037  000000G                 CMP     R0,CCTLR
000022  001053                          BNE     4$
000024  010346                          MOV     R3,-(SP)                        ; COUNT,*               4681
000026  012746  000060                  MOV     #60,-(SP)
```

```
000032  004737  000000G                        JSR     PC,BL$MUL
000036  022626                                  CMP     (SP)+,(SP)+
000040  126027  000014G 000242                 CMPB    RETPKT+14(R0),#242
000046  001041                                  BNE     4$
000050  010346                                  MOV     R3,-(SP)                ; COUNT,*                       4684
000052  012746  000060                         MOV     #60,-(SP)
000056  004737  000000G                        JSR     PC,BL$MUL
000062  022626                                  CMP     (SP)+,(SP)+
000064  013701  000000G                        MOV     RP.ADDR,R1
000070  016102  000004                         MOV     4(R1),R2
000074  005302                                  DEC     R2
000076  026002  000004G                        CMP     RETPKT+4(R0),R2
000102  001004                                  BNE     2$
000104  026061  000006G 000006                 CMP     RETPKT+6(R0),6(R1)      ;                               4685
000112  001415                                  BEQ     3$
000114  016102  000006              2$:         MOV     6(R1),R2                ;                               4686
000120  005302                                  DEC     R2
000122  026002  000006G                        CMP     RETPKT+6(R0),R2
000126  001011                                  BNE     4$
000130  026027  000004G 177777                 CMP     RETPKT+4(R0),#-1        ;                               4687
000136  001005                                  BNE     4$
000140  005761  000004                         TST     4(R1)                   ;                               4688
000144  001002                                  BNE     4$
000146  010304                      3$:         MOV     R3,R4                   ; COUNT,INDEX                   4691
000150  000404                                  BR      5$                      ;                               4690
000152  005203                      4$:         INC     R3                      ; COUNT                         4678
000154  020327  000003                         CMP     R3,#3                   ; COUNT,*
000160  003714                                  BLE     1$
000162  010400                      5$:         MOV     R4,R0                   ; INDEX,*                       4673
000164  000207                                  RTS     PC                      ;                               4655

; Routine Size:  59 words,      Routine Base:  $CODE$ + 20722
; Maximum stack depth per invocation:  8 words
```

```
;        4697  routine DR_RETPKT : novalue =
;        4698
;        4699  !+
;        4700  !       THIS ROUTINE IS CALLED BY PROC_RETPKT FOR ALL PACKETS ORIGINATING AT
;        4701  !       THE "DRIVER" PORTION OF THE PROGRAM. THIS INCLUDES PACKETS DESCRIBING
;        4702  !       FATAL DEVICE ERRORS.
;        4703  !
;        4704  !       FOR FATAL DEVICE ERRORS, THIS ROUTINE RELEASES ALL RESOURCES HELD BY
;        4705  !       THE CONTROLLER. THE CONTROLLER IS MARKED OFFLINE IN ITS CST, AND ALL
;        4706  !       UNITS ATTACHED TO THE CONTROLLER ARE DROPPED.
;        4707  !
;        4708  !       IMPLICIT INPUTS:
;        4709  !               RP_INDX - INDEX OF THE CURRENT RETURN PACKET
;        4710  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;        4711  !               CST_ADDR - ADDRESS OF THE CURRENT CONTROLLER'S CST
;        4712  !               CCTLR - CURRENT CONTROLLER NUMBER
;        4713  !-
;        4714
;        4715      begin
;        4716
;        4717
;        4718      PUTA_BUFF ();                           ! RELEASE ALL I/O BUFFERS HELD BY CONTROLLER
;        4719
;        4720      incr index from 0 to RP_CNT - 1 do      ! FOR EACH ENTRY IN CONTROLLER'S RP_SAVE
;        4721
;        4722          if .RP_USE [.index] eql .CCTLR      ! IF VALID RETPKT INDEX
;        4723          then
;        4724              PUT_RETPKT (.index);            ! RETURN RETPKT TO POOL
;        4725
;        4726      QIO [.CCTLR] = 0;                       ! CLEAR NO. OF OUTSTANDING QIOS
;        4727      CST_ADDR [STATE] = OFFLINE;             ! MARK CST OFFLINE
;        4728      DROP_CTLR (.CCTLR, DU_CFATAL);          ! DROP CONTROLLER'S UNITS
;        4729      PUT_RETPKT (.RP_INDX);                  ! PUT BACK RETPKT
;        4730      end;                                    ! ROUTINE DR_RETPKT

                                    .SBTTL  DR.RETPKT MULTI-DRIVE TEST ROUTINES
000000  010146                      DR.RETPKT:
                                    MOV     R1,-(SP)                        ;                       4697
000002  004737  000000G             JSR     PC,PUTA.BUFF                    ;                       4718
000006  005001                      CLR     R1                              ; INDEX                 4720
000010  116100  000000G     •       1$:     MOVB    RP.USE(R1),R0           ; *(INDEX),*            4722
000014  020037  000000G             CMP     R0,CCTLR
000020  001004                      BNE     2$
000022  010146                      MOV     R1,-(SP)                        ; INDEX,*               4724
000024  004737  000000G             JSR     PC,PUT.RETPKT
000030  005726                      TST     (SP)+
000032  005201                      2$:     INC     R1                      ; INDEX                 4720
000034  020127  000003              CMP     R1,#3                           ; INDEX,*
000040  003763                      BLE     1$
000042  013701  000000G             MOV     CCTLR,R1                        ;                       4726
000046  105061  000000G             CLRB    QIO(R1)
000052  013700  000000G             MOV     CST.ADDR,R0                     ;                       4727
```

```
000056  042760  100000  000002          BIC     #100000,2(R0)
000064  010146                          MOV     R1,-(SP)        ;                       4728
000066  012746  000006                  MOV     #6,-(SP)
000072  004737  000000G                 JSR     PC,DROP.CTLR
000076  013716  000000G                 MOV     RP.INDX,(SP)    ;                       4729
000102  004737  000000G                 JSR     PC,PUT.RETPKT
000106  022626                          CMP     (SP)+,(SP)+     ;                       4715
000110  012601                          MOV     (SP)+,R1        ;                       4697
000112  000207                          RTS     PC

; Routine Size:  38 words,      Routine Base:  $CODE$ + 21110
; Maximum stack depth per invocation:  4 words
```

```
;          4731  %sbttl 'RDRX INTERRUPT SERVICE ROUTINES'
;          4732
;          4733  !+
;          4734  !       THERE EXISTS AN RDRX INTERRUPT SERVICE ROUTINE FOR EACH DEVICE
;          4735  !       CONTROLLER. EACH SERVICE ROUTINE BEGINS BY SIMPLY SETTING THE
;          4736  !       APPROPRIATE CONTROLLER NUMBER INTO "ICTLR". ALL SERVICE ROUTINES THEN
;          4737  !       BRANCH TO A COMMON INTERRUPT PROCESSING ROUTINE.
;          4738  !-
;          4739
;          4740  BGNSRV (AZINT0);
;          4741  ICTLR = 0;
;          4742  AZINT ();
;          4743  ENDSRV;


                                        .SBTTL  AZINT0 RDRX INTERRUPT SERVICE ROUTINES
000000  010046                  AZINT0::MOV     R0,-(SP)                        ;       4740
000002  005037  000156'                 CLR     ICTLR                           ;       4741
000006  004737  000000V                 JSR     PC,AZINT                        ;       4742
000012  012600                          MOV     (SP)+,R0                        ;       4740
000014  000002                          RTI

; Routine Size:  7 words,       Routine Base:  $CODE$ + 21224
; Maximum stack depth per invocation:  2 words
```

```
;        4744  routine AZINT : novalue =
;        4745
;        4746  !+
;        4747  !       THIS IS THE COMMON INTERRUPT SERVICE ROUTINE FOR ALL RDRX CONTROLLERS.
;        4748  !       AFTER CALCULATING THE DCT ADDRESS FOR THE INTERRUPTING DEVICE, THIS
;        4749  !       ROUTINE WILL SAVE THE CURRENT CONTENTS OF THE SA REGISTER IN THE DCT.
;        4750  !       THEN, IF THE "IGNORE INTERRUPT" BIT IS SET, NO FURTHER ACTION IS TAKEN.
;        4751  !       OTHERWISE, THE SA VALUE IS CHECKED FOR A FATAL ERROR, AND THE COMMAND
;        4752  !       AND RESPONSE RINGS ARE POLLED.
;        4753  !-
;        4754
;        4755      begin
;        4756      IDCT_ADDR = DCT + (.ICTLR * DCT_LEN * 2);    ! GET DCT ADDRESS
;        4757      ICST_ADDR = CST + (.ICTLR * CST_LEN * 2);    ! GET CST ADDRESS
;        4758      IRDRX_ADDR = .ICST_ADDR [IP_ADDR];           ! GET RDRX ADDRESS
;        4759      ICOM_ADDR = COMM_AREA + (.ICTLR * COMM_LEN * 2);    ! GET COMM_AREA ADDR
;        4760      IDCT_ADDR [SA_SAVE] = .IRDRX_ADDR [RCSA, RC_ALL];   ! SAVE SA REGISTER
;        4761
;        4762      if .IDCT_ADDR [IG_INT]                       ! IGNORE INTERRUPT?
;        4763      then
;        4764          return;                                  ! RETURN IF INTERRUPTS IGNORED
;        4765
;        4766      if BIT_TST (IDCT_ADDR [SA_SAVE], SA_ERR)     ! IF FATAL ERROR
;        4767      then
;        4768          FATAL_ERROR ()
;        4769      else
;        4770          begin
;        4771          POLL_CRING ();                           ! POLL COMMAND RING
;        4772          POLL_RRING ();                           ! POLL RESPONSE RING
;        4773          end;
;        4774
;        4775      end;


                                .SBTTL  AZINT RDRX INTERRUPT SERVICE ROUTINES
000000  010146          AZINT:  MOV     R1,-(SP)                        ;                       4744
000002  005746                  TST     -(SP)
000004  013701  000156'         MOV     ICTLR,R1                        ;                       4756
000010  010146                  MOV     R1,-(SP)
000012  012746  000022          MOV     #22,-(SP)
000016  004737  000000G         JSR     PC,BL$MUL
000022  062700  000000G         ADD     #DCT,R0
000026  010037  000110'         MOV     R0,IDCT.ADDR
000032  010116                  MOV     R1,(SP)                         ;                       4757
000034  012746  000056          MOV     #56,-(SP)
000040  004737  000000G         JSR     PC,BL$MUL
000044  062700  000000G         ADD     #CST,R0
000050  010037  000106'         MOV     R0,ICST.ADDR
000054  011037  000000G         MOV     (R0),IRDRX.ADDR                 ; ICST.ADDR,*           4758
000060  010116                  MOV     R1,(SP)                         ;                       4759
000062  012746  000050          MOV     #50,-(SP)
000066  004737  000000G         JSR     PC,BL$MUL
000072  062700  000000'         ADD     #COMM.AREA,R0
```

```
000076  010037  000104'                        MOV     R0,ICOM.ADDR
000102  013700  000110'                        MOV     IDCT.ADDR,R0            ;                                       4760
000106  013701  000000G                        MOV     IRDRX.ADDR,R1
000112  016166  000002  000010                 MOV     2(R1),10(SP)            ; *,RC.REG
000120  016660  000010  000002                 MOV     10(SP),2(R0)            ; RC.REG,*
000126  032710  040000                         BIT     #40000,(R0)             ; *,IDCT.ADDR                           4762
000132  001016                                 BNE     2$                      ;                                       4744
000134  016600  000010                         MOV     10(SP),R0               ;                                       4766
000140  042700  077777                         BIC     #77777,R0
000144  020027  100000                         CMP     R0,#-100000
000150  001003                                 BNE     1$
000152  004737  000000V                        JSR     PC,FATAL.ERROR          ;                                       4768
000156  000404                                 BR      2$                      ;                                       4766
000160  004737  000000V             1$:        JSR     PC,POLL.CRING           ;                                       4771
000164  004737  000000V                        JSR     PC,POLL.RRING           ;                                       4772
000170  062706  000012              2$:        ADD     #12,SP                  ;                                       4744
000174  012601                                 MOV     (SP)+,R1
000176  000207                                 RTS     PC

; Routine Size:  64 words,      Routine Base:  $CODE$ + 21242
; Maximum stack depth per invocation:  7 words
```

```
;        4776  routine FATAL_ERROR : novalue =
;        4777
;        4778  !+
;        4779  !       THIS ROUTINE IS CALLED BY THE INTERRUPT SERVICE ROUTINE (AZINT) UPON
;        4780  !       DETECTING AN UNRECOVERABLE ERROR THROUGH THE DEVICE'S SA REGISTER.
;        4781  !       ITS PURPOSE IS TO CLEAN UP DEVICE DATA IN THE "DRIVER" PORTION OF
;        4782  !       THE EXERCISER, AND TO INFORM THE "PROGRAM" PORTION OF THE EVENT VIA
;        4783  !       RETURN PACKET.
;        4784  !
;        4785  !       IMPLICIT INPUTS:
;        4786  !               ICTLR - INTERRUPTING CONTROLLER NUMBER
;        4787  !               IDCT_ADDR - ADDRESS OF INTERRUPTING CONTROLLER'S DCT
;        4788  !               ICST_ADDR - ADDRESS OF INTERRUPTING CONTROLLER'S CST
;        4789  !               IRDRX_ADDR - ADDRESS OF INTERRUPTING CONTROLLER'S IP REGISTER
;        4790  !-
;        4791
;        4792      begin
;        4793
;        4794      local
;        4795          index : signed word,
;        4796          U_SAVE : word;
;        4797
;        4798      SA_REG = .IDCT_ADDR [SA_SAVE];
;        4799      U_SAVE = .L$LUN;                                ! SAVE PRE-INTERRUPT CURRENT UNIT NUMBER
;        4800      C_ERR_TBL [.ICTLR, C_ERR_HRD] = .C_ERR_TBL [.ICTLR, C_ERR_HRD] + 1;
;        4801      L$LUN = .ICST_ADDR [OF_UN, D_UNIT];             ! SET CURRENT UNIT TO FIRST IN CONTROLLER
;        4802      ERRDF (14, EGD_14, EMS_14);                     ! FATAL CONTROLLER ERROR
;        4803      L$LUN = .U_SAVE;                                ! RESTORE PRE-INTERRUPT CURRENT UNIT
;        4804      DRV_CTLERR (.ICTLR);                            ! CLEAN UP DRIVER DATA FOR CONTROLLER
;        4805
;        4806      if (index = GET_RETPKT (.ICTLR)) lss 0          ! TRY TO GET A RETPKT; IF FAILURE
;        4807      then
;        4808          PRINTF (DBM18)                              ! "FATAL_ERROR: RETPKT NOT AVAILABLE"
;        4809      else
;        4810          begin                                       ! IF RETPKT WAS ALLOCATED
;        4811          RETPKT [.index, CONID] = CID_DRIVER;        ! SET CONNECTION ID TO "DRIVER"
;        4812          RETPKT [.index, MESTYP] = MT_FATAL;         ! FATAL ERROR
;        4813          RETPKT [.index, CTLR] = .ICTLR;             ! CONTROLLER NUMBER
;        4814          IN_IODQ (.index);                           ! LOAD RETPKT INDEX INTO IODQ
;        4815          end;                                        ! IF RETPKT WAS ALLOCATED
;        4816
;        4817      end;                                            ! ROUTINE FATAL_ERR


                                        .SBTTL  FATAL.ERROR RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G          FATAL.ERROR:
                                        JSR     R1,$SAVE2                       ;                       4776
000004  013700  000110'                 MOV     IDCT.ADDR,R0                    ;                       4798
000010  016037  000002  000000G         MOV     2(R0),SA.REG
000016  013701  000000G                 MOV     L$LUN,R1                        ; *,U.SAVE              4799
000022  013700  000156'                 MOV     ICTLR,R0                        ;                       4800
000026  006300                          ASL     R0
000030  105260  000000G                 INCB    C.ERR.TBL(R0)
```

```
000034  013700  000106'                        MOV     ICST.ADDR,R0                    ;                               4801
000040  016002  000006                         MOV     6(R0),R2
000044  000302                                 SWAB    R2
000046  042702  177760                         BIC     #177760,R2
000052  010237  000000G                        MOV     R2,L$LUN
000056  104455                                 TRAP    55                              ;                               4802
000060  000016                                 .WORD   16
000062  000000G                                .WORD   EGD.14
000064  000000G                                .WORD   EMS.14
000066  010137  000000G                        MOV     R1,L$LUN                        ; U.SAVE,*                      4803
000072  013746  000156'                        MOV     ICTLR,-(SP)                     ;                               4804
000076  004737  000000G                        JSR     PC,DRV.CTLERR
000102  013716  000156'                        MOV     ICTLR,(SP)                      ;                               4806
000106  004737  000000G                        JSR     PC,GET.RETPKT
000112  010001                                 MOV     R0,R1                           ; *,INDEX
000114  002007                                 BGE     1$
000116  012716  000000G                        MOV     #DBM18,(SP)                     ;                               4808
000122  012746  000001                         MOV     #1,-(SP)
000126  010600                                 MOV     SP,R0                           ; SP,*
000130  104417                                 TRAP    17
000132  000424                                 BR      2$                              ;                               4806
000134  010116                         1$:     MOV     R1,(SP)                         ; INDEX,*                       4811
000136  012746  000060                         MOV     #60,-(SP)
000142  004737  000000G                        JSR     PC,BL$MUL
000146  062700  000002G                        ADD     #RETPKT+2,R0
000152  112760  000003  000001                 MOVB    #3,1(R0)
000160  013702  000156'                        MOV     ICTLR,R2                        ;                               4813
000164  042702  177760                         BIC     #177760,R2
000170  112710  000060                         MOVB    #60,(R0)
000174  150210                                 BISB    R2,(R0)
000176  010116                                 MOV     R1,(SP)                         ; INDEX,*                       4814
000200  C04737  000000G                        JSR     PC,IN.IODQ
000204  022626                         2$:     CMP     (SP)+,(SP)+                     ;                               4792
000206  000207                                 RTS     PC                              ;                               4776

; Routine Size:  68 words,      Routine Base:  $CODE$ + 21442
; Maximum stack depth per invocation:  7 words
```

```
;         4818  routine POLL_CRING : novalue =
;         4819
;         4820  !+
;         4821  !       THIS ROUTINE IS CALLED BY THE RDRX INTERRUPT SERVICE ROUTINE (AZINT)
;         4822  !       FOR EACH DEVICE INTERRUPT EXCEPT DURING INITIALIZATION OR FATAL ERROR.
;         4823  !       ITS PURPOSE IS TO SCAN THE DEVICE'S COMMAND RING AND CHECK FOR ANY
;         4824  !       COMMAND SLOTS THAT HAVE BEEN "TAKEN" BY THE CONTROLLER. SUCH SLOTS
;         4825  !       HAVE BEEN RETURNED TO THE HOST, INDICATED BY A ZERO OWNERSHIP BIT. FOR
;         4826  !       EACH SLOT THAT HAS BEEN RETURNED TO THE HOST, THE CRING COUNT IS
;         4827  !       DECREMENTED, AND THE CR_POLL ADDRESS IS ADVANCED TO THE NEXT SLOT IN
;         4828  !       THE COMMAND RING.
;         4829  !
;         4830  !       IMPLICIT INPUTS:
;         4831  !               ICTLR - INTERRUPTING CONTROLLER NUMBER
;         4832  !               IDCT_ADDR - ADDRESS OF INTERRUPTING CONTROLLER'S DCT
;         4833  !               ICOM_ADDR - ADDRESS OF INTERRUPTING CONTROLLER'S COMM_AREA
;         4834  !-
;         4835
;         4836      begin
;         4837
;         4838      while ((.IDCT_ADDR [CRING_CNT] gtru 0) and                      ! WHILE # OF COMMANDS IN CRING > 0 AND
;         4839          not (BIT_TST ((.IDCT_ADDR [CR_POLL] + 2), ED_OWN))) do      ! CURRENT SLOT IS HOST-OWNED
;         4840          begin
;         4841          IDCT_ADDR [CRING_CNT] = .IDCT_ADDR [CRING_CNT] - 1;         ! DECREMENT # CMDS IN CRING
;         4842          IDCT_ADDR [CR_POLL] = .IDCT_ADDR [CR_POLL] + 4;             ! ADVANCE TO NEXT SLOT TO POLL
;         4843
;         4844          if .IDCT_ADDR [CR_POLL] gtra .IDCT_ADDR [CR_END]            ! IF BEYOND END OF RING
;         4845          then
;         4846              IDCT_ADDR [CR_POLL] = .IDCT_ADDR [CR_BEG];              ! SET POINTER TO TOP OF CRING
;         4847
;         4848          end;
;         4849
;         4850      ICOM_ADDR [CMD_INT] = 0;                                        ! CLEAR COMMAND INTERRUPT WORD IN RING HEADER
;         4851      end;


                                        .SBTTL  POLL.CRING RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G          POLL.CRING:
                                        JSR     R1,$SAVE2                       ;                                       4818
000004  013701  000110'                 MOV     IDCT.ADDR,R1                    ;                                       4838
000010  012702  000016                  MOV     #16,R2                          ;                                       4842
000014  060102                          ADD     R1,R2
000016  105711                  1$:     TSTB    (R1)                            ;                                       4838
000020  001422                          BEQ     2$
000022  016100  000016                  MOV     16(R1),R0                       ;                                       4839
000026  016000  000002                  MOV     2(R0),R0
000032  042700  077777                  BIC     #77777,R0
000036  020027  100000                  CMP     R0,#-100000
000042  001411                          BEQ     2$
000044  105311                          DECB    (R1)                            ;                                       4841
000046  062712  000004                  ADD     #4,(R2)                         ;                                       4842
000052  021261  000012                  CMP     (R2),12(R1)                     ;                                       4844
000056  101757                          BLOS    1$
```

```
000060  016112  000010                          MOV     10(R1),(R2)                     ;                       4846
000064  000754                                  BR      1$                              ;                       4838
000066  013700  000104'                 2$:     MOV     ICOM.ADDR,R0                    ;                       4850
000072  005060  000004                          CLR     4(R0)
000076  000207                                  RTS     PC                              ;                       4818

; Routine Size:  32 words,      Routine Base:  $CODE$ + 21652
; Maximum stack depth per invocation:  4 words
```

```
;        4852  routine POLL_RRING : novalue =
;        4853
;        4854  !+
;        4855  !       THIS ROUTINE IS CALLED BY THE RDRX INTERRUPT SERVICE ROUTINE (AZINT)
;        4856  !       FOR EACH DEVICE INTERRUPT EXCEPT DURING INITIALIZATION OR FATAL ERROR.
;        4857  !       ITS PURPOSE IS TO SCAN THE DEVICE'S RESPONSE RING AND CHECK FOR ANY
;        4858  !       SLOTS WHICH HAVE BEEN RETURNED TO THE HOST (OWNERSHIP BIT = 0). FOR
;        4859  !       EACH SUCH SLOT, THE ASSOCIATED MESSAGE IS PROCESSED BASED ON ITS
;        4860  !       CONNECTION ID (MSCP OR DUP). AFTER PROCESSING, THE MESSAGE PACKET
;        4861  !       IS RE-INITIALIZED AND RETURNED TO THE CONTROLLER (OWNERSHIP BIT SET
;        4862  !       TO 1).
;        4863  !
;        4864  !       IMPLICIT INPUTS:
;        4865  !               ICTLR - NUMBER OF INTERRUPTING CONTROLLER
;        4866  !               IDCT_ADDR - ADDRESS OF INTERRUPTING CONTROLLER'S DCT
;        4867  !-
;        4868
;        4869      begin
;        4870
;        4871      while not (BIT_TST ((.IDCT_ADDR [RR_POLL] + 2), ED_OWN)) do ! WHILE 0 = 0
;        4872          begin
;        4873          IPKT_ADDR = ..IDCT_ADDR [RR_POLL] - 10; ! ADDRESS OF RESPONSE PACKET
;        4874          CREDIT_BAL = .CREDIT_BAL + .IPKT_ADDR [CREDITS];
;        4875
;        4876          selectoneu .IPKT_ADDR [CONNID] of
;        4877              set
;        4878
;        4879              [CID_MSCP] :        MSCP_RSP ();
;        4880              [CID_DUP]  :        DUP_RSP ();
;        4881
;        4882              [otherwise] :       PRINTF (DBM20, .IPKT_ADDR [CONNID], .IRDRX_ADDR);
;        4883                                                  ! 'BAD CONN ID = XXXXX RECEIVED FROM XXXXXX'
;        4884              tes;
;        4885
;        4886          IPKT_ADDR [MSGLEN] = MSG_LEN * 2;               ! RE-INIT PKT FIELDS; MESSAGE LENGTH
;        4887          IDCT_ADDR [RR_POLL] = .IDCT_ADDR [RR_POLL] + 2; ! ADVANCE TO HI ORDER WORD OF RING SLOT
;        4888          .IDCT_ADDR [RR_POLL] = .IPKT_ADDR [PKT_HI];     ! RETURN SLOT TO CONTROLLER
;        4889          .IDCT_ADDR [RR_POLL] = ..IDCT_ADDR [RR_POLL] or ED_OWN or ED_FLAG; ! OWNERSHIP TOO
;        4890          IDCT_ADDR [RR_POLL] = .IDCT_ADDR [RR_POLL] + 2; ! ADVANCE TO NEXT RRING SLOT
;        4891
;        4892          if .IDCT_ADDR [RR_POLL] gtra .IDCT_ADDR [RR_END]! IF BEYOND END OF RING
;        4893          then
;        4894              IDCT_ADDR [RR_POLL] = .IDCT_ADDR [RR_BEG];  ! CYCLE TO TOP OF RING
;        4895
;        4896          end;                                            ! WHILE LOOP
;        4897
;        4898      ICOM_ADDR [RSP_INT] = 0;                            ! CLEAR RESPONSE INTERRUPT WORD IN RING HEADER
;        4899      end;


                                        .SBTTL  POLL.RRING RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G          POLL.RRING:
                                        JSR     R1,$SAVE2                       ;                                       4852
```

```
000004  013702  000110'                    MOV     IDCT.ADDR,R2                    ;                                       4871
000010  062702  000014                      ADD     #14,R2
000014  011200                      1$:     MOV     (R2),R0
000016  016000  000002                      MOV     2(R0),R0
000022  042700  077777                      BIC     #77777,R0
000026  020027  100000                      CMP     R0,#-100000
000032  001503                            BEQ     5$
000034  017237  000000  000000G             MOV     @0(R2),IPKT.ADDR                ;                                       4873
000042  162737  000012  000000G             SUB     #12,IPKT.ADDR
000050  013700  000000G                     MOV     IPKT.ADDR,R0                    ;                                       4874
000054  116001  000010                      MOVB    10(R0),R1
000060  042701  177760                      BIC     #177760,R1
000064  063701  000000G                     ADD     CREDIT.BAL,R1
000070  010137  000000G                     MOV     R1,CREDIT.BAL
000074  005001                            CLR     R1                              ;                                       4876
000076  156001  000011                      BISB    11(R0),R1
000102  005701                            TST     R1
000104  001003                            BNE     2$
000106  004737  000000V                     JSR     PC,MSCP.RSP                     ;                                       4879
000112  000421                            BR      4$                              ;                                       4876
000114  020127  000002              2$:     CMP     R1,#2
000120  001003                            BNE     3$
000122  004737  000000V                     JSR     PC,DUP.RSP                      ;                                       4880
000126  000413                            BR      4$                              ;                                       4876
000130  013746  000000G              3$:     MOV     IRDRX.ADDR,-(SP)                ;                                       4882
000134  010146                            MOV     R1,-(SP)
000136  012746  000000G                     MOV     #DBM20,-(SP)
000142  012746  000003                      MOV     #3,-(SP)
000146  010600                            MOV     SP,R0                           ; SP,*
000150  104417                            TRAP    17
000152  062706  000010                      ADD     #10,SP
000156  013700  000000G              4$:     MOV     IPKT.ADDR,R0                    ;                                       4886
000162  012760  000074  000006             MOV     #74,6(R0)
000170  013701  000110'                     MOV     IDCT.ADDR,R1                    ;                                       4887
000174  010102                            MOV     R1,R2
000176  062702  000014                      ADD     #14,R2
000202  062712  000002                      ADD     #2,(R2)
000206  016072  000002  000000             MOV     2(R0),@0(R2)                    ;                                       4888
000214  052772  140000  000000             BIS     #-40000,@0(R2)                  ;                                       4889
000222  062712  000002                      ADD     #2,(R2)                         ;                                       4890
000226  021261  000006                      CMP     (R2),6(R1)                      ;                                       4892
000232  101670                            BLOS    1$
000234  016112  000004                      MOV     4(R1),(R2)                      ;                                       4894
000240  000665                            BR      1$                              ;                                       4871
000242  013700  000104'              5$:     MOV     ICOM.ADDR,R0                    ;                                       4898
000246  005060  000006                      CLR     6(R0)
000252  000207                            RTS     PC                              ;                                       4852

; Routine Size:  86 words,      Routine Base:  $CODE$ + 21752
; Maximum stack depth per invocation:  9 words
```

```
4900  !^
4901  ROUTINE DUP_RSP : NOVALUE =
4902
4903  !+
4904  !       THIS ROUTINE IS CALLED BY POLL_RRING FOR EACH DUP RESPONSE
4905  !       ITS GENERAL PURPOSE IS TO ACT ON A DATAGRAM OR SEQUENTIAL MESSAGE.
4906  !       IF THE MESSAGE TYPE IS SEQUENTIAL, THE ROUTINE COPIES THE
4907  !       CONTENTS OF THE MESSAGE ENVELOPE INTO A RETURN PACKET SO THAT THE
4908  !       ENVELOPE CAN BE RETURNED TO THE CONTROLLER.
4909  !
4910  !       IMPLICIT INPUTS:
4911  !               ICTLR - INTERRUPTING CONTROLLER NUMBER
4912  !               IPKT_ADDR - ADDRESS OF MSCP ENVELOPE CONTAINING RESPONSE
4913  !-
4914      begin
4915
4916      local
4917          R_INDEX : signed word,
4918          SRC_ADDR,
4919          DST_ADDR;
4920          R_ADDR : ref block [RP_LEN, word] field (RP_FIELDS);
4921      !PRINTX (DER34);
4922
4923      incr COUNT from 0 to PKT_CNT - 1 do
4924
4925          if (.MSCP_PKT [.COUNT, CRN_LO] eql .IPKT_ADDR [CRN_LO]) and     ! IF THIS IS THE ASSOC CMD
4926             (.MSCP_PKT [.COUNT, CRN_HI] eql .IPKT_ADDR [CRN_HI]) and
4927             (.MSCP_PKT [.COUNT, PKT_LO] neqa .IPKT_ADDR [PKT_LO]) and
4928             ((.MSCP_PKT [.COUNT, OPCODE] and OP_END) neq OP_END) and
4929             (.MSCP_PKT [.COUNT, MSGTYP] eql MT_SEQ) and
4930             (.MSCP_PKT [.COUNT, CONNID] eql CID_DUP) and
4931             ((.IPKT_ADDR [OPCODE] and OP_END) eql OP_END)
4932          then
4933              begin
4934              P_INDEX = .COUNT;                          ! SET PKT NUMBER
4935              exitloop;
4936              end;
4937
4938      if .P_INDEX lss 0                                  ! IF COMMAND NOT FOUND
4939      then
4940          begin
4941          PRINTF (DBM108, .IPKT_ADDR [CRN_LO]);          ! UNKNOWN COMMAND REF. NUMBER
4942          return;
4943          end;
4944
4945      if (R_INDEX = GET_RETPKT (.ICTLR)) lss 0           ! IF RETPKT IS NOT AVAILABLE
4946      then
4947          PRINTF (DBM112)                                ! "DUP-RSP: RETPKT NOT AVAILABLE"
4948      else
4949          begin
4950          SRC_ADDR = .IPKT_ADDR + 6;                     ! SET UP COPY (SKIP OVER PKT DESC)
4951          R_ADDR = DST_ADDR = RETPKT + (.R_INDEX * RP_LEN * 2);   ! START OF ALLOCATED RETPKT
4952
```

```
;          4953          incr COUNT from 1 to RP_LEN do
;          4954              begin
;          4955              .DST_ADDR = ..SRC_ADDR;                 ! COPY 1 WORD
;          4956              DST_ADDR = .DST_ADDR + 2;               ! ADVANCE DESTINATION ADDR
;          4957              SRC_ADDR = .SRC_ADDR + 2;               ! ADVANCE SOURCE ADDR
;          4958              end;                                    ! COPY LOOP
;          4959
;          4960          R_ADDR [CTLR] = .ICTLR;                     ! LOAD CONTROLLER NUMBER INTO PACKET
;          4961          IN_IODQ (.R_INDEX);                         ! PUT RETPKT INDEX INTO IODQ
;          4962          end;                                        ! IF RETPKT WAS ALLOCATED
;          4963
;          4964
;          4965      if .P_INDEX geq 0                               ! IF ASSOC CMD PKT WAS FOUND
;          4966      then
;          4967          PUT_PKT (.P_INDEX);                         ! RETURN COMMAND PACKET TO POOL
;          4968
;          4969      end;                                            ! ROUTINE DUP_RSP

                                    .SBTTL  DUP.RSP RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G          DUP.RSP:JSR     R1,$SAVE4                       ;                       4901
000004  013701  000000G                  MOV     IPKT.ADDR,R1                    ;                       4925
000010  005002                          CLR     R2                              ; COUNT                 4923
000012  010246                  1$:     MOV     R2,-(SP)                        ; COUNT,*               4925
000014  012746  000104                  MOV     #104,-(SP)
000020  004737  000000G                 JSR     PC,BL$MUL
000024  022626                          CMP     (SP)+,(SP)+
000026  026061  000012G 000012          CMP     MSCP.PKT+12(R0),12(R1)
000034  001030                          BNE     2$
000036  026061  000014G 000014          CMP     MSCP.PKT+14(R0),14(R1)          ;                       4926
000044  001024                          BNE     2$
000046  026011  000000G                 CMP     MSCP.PKT(R0),(R1)               ;                       4927
000052  001421                          BEQ     2$
000054  105760  000022G                 TSTB    MSCP.PKT+22(R0)                 ;                       4928
000060  100416                          BMI     2$
000062  132760  000360  000010G        BITB    #360,MSCP.PKT+10(R0)            ;                       4929
000070  001012                          BNE     2$
000072  126027  000011G 000002          CMPB    MSCP.PKT+11(R0),#2              ;                       4930
000100  001006                          BNE     2$
000102  105761  000022                  TSTB    22(R1)                          ;                       4931
000106  100003                          BPL     2$
000110  010237  000000G                 MOV     R2,P.INDEX                      ; COUNT,*               4934
000114  000406                          BR      3$                              ;                       4933
000116  005202                  2$:     INC     R2                              ; COUNT                 4923
000120  020227  000013                  CMP     R2,#13                          ; COUNT,*
000124  003732                          BLE     1$
000126  005737  000000G                 TST     P.INDEX                         ;                       4938
000132  002013                          BGE     4$
000134  016146  000012          3$:     MOV     12(R1),-(SP)                    ;                       4941
000140  012746  000000G                 MOV     #DBM108,-(SP)
000144  012746  000002                  MOV     #2,-(SP)
000150  010600                          MOV     SP,R0                           ; SP,*
000152  104417                          TRAP    17
```

```
000154  062706  000006                          ADD     #6,SP                           ;                               4938
000160  000207                                  RTS     PC                              ;                               4940
000162  013746  000156'                 4$:     MOV     ICTLR,-(SP)                     ;                               4945
000166  004737  000000G                         JSR     PC,GET.RETPKT
000172  010004                                  MOV     R0,R4                           ; *,R.INDEX
000174  005726                                  TST     (SP)+
000176  005704                                  TST     R4                              ; R.INDEX
000200  002007                                  BGE     5$
000202  012746  000000G                         MOV     #DBM112,-(SP)                   ;                               4947
000206  012746  000001                         MOV     #1,-(SP)
000212  010600                                  MOV     SP,R0                           ; SP,*
000214  104417                                  TRAP    17
000216  000435                                  BR      7$                              ;                               4945
000220  013702  000000G                 5$:     MOV     IPKT.ADDR,R2                    ; *,SRC.ADDR                    4950
000224  062702  000006                         ADD     #6,R2                           ; *,SRC.ADDR
000230  010446                                  MOV     R4,-(SP)                        ; R.INDEX,*                     4951
000232  012746  000060                         MOV     #60,-(SP)
000236  004737  000000G                         JSR     PC,BL$MUL
000242  062700  000000G                         ADD     #RETPKT,R0
000246  010003                                  MOV     R0,R3                           ; *,DST.ADDR
000250  012701  000030                         MOV     #30,R1                          ; *,COUNT                       4953
000254  012223                         6$:     MOV     (R2)+,(R3)+                     ; SRC.ADDR,DST.ADDR             4955
000256  005301                                  DEC     R1                              ; COUNT                         4953
000260  001375                                  BNE     6$
000262  013701  000156'                         MOV     ICTLR,R1                        ;                               4960
000266  042701  177760                         BIC     #177760,R1
000272  142760  000017  000002                 BICB    #17,2(R0)                       ; *,*(R.ADDR)
000300  150160  000002                         BISB    R1,2(R0)                        ; *,*(R.ADDR)
000304  010416                                  MOV     R4,(SP)                         ; R.INDEX,*                     4961
000306  004737  000000G                         JSR     PC,IN.IODQ
000312  013700  000000G                 7$:     MOV     P.INDEX,R0                      ;                               4965
000316  002403                                  BLT     8$
000320  010016                                  MOV     R0,(SP)                         ;                               4967
000322  004737  000000G                         JSR     PC,PUT.PKT
000326  022626                         8$:     CMP     (SP)+,(SP)+                     ;                               4914
000330  000207                                  RTS     PC                              ;                               4901

; Routine Size:  109 words,     Routine Base:  $CODE$ + 22226
; Maximum stack depth per invocation:  10 words


;         4970
```

```
;          4971  routine MSCP_RSP : novalue =
;          4972
;          4973  !+
;          4974  !       THIS ROUTINE IS CALLED BY POLL_RRING FOR EACH RESPONSE MESSAGE
;          4975  !       WHICH HAS A CONNECTION ID INDICATING A DISK MSCP ORIGINATOR
;          4976  !       (I.E., ALL EXCEPT DUP RESPONSES). ITS PURPOSE IS TO PASS
;          4977  !       CONTROL TO THE APPROPRIATE ROUTINE BASED ON THE MESSAGE TYPE
;          4978  !       FIELD (SEQUENTIAL, DATAGRAM, OR CREDIT NOTIFICATION).
;          4979  !
;          4980  !       IMPLICIT INPUTS:
;          4981  !               IPKT_ADDR - ADDRESS OF MSCP PACKET CONTAINING RESPONSE
;          4982  !                           MESSAGE
;          4983  !-
;          4984
;          4985      selectoneu .IPKT_ADDR [MSGTYP] of
;          4986          set
;          4987
;          4988          [MT_SEQ] :      SEQUEN ();
;          4989
;          4990          [MT_DG] :       DATAGM ();
;          4991
;          4992          [otherwise] :   PRINTF (DBM21, .IPKT_ADDR [MSGTYP]);    ! 'MESSAGE TYPE XX RECEIVED"
;          4993          tes;


                                        .SBTTL  MSCP.RSP RDRX INTERRUPT SERVICE ROUTINES
000000  010146                  MSCP.RSP:
                                        MOV     R1,-(SP)                        ;                       4971
000002  013700  000000G                 MOV     IPKT.ADDR,R0                    ;                       4985
000006  116001  000010                  MOVB    10(R0),R1
000012  006201                          ASR     R1
000014  006201                          ASR     R1
000016  006201                          ASR     R1
000020  006201                          ASR     R1
000022  042701  177760                  BIC     #177760,R1
000026  001003                          BNE     1$
000030  004737  000000V                 JSR     PC,SEQUEN                       ;                       4988
000034  000417                          BR      3$                              ;                       4985
000036  020127  000001          1$:     CMP     R1,#1
000042  001003                          BNE     2$
000044  004737  000000V                 JSR     PC,DATAGM                       ;                       4990
000050  000411                          BR      3$                              ;                       4985
000052  010146                  2$:     MOV     R1,-(SP)                        ;                       4992
000054  012746  000000G                 MOV     #DBM21,-(SP)
000060  012746  000002                  MOV     #2,-(SP)
000064  010600                          MOV     SP,R0                           ; SP,*
000066  104417                          TRAP    17
000070  062706  000006                  ADD     #6,SP
000074  012601                  3$:     MOV     (SP)+,R1                        ;                       4971
000076  000207                          RTS     PC

; Routine Size:  32 words,      Routine Base:  $CODE$ + 22560
; Maximum stack depth per invocation:  6 words
```

```
;        4994  routine SEQUEN : novalue =
;        4995
;        4996  !+
;        4997  !       THIS ROUTINE IS CALLED BY MSCP_RSP FOR EACH DISK MSCP RESPONSE MESSAGE
;        4998  !       WITH THE "SEQUENTIAL" MESSAGE TYPE. ITS GENERAL PURPOSE IS TO COPY THE
;        4999  !       CONTENTS OF THE MESSAGE PACKET INTO A RETURN PACKET SO THAT THE
;        5000  !       PACKET CAN BE RETURNED TO THE CONTROLLER.   IN ADDITION,
;        5001  !       IF THE COMMAND WAS AN I/O TRANSFER (READ, WRITE, OR ACCESS), THEN SOME
;        5002  !       FIELDS OF THE COMMAND PACKET ARE COPIED INTO THE RETURN PACKET.
;        5003  !
;        5004  !       IMPLICIT INPUTS:
;        5005  !               ICTLR - INTERRUPTING CONTROLLER NUMBER
;        5006  !               IPKT_ADDR - ADDRESS OF MSCP PACKET CONTAINING RESPONSE
;        5007  !-
;        5008
;        5009      begin
;        5010
;        5011      local
;        5012          R_INDEX : signed word,
;        5013          SRC_ADDR,
;        5014          DST_ADDR,
;        5015          R_ADDR : ref block [RP_LEN, word] field (RP_FIELDS),
;        5016          TEMP_UNIT,
;        5017          SFT_ERR_PRINTED : byte initial (byte (FALSE));
;        5018
;        5019      incr COUNT from 0 to PKT_CNT - 1 do
;        5020
;        5021          if (.MSCP_PKT [.COUNT, CRN_LO] eql .IPKT_ADDR [CRN_LO]) and     ! IF THIS IS THE ASSOC CMD
;        5022             (.MSCP_PKT [.COUNT, CRN_HI] eql .IPKT_ADDR [CRN_HI]) and
;        5023             (.MSCP_PKT [.COUNT, PKT_LO] neqa .IPKT_ADDR [PKT_LO]) and
;        5024             ((.MSCP_PKT [.COUNT, OPCODE] and OP_END) neq OP_END) and
;        5025             (.MSCP_PKT [.COUNT, MSGTYP] eql MT_SEQ) and
;        5026             ((.IPKT_ADDR [OPCODE] and OP_END) eql OP_END) and
;        5027             (.PKT_USE [.count] eql .ICTLR)                             ! don't want old packets from other controll
;        5028          then
;        5029              begin
;        5030              P_INDEX = .COUNT;                      ! SET PKT NUMBER
;        5031              exitloop;
;        5032              end;
;        5033
;        5034      if .P_INDEX lss 0                              ! IF COMMAND NOT FOUND
;        5035      then
;        5036          begin
;        5037          PRINTF (DBM108, .IPKT_ADDR [CRN_LO]);      ! UNKNOWN COMMAND REF. NUMBER
;        5038          return;
;        5039          end;
;        5040
;        5041      if (R_INDEX = GET_RETPKT (.ICTLR)) lss 0       ! IF RETPKT IS NOT AVAILABLE
;        5042      then
;        5043  !       PRINTF (DBM22)                             ! "SEQUEN: RETPKT NOT AVAILABLE"
;        5044      else
;        5045          begin
;        5046          SRC_ADDR = .IPKT_ADDR + 6;                 ! SET UP COPY (SKIP OVER PKT DESC)
```

```
;         5047          R_ADDR = DST_ADDR = RETPKT + (.R_INDEX * RP_LEN * 2);   ! START OF ALLOCATED RETPKT
;         5048
;         5049          incr COUNT from 1 to RP_LEN do
;         5050              begin
;         5051              .DST_ADDR = ..SRC_ADDR;                 ! COPY 1 WORD
;         5052              DST_ADDR = .DST_ADDR + 2;               ! ADVANCE DESTINATION ADDR
;         5053              SRC_ADDR = .SRC_ADDR + 2;               ! ADVANCE SOURCE ADDR
;         5054
;         5055              if .IPKT_ADDR [OPCODE] eql (OP_ONL or OP_END)       ! IF THIS IS THE ONLINE END MESSAGE
;         5056              then
;         5057
;         5058                  if .COUNT eql 10                    ! SKIP OVER RESERVED WORDS
;         5059                  then
;         5060                      SRC_ADDR = .SRC_ADDR + 4;       !       IN ONLINE END - MESSAGE
;         5061
;         5062              end;                                    ! COPY LOOP
;         5063
;         5064          R_ADDR [CTLR] = .ICTLR;                     ! LOAD CONTROLLER NUMBER INTO PACKET
;         5065
;         5066          if .P_INDEX geq 0                           ! IF ASSOC. CMD PKT WAS FOUND
;         5067          then
;         5068
;         5069              if (.IPKT_ADDR [OPCODE] eql (OP_RD or OP_END)) or   ! IF END MESSAGE IS
;         5070                 (.IPKT_ADDR [OPCODE] eql (OP_WRT or OP_END)) or  !       READ, WRITE, OR
;         5071                 (.IPKT_ADDR [OPCODE] eql (OP_ACC or OP_END))     !       ACCESS
;         5072              then
;         5073                  begin
;         5074                  R_ADDR [CMDMOD] = .MSCP_PKT [.P_INDEX, MODIFY]; ! COPY
;         5075                  R_ADDR [CBCNT_LO] = .MSCP_PKT [.P_INDEX, BC_LO];! RELEVANT
;         5076                  R_ADDR [CBCNT_HI] = .MSCP_PKT [.P_INDEX, BC_HI];! FIELDS
;         5077                  R_ADDR [LBN_LO] = .MSCP_PKT [.P_INDEX, LBN_L];  ! FROM
;         5078                  R_ADDR [LBN_HI] = .MSCP_PKT [.P_INDEX, LBN_H];  ! COMMAND
;         5079                  R_ADDR [BUFF_0] = .MSCP_PKT [.P_INDEX, BUF_0];  ! PACKET
;         5080                  R_ADDR [BUFF_1] = .MSCP_PKT [.P_INDEX, BUF_1];  ! TO RETPKT
;         5081                  end;                                            ! IF ENDCODE WAS READ, WRITE, OR ACCESS
;         5082
;         5083              IN_IODQ (.R_INDEX);                                 ! PUT RETPKT INDEX INTO IODQ
;         5084              end;                                                ! IF RETPKT WAS ALLOCATED
;         5085
;         5086          if (.IPKT_ADDR [STATUS_CODE] neq  ST_SUC) or
;         5087             (.IPKT_ADDP [STATUS_SUBCODE] neq 0)
;         5088          then
;         5089              LAST_PKT [.ICTLR, LAST_HRD_ERR] = HRD_OCCURED           ! SAVE ERROR CONDITION
;         5090          else
;         5091              LAST_PKT [.ICTLR, LAST_HRD_ERR] = HRD_NOT_OCCURED;      !
;         5092
;         5093          LAST_PKT [.ICTLR, LAST_CRN_LO] = .IPKT_ADDR [CRN_LO];       ! SAVE COMMAND REFERENCE NUMBER
;         5094          LAST_PKT [.ICTLR, LAST_CRN_HI] = .IPKT_ADDR [CRN_HI];       !
;         5095
;         5096          incr index from 0 to EP_CNT - 1 do                          ! IF CORRESPONDING REF NUM HAD AN ERROR-LOG
;         5097
;         5098              if (.ELOG_PKT [.index, EL_CNTR] eql .ICTLR) and
;         5099                 (.ELOG_PKT [.index, EL_CRN_LO] eql .IPKT_ADDR [CRN_LO]) and
```

```
5100            (.ELOG_PKT [.index, EL_CRN_HI] eql .IPKT_ADDR [CRN_HI]) and
5101            (.ELOG_PKT [.index, EL_CONTENTS] eql FULL)
5102        then
5103            begin
5104
5105            if .LAST_PKT [.ICTLR, LAST_HRD_ERR] eql HRD_NOT_OCCURED     ! IF SOFT ERROR OCCURED
5106            then
5107
5108                if .ELOG_PKT [.index, EL_FORMAT] lequ 4
5109                then
5110                    begin
5111                    SOFT_ERROR (.index);                                ! UPATE SOFT ERROR COUNT
5112                    TEMP_UNIT = .L$LUN;                                 ! SAVE UNIT NUMBER AS KNOWN TO DRS
5113
5114                    incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
5115
5116                        if (.ICST_ADDR [.OFFSET, D_DISK_NUM] eql .ELOG_PKT [.index, EL_DK_NUM]) and
5117                           (.ICST_ADDR [.OFFSET, D_PRES] eql PRESENT)
5118                        then
5119                            begin
5120                            L$LUN = .ICST_ADDR [.OFFSET, D_UNIT];       ! CORECT UNIT NUMBER FOR ERROR MESSAGE
5121                            exitloop;
5122                            end;
5123
5124                    case .ELOG_PKT [.index, EL_FORMAT] from 0 to 4 of
5125                        set
5126
5127                        [0]:    ERRSOFT (60, 0, 0);                     ! CONTROLLER ERROR
5128
5129                        [1]:    ERRSOFT (61, 0, 0);                     ! HOST MEMORY ACCESS ERROR
5130
5131                        [2]:    ERRSOFT (62, 0, 0);                     ! DISK TRANSFER ERROR
5132
5133                        [3]:    ERRSOFT (63, 0, 0);                     ! SDI ERROR
5134
5135                        [4]:    ERRSOFT (64, 0, 0);                     ! SMALL DISK ERROR
5136                        tes;
5137
5138                    L$LUN = .TEMP_UNIT;                                 ! RESTORE UNIT NUMBER
5139                    SFT_ERR_PRINTED = TRUE;                             ! SOFT ERROR PRINTOUT OCCURED
5140                    end
5141
5142                else
5143                    PRINTF (DBM109, .ELOG_PKT [.index, EL_FORMAT]);     ! UNKNOWN ERROR-LOG FORMAT
5144
5145            if not (.SFT_ERR_PRINTED)
5146            then
5147                PRINTB (CRLF);                                          ! EXTRA CARRIAGE-RETURN/LINE-FEED
5148
5149            EMS_EL (.index);                                            ! PRINT ERROR-LOG CONTENTS
5150            end;
5151
5152    if .P_INDEX geq 0                                                   ! IF ASSOC CMD PKT WAS FOUND
```

```
;         5153      then
;         5154          PUT_PKT (.P_INDEX);                       ! RETURN COMMAND PACKET TO POOL
;         5155
;         5156      end;                                          ! ROUTINE DISK_RSP


                                .SBTTL  SEQUEN RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G       SEQUEN: JSR     R1,$SAVE5                       ;                       4994
000004  105046                       CLRB    -(SP)                           ; SFT.ERR.PRINTED       5009
000006  013701  000000G               MOV     IPKT.ADDR,R1                    ;                       5021
000012  005002                       CLR     R2                              ; COUNT                 5019
000014  010246               1$:     MOV     R2,-(SP)                        ; COUNT,*               5021
000016  012746  000104               MOV     #104,-(SP)
000022  004737  000000G               JSR     PC,BL$MUL
000026  022626                       CMP     (SP)+,(SP)+
000030  026061  000012G 000012       CMP     MSCP.PKT+12(R0),12(R1)
000036  001031                       BNE     2$
000040  026061  000014G 000014       CMP     MSCP.PKT+14(R0),14(R1)          ;                       5022
000046  001025                       BNE     2$
000050  026011  000000G               CMP     MSCP.PKT(R0),(R1)               ;                       5023
000054  001422                       BEQ     2$
000056  105760  000022G               TSTB    MSCP.PKT+22(R0)                 ;                       5024
000062  100417                       BMI     2$
000064  132760  000360  000010G      BITB    #360,MSCP.PKT+10(R0)            ;                       5025
000072  001013                       BNE     2$
000074  105761  000022               TSTB    22(R1)                          ;                       5026
000100  100010                       BPL     2$
000102  116200  000000G               MOVB    PKT.USE(R2),R0                  ; *(COUNT),*            5027
000106  020037  000156'               CMP     R0,ICTLR
000112  001003                       BNE     2$
000114  010237  000000G               MOV     R2,P.INDEX                      ; COUNT,*               5030
000120  000406                       BR      3$                              ;                       5029
000122  005202               2$:     INC     R2                              ; COUNT                 5019
000124  020227  000013               CMP     R2,#13                          ; COUNT,*
000130  003731                       BLE     1$
000132  005737  000000G               TST     P.INDEX                         ;                       5034
000136  002014               3$:     BGE     4$
000140  016146  000012               MOV     12(R1),-(SP)                    ;                       5037
000144  012746  000000G               MOV     #DBM108,-(SP)
000150  012746  000002               MOV     #2,-(SP)
000154  010600                       MOV     SP,R0                           ; SP,*
000156  104417                       TRAP    17
000160  062706  000006               ADD     #6,SP                           ;                       5034
000164  000137  024104'               JMP     30$                             ;                       5036
000170  013746  000156'       4$:     MOV     ICTLR,-(SP)                     ;                       5041
000174  004737  000000G               JSR     PC,GET.RETPKT
000200  010005                       MOV     R0,R5                           ; *,R.INDEX
000202  005726                       TST     (SP)+
000204  005705                       TST     R5                              ; R.INDEX
000206  002526                       BLT     9$
000210  013704  000000G               MOV     IPKT.ADDR,R4                    ; *,SRC.ADDR            5046
000214  062704  000006               ADD     #6,R4                           ; *,SRC.ADDR
000220  010546                       MOV     R5,-(SP)                        ; R.INDEX,*             5047
```

```
000222  012746  000060                        MOV     #60,-(SP)
000226  004737  000000G                       JSR     PC,BL$MUL
000232  062700  000000G                       ADD     #RETPKT,R0
000236  010003                                MOV     R0,R3          ; *,DST.ADDR
000240  010002                                MOV     R0,R2          ; *,R.ADDR
000242  013701  000000G                       MOV     IPKT.ADDR,R1   ;                        5055
000246  012700  000001                        MOV     #1,R0          ; *,COUNT                5049
000252  012423                        5$:     MOV     (R4)+,(R3)+    ; SRC.ADDR,DST.ADDR      5051
000254  126127  000022  000211               CMPB    22(R1),#211    ;                        5055
000262  001005                                BNE     6$
000264  020027  000012                        CMP     R0,#12         ; COUNT,*                5058
000270  001002                                BNE     6$
000272  062704  000004                        ADD     #4,R4          ; *,SRC.ADDR             5060
000276  005200                        6$:     INC     R0             ; COUNT                  5049
000300  020027  000030                        CMP     R0,#30         ; COUNT,*
000304  003762                                BLE     5$
000306  013700  000156'                       MOV     ICTLR,R0       ;                        5064
000312  042700  177760                        BIC     #177760,R0
000316  142762  000017  000002               BICB    #17,2(R2)      ; *,*(R.ADDR)
000324  150062  000002                        BISB    R0,2(R2)       ; *,*(R.ADDR)
000330  013704  000000G                       MOV     P.INDEX,R4     ;                        5066
000334  002447                                BLT     8$
000336  005000                                CLR     R0             ;                        5069
000340  156100  000022                        BISB    22(R1),R0
000344  020027  000241                        CMP     R0,#241
000350  001406                                BEQ     7$
000352  020027  000242                        CMP     R0,#242        ;                        5070
000356  001403                                BEQ     7$
000360  020027  000220                        CMP     R0,#220        ;                        5071
000364  001033                                BNE     8$
000366  010416                        7$:     MOV     R4,(SP)        ;                        5074
000370  012746  000104                        MOV     #104,-(SP)
000374  004737  000000G                       JSR     PC,BL$MUL
000400  016062  000024G 000012               MOV     MSCP.PKT+24(R0),12(R2) ; *,*(R.ADDR)
000406  016062  000026G 000044               MOV     MSCP.PKT+26(R0),44(R2) ; *,*(R.ADDR)     5075
000414  016062  000030G 000046               MOV     MSCP.PKT+30(R0),46(R2) ; *,*(R.ADDR)     5076
000422  016062  000046G 000050               MOV     MSCP.PKT+46(R0),50(R2) ; *,*(R.ADDR)     5077
000430  016062  000050G 000052               MOV     MSCP.PKT+50(R0),52(R2) ; *,*(R.ADDR)     5078
000436  016062  000032G 000024               MOV     MSCP.PKT+32(R0),24(R2) ; *,*(R.ADDR)     5079
000444  016062  000034G 000026               MOV     MSCP.PKT+34(R0),26(R2) ; *,*(R.ADDR)     5080
000452  005726                                TST     (SP)+          ;                        5073
000454  010516                        8$:     MOV     R5,(SP)        ; R.INDEX,*              5083
000456  004737  000000G                       JSR     PC,IN.IODQ
000462  022626                                CMP     (SP)+,(SP)+    ;                        5045
000464  013746  000156'                9$:     MOV     ICTLR,-(SP)    ;                        5089
000470  012746  000006                        MOV     #6,-(SP)
000474  004737  000000G                       JSR     PC,BL$MUL
000500  013702  000000G                       MOV     IPKT.ADDR,R2   ;                        5086
000504  012703  000024                        MOV     #24,R3
000510  060203                                ADD     R2,R3
000512  132713  000037                        BITB    #37,(R3)
000516  001003                                BNE     10$
000520  032713  177740                        BIT     #177740,(R3)   ;                        5087
```

```
000524  001404                          BEQ     11$
000526  012760  000001  000170'  10$:    MOV     #1,LAST.PKT(R0)                 ;                               5089
000534  000402                          BR      12$                             ;                               5086
000536  005060  000170'           11$:    CLR     LAST.PKT(R0)                    ;                               5091
000542  016260  000012  000172'  12$:    MOV     12(R2),LAST.PKT+2(R0)           ;                               5093
000550  016260  000014  000174'          MOV     14(R2),LAST.PKT+4(R0)           ;                               5094
000556  005003                          CLR     R3                              ; INDEX                         5096
000560  010316                  13$:    MOV     R3,(SP)                         ; INDEX,*                       5098
000562  012746  000102                  MOV     #102,-(SP)
000566  004737  000000G                 JSR     PC,BL$MUL
000572  010005                          MOV     R0,R5
000574  005726                          TST     (SP)+
000576  005000                          CLR     R0
000600  156500  000000G                 BISB    ELOG.PKT(R5),R0
000604  020037  000156'                 CMP     R0,ICTLR
000610  001170                          BNE     27$
000612  013700  000000G                 MOV     IPKT.ADDR,R0                    ;                               5099
000616  026560  000006G  000012        CMP     ELOG.PKT+6(R5),12(R0)
000624  001162                          BNE     27$
000626  026560  000010G  000014        CMP     ELOG.PKT+10(R5),14(R0)          ;                               5100
000634  001156                          BNE     27$
000636  126527  000001G  000001        CMPB    ELOG.PKT+1(R5),#1               ;                               5101
000644  001152                          BNE     27$
000646  013716  000156'                 MOV     ICTLR,(SP)                      ;                               5105
000652  012746  000006                  MOV     #6,-(SP)
000656  004737  000000G                 JSR     PC,BL$MUL
000662  005726                          TST     (SP)+
000664  005760  000170'                 TST     LAST.PKT(R0)
000670  001122                          BNE     25$
000672  126527  000016G  000004        CMPB    ELOG.PKT+16(R5),#4              ;                               5108
000700  101104                          BHI     24$
000702  010316                          MOV     R3,(SP)                         ; INDEX,*                       5111
000704  004737  000000V                 JSR     PC,SOFT.ERROR
000710  013702  000000G                 MOV     L$LUN,R2                        ; *,TEMP.UNIT                   5112
000714  012700  000006                  MOV     #6,R0                           ; *,OFFSET                      5114
000720  010004                  14$:    MOV     R0,R4                           ; OFFSET,*                      5116
000722  063704  000106'                 ADD     ICST.ADDR,R4
000726  016546  000012G                 MOV     ELOG.PKT+12(R5),-(SP)
000732  111401                          MOVB    (R4),R1
000734  042701  177774                  BIC     #177774,R1
000740  020126                          CMP     R1,(SP)+
000742  001012                          BNE     15$
000744  032714  040000                  BIT     #40000,(R4)                     ;                               5117
000750  001407                          BEQ     15$
000752  011401                          MOV     (R4),R1                         ;                               5120
000754  000301                          SWAB    R1
000756  042701  177760                  BIC     #177760,R1
000762  010137  000000G                 MOV     R1,L$LUN
000766  000405                          BR      16$                             ;                               5119
000770  062700  000012          15$:    ADD     #12,R0                          ; *,OFFSET                      5114
000774  020027  000044                  CMP     R0,#44                          ; OFFSET,*
001000  003747                          BLE     14$
001002  005000                  16$:    CLR     R0                              ;                               5124
```

```
001004  156500  000016G                 BISB    ELOG.PKT+16(R5),R0
001010  006300                          ASL     R0
001012  066007  000000'                 ADD     P.AAA(R0),PC                    ; Case dispatch
001016  104457                  18$:    TRAP    57                              ;                       5127
001020  000074                          .WORD   74
001022  000000                          .WORD   0
001024  000000                          .WORD   0
001026  000423                          BR      23$                             ;                       5124
001030  104457                  19$:    TRAP    57                              ;                       5129
001032  000075                          .WORD   75
001034  000000                          .WORD   0
001036  000000                          .WORD   0
001040  000416                          BR      23$                             ;                       5124
001042  104457                  20$:    TRAP    57                              ;                       5131
001044  000076                          .WORD   76
001046  000000                          .WORD   0
001050  000000                          .WORD   0
001052  000411                          BR      23$                             ;                       5124
001054  104457                  21$:    TRAP    57                              ;                       5133
001056  000077                          .WORD   77
001060  000000                          .WORD   0
001062  000000                          .WORD   0
001064  000404                          BR      23$                             ;                       5124
001066  104457                  22$:    TRAP    57                              ;                       5135
001070  000100                          .WORD   100
001072  000000                          .WORD   0
001074  000000                          .WORD   0
001076  010237  000000G         23$:    MOV     R2,L$LUN                        ; TEMP.UNIT,*           5138
001102  112766  000001  000004          MOVB    #1,4(SP)                        ; *,SFT.ERR.PRINTED     5139
001110  000412                          BR      25$                             ;                       5108
001112  005016                  24$:    CLR     (SP)                            ;                       5143
001114  116516  000016G                 MOVB    ELOG.PKT+16(R5),(SP)
001120  012746  000000G                 MOV     #DBM109,-(SP)
001124  012746  000002                  MOV     #2,-(SP)
001130  010600                          MOV     SP,R0                           ; SP,*
001132  104417                          TRAP    17
001134  022626                          CMP     (SP)+,(SP)+
001136  032766  000001  000004  25$:    BIT     #1,4(SP)                        ; *,SFT.ERR.PRINTED     5145
001144  001007                          BNE     26$
001146  012716  000000G                 MOV     #CRLF,(SP)                      ;                       5147
001152  012746  000001                  MOV     #1,-(SP)
001156  010600                          MOV     SP,R0                           ; SP,*
001160  104414                          TRAP    14
001162  005726                          TST     (SP)+
001164  010316                  26$:    MOV     R3,(SP)                         ; INDEX,*               5149
001166  004737  000000G                 JSR     PC,EMS.EL
001172  005203                  27$:    INC     R3                              ; INDEX                 5096
001174  020327  000013                  CMP     R3,#13                          ; INDEX,*
001200  003002                          BGT     28$
001202  000137  023440'                 JMP     13$
001206  013700  000000G         28$:    MOV     P.INDEX,R0                      ;                       5152
001212  002403                          BLT     29$
001214  010016                          MOV     R0,(SP)                         ;                       5154
```

```
001216  004737  000000G                        JSR     PC,PUT.PKT
001222  022626                          29$:    CMP     (SP)+,(SP)+             ;                       5009
001224  005726                          30$:    TST     (SP)+                   ;                       4994
001226  000207                                  RTS     PC

; Routine Size:  332 words,     Routine Base:  $CODE$ + 22660
; Maximum stack depth per invocation:  13 words


000000                                  .PSECT  $PLIT$,  RO ,  D

                                P.AAA:                                          ; CASE Table for SEQUEN+1012    5124
000000  000000                  17$:    .WORD   0                               ; [18$]
000002  000012                          .WORD   12                              ; [19$]
000004  000024                          .WORD   24                              ; [20$]
000006  000036                          .WORD   36                              ; [21$]
000010  000050                          .WORD   50                              ; [22$]
```

```
;         5157  routine DATAGM : novalue =
;         5158
;         5159  !+
;         5160  !       THIS ROUTINE HANDLES ALL DATAGRAM (ERROR LOG) MESSAGES RECEIVED FROM
;         5161  !       THE RDRX
;         5162  !
;         5163  !       IMPLICIT INPUTS:
;         5164  !               IPKT_ADDR - ADDRESS OF MSCP PACKET CONTAINING ERROR LOG
;         5165  !                           MESSAGE
;         5166  !               ICST_ADDR - ADDRESS OF THE INTERRUPTING CONTROLLER'S CST
;         5167  !-
;         5168
;         5169      begin
;         5170
;         5171      local
;         5172          index : signed word initial (-1),
;         5173          SAVE_ADDR : ref block [EP_LEN, word] field (EP_FIELDS),
;         5174          SRC_ADDR,
;         5175          DST_ADDR,
;         5176          TEMP_UNIT,
;         5177          SFT_ERR_PRINTED : byte initial (byte (FALSE));
;         5178
;         5179  !
;         5180  ! FIND AN EMPTY SLOT IN THE ERROR-LOG PACKET SAVE AREA
;         5181  !
;         5182
;         5183      incr COUNT from 0 to EP_CNT - 1 do
;         5184
;         5185          if .ELOG_PKT [.COUNT, EL_CONTENTS] eql EMPTY    ! IF EMPTY SLOT FOUND
;         5186          then
;         5187              begin
;         5188              index = .COUNT;                             ! SAVE INDEX INTO THE SAVE AREA
;         5189              exitloop;
;         5190              end;
;         5191
;         5192  !
;         5193  ! IF AN EMPTY SLOT FOUND, SAVE THE PACKET CONTENTS
;         5194  !
;         5195
;         5196      if .index geq 0
;         5197      then
;         5198          begin
;         5199          SAVE_ADDR = ELOG_PKT + (.index * EP_LEN * 2);   ! ADDRESS OF THE SAVE AREA
;         5200          SAVE_ADDR [EL_CONTENTS ] = FULL;                ! MARK IT FULL
;         5201          SAVE_ADDR [EL_CNTR] = .ICTLR;                   ! OWNERSHIP
;         5202          SRC_ADDR = .IPKT_ADDR + 6;                      ! SETUP COPY ADDRESSES
;         5203          DST_ADDR = .SAVE_ADDR + 2;                      !
;         5204
;         5205          incr COUNT from 1 to ((.IPKT_ADDR [MSGLEN] + 1) / 2) + 2 do
;         5206              begin
;         5207              .DST_ADDR = ..SRC_ADDR;                     ! COPY A WORD
;         5208              SRC_ADDR = .SRC_ADDR + 2;                   ! UPDATE ADDRESS POINTERS
;         5209              DST_ADDR = .DST_ADDR + 2;                   !
```

```
5210                end;
5211
5212            end
5213        else
5214            begin
5215            PRINTF (DBM110);                                    ! IF EMPTY SLOT NOT FOUND, PRINT MESSAGE
5216            return;
5217            end;
5218
5219    !
5220    ! CHECK IF THE CORRESPONDING RESPONSE HAS ALREADY BEEN RECEIVED
5221    !
5222
5223        if (.SAVE_ADDR [EL_CRN_LO] eql .LAST_PKT [.ICTLR, LAST_CRN_LO]) and
5224           (.SAVE_ADDR [EL_CRN_HI] eql .LAST_PKT [.ICTLR, LAST_CRN_HI])
5225        then
5226            begin
5227
5228            if .LAST_PKT [.ICTLR, LAST_HRD_ERR] eql HRD_NOT_OCCURED ! IF SOFT ERROR HAD OCCURED
5229            then
5230
5231                if .SAVE_ADDR [EL_FORMAT] lequ 4
5232                then
5233                    begin
5234                    SOFT_ERROR (.index);                        ! UPDATE SOFT ERROR COUNT
5235                    TEMP_UNIT = .L$LUN;                         ! SAVE UNIT NUMBER AS KNOWN TO DRS
5236
5237                    incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
5238
5239                        if (.ICST_ADDR [.OFFSET, D_DISK_NUM] eql .SAVE_ADDR [EL_DK_NUM]) and
5240                           (.ICST_ADDR [.OFFSET, D_PRES] eql PRESENT)
5241                        then
5242                            begin
5243                            L$LUN = .ICST_ADDR [.OFFSET, D_UNIT];   ! CORRECT UNIT NUMBER FOR ERROR MESSAGE
5244                            exitloop;
5245                            end;
5246
5247                    case .SAVE_ADDR [EL_FORMAT] from 0 to 4 of
5248                        set
5249
5250                        [0] :       ERRSOFT (60, 0, 0);         ! CONTROLLER ERROR
5251
5252                        [1] :       ERRSOFT (61, 0, 0);         ! HOST MEMORY ACCESS ERROR
5253
5254                        [2] :       ERRSOFT (62, 0, 0);         ! DISK TRANSFER ERROR
5255
5256                        [3] :       ERRSOFT (63, 0, 0);         ! SDI ERROR
5257
5258                        [4] :       ERRSOFT (64, 0, 0);         ! SMALL DISK ERROR
5259                        tes;
5260
5261                    L$LUN = .TEMP_UNIT;                         ! RESTORE UNIT NUMBER
5262                    SFT_ERR_PRINTED = TRUE;                     ! SOFT ERROR PRINTOUT OCCURED
```

```
;          5263                    end
;          5264
;          5265                else
;          5266                    PRINTF (DBM109, .SAVE_ADDR [EL_FORMAT]);        ! ERROR LOG FORMAT UNKNOWN
;          5267
;          5268            if not (.SFT_ERR_PRINTED)
;          5269            then
;          5270                PRINTB (CRLF);                                      ! EXTRA CARRIEGE-RETURN/LINE-FEED
;          5271
;          5272            EMS_EL (.index);                                        ! PRINT PACKET CONTENTS
;          5273            end;                                                    ! CORRESPONDING RESPONSE RECEIVED
;          5274
;          5275        end;


                                        .SBTTL  DATAGM RDRX INTERRUPT SERVICE ROUTINES
024110                                  .PSECT  $CODE$,  RO

000000  004137  000000G        DATAGM: JSR     R1,$SAVE5                       ;                                       5157
000004  012705  177777                  MOV     #-1,R5                          ; *,INDEX                               5169
000010  105046                          CLRB    -(SP)                           ; SFT.ERR.PRINTED
000012  005001                          CLR     R1                              ; COUNT                                 5183
000014  010146                  1$:     MOV     R1,-(SP)                        ; COUNT,*                               5185
000016  012746  000102                  MOV     #102,-(SP)
000022  004737  000000G                 JSR     PC,BL$MUL
000026  022626                          CMP     (SP)+,(SP)+
000030  105760  000001G                 TSTB    ELOG.PKT+1(RO)
000034  001002                          BNE     2$
000036  010105                          MOV     R1,R5                           ; COUNT,INDEX                           5188
000040  000405                          BR      3$                              ;                                       5187
000042  005201                  2$:     INC     R1                              ; COUNT                                 5183
000044  020127  000013                  CMP     R1,#13                          ; COUNT,*
000050  003761                          BLE     1$
000052  005705                          TST     R5                              ; INDEX                                 5196
000054  002446                  3$:     BLT     6$
000056  010546                          MOV     R5,-(SP)                        ; INDEX,*                               5199
000060  012746  000102                  MOV     #102,-(SP)
000064  004737  000000G                 JSR     PC,BL$MUL
000070  062700  000000G                 ADD     #ELOG.PKT,RO
000074  010004                          MOV     RO,R4                           ; *,SAVE.ADDR
000076  111764  000001                  MOVB    (PC),1(R4)                      ; *,*(SAVE.ADDR)                        5200
000102  113714  000156'                 MOVB    ICTLR,(R4)                      ; *,SAVE.ADDR                           5201
000106  013700  000000G                 MOV     IPKT.ADDR,RO                    ;                                       5202
000112  012702  000006                  MOV     #6,R2                           ; *,SRC.ADDR
000116  060002                          ADD     RO,R2                           ; *,SRC.ADDR
000120  012703  000002                  MOV     #2,R3                           ; *,DST.ADDR                            5203
000124  060403                          ADD     R4,R3                           ; SAVE.ADDR,DST.ADDR
000126  016016  000006                  MOV     6(RO),(SP)                      ;                                       5205
000132  005216                          INC     (SP)
000134  012746  000002                  MOV     #2,-(SP)
000140  004737  000000G                 JSR     PC,BL$DIV
000144  062700  000002                  ADD     #2,RO
000150  005001                          CLR     R1                              ; COUNT
```

```
000152  000401                          BR      5$
000154  012223                  4$:     MOV     (R2)+,(R3)+             ; SRC.ADDR,DST.ADDR                     5207
000156  005201                  5$:     INC     R1                      ; COUNT                                 5205
000160  020100                          CMP     R1,R0                   ; COUNT,*
000162  003774                          BLE     4$                                                              5198
000164  062706  000006                  ADD     #6,SP                   ;                                       5196
000170  000410                          BR      7$                      ;                                       5215
000172  012746  000000G         6$:     MOV     #DBM110,-(SP)           ;
000176  012746  000001                  MOV     #1,-(SP)
000202  010600                          MOV     SP,R0                   ; SP,*
000204  104417                          TRAP    17                                                              5196
000206  022626                          CMP     (SP)+,(SP)+             ;                                       5214
000210  000560                          BR      22$                     ;                                       5223
000212  013746  000156'        7$:     MOV     ICTLR,-(SP)             ;
000216  012746  000006                  MOV     #6,-(SP)
000222  004737  000000G                 JSR     PC,BL$MUL
000226  022626                          CMP     (SP)+,(SP)+
000230  026460  000006  000172'        CMP     6(R4),LAST.PKT+2(R0)    ; *(SAVE.ADDR),*
000236  001145                          BNE     22$
000240  026460  000010  000174'        CMP     10(R4),LAST.PKT+4(R0)   ; *(SAVE.ADDR),*                        5224
000246  001141                          BNE     22$
000250  005760  000170'                TST     LAST.PKT(R0)            ;                                       5228
000254  001120                          BNE     20$
000256  005001                          CLR     R1                      ;                                       5231
000260  156401  000016                  BISB    16(R4),R1               ; *(SAVE.ADDR),*
000264  020127  000004                  CMP     R1,#4
000270  101101                          BHI     18$
000272  010546                          MOV     R5,-(SP)                ; INDEX,*                               5234
000274  004737  000000V                 JSR     PC,SOFT.ERROR
000300  013703  000000G                 MOV     L$LUN,R3                ; *,TEMP.UNIT                           5235
000304  012700  000006                  MOV     #6,R0                   ; *,OFFSET                              5237
000310  010002                  8$:     MOV     R0,R2                   ; OFFSET,*                              5239
000312  063702  000106'                ADD     ICST.ADDR,R2
000316  016446  000012                  MOV     12(R4),-(SP)            ; *(SAVE.ADDR),*
000322  111246                          MOVB    (R2),-(SP)
000324  042716  177774                  BIC     #177774,(SP)
000330  022626                          CMP     (SP)+,(SP)+
000332  001012                          BNE     9$
000334  032712  040000                  BIT     #40000,(R2)             ;                                       5240
000340  001407                          BEQ     9$
000342  011246                          MOV     (R2),-(SP)              ;                                       5243
000344  000316                          SWAB    (SP)
000346  042716  177760                  BIC     #177760,(SP)
000352  012637  000000G                 MOV     (SP)+,L$LUN
000356  000405                          BR      10$                     ;                                       5242
000360  062700  000012          9$:     ADD     #12,R0                  ; *,OFFSET                              5237
000364  020027  000044                  CMP     R0,#44                  ; OFFSET,*
000370  003747                          BLE     8$
000372  006301                  10$:    ASL     R1                      ;                                       5247
000374  066107  000012'                ADD     P.AAB(R1),PC            ; Case dispatch
000400  104457                  12$:    TRAP    57                      ;                                       5250
000402  000074                          .WORD   74
000404  000000                          .WORD   0
```

```
000406  000000                          .WORD   0
000410  000423                          BR      17$                     ;                       5247
000412  104457                  13$:    TRAP    57                      ;                       5252
000414  000075                          .WORD   75
000416  000000                          .WORD   0
000420  000000                          .WORD   0
000422  000416                          BR      17$                     ;                       5247
000424  104457                  14$:    TRAP    57                      ;                       5254
000426  000076                          .WORD   76
000430  000000                          .WORD   0
000432  000000                          .WORD   0
000434  000411                          BR      17$                     ;                       5247
000436  104457                  15$:    TRAP    57                      ;                       5256
000440  000077                          .WORD   77
000442  000000                          .WORD   0
000444  000000                          .WORD   0
000446  000404                          BR      17$                     ;                       5247
000450  104457                  16$:    TRAP    57                      ;                       5258
000452  000100                          .WORD   100
000454  000000                          .WORD   0
000456  000000                          .WORD   0
000460  010337  000000G         17$:    MOV     R3,L$LUN                ; TEMP.UNIT,*           5261
000464  112766  000001  000002          MOVB    #1,2(SP)                ; *,SFT.ERR.PRINTED     5262
000472  000410                          BR      19$                     ;                       5231
000474  010146                  18$:    MOV     R1,-(SP)                ;                       5266
000476  012746  000000G                 MOV     #DBM109,-(SP)
000502  012746  000002                  MOV     #2,-(SP)
000506  010600                          MOV     SP,R0                   ; SP,*
000510  104417                          TRAP    17
000512  022626                          CMP     (SP)+,(SP)+
000514  005726                  19$:    TST     (SP)+                   ;                       5231
000516  032716  000001          20$:    BIT     #1,(SP)                 ; *,SFT.ERR.PRINTED     5268
000522  001007                          BNE     21$
000524  012746  000000G                 MOV     #CRLF,-(SP)             ;                       5270
000530  012746  000001                  MOV     #1,-(SP)
000534  010600                          MOV     SP,R0                   ; SP,*
000536  104414                          TRAP    14
000540  022626                          CMP     (SP)+,(SP)+
000542  010546                  21$:    MOV     R5,-(SP)                ; INDEX,*               5272
000544  004737  000000G                 JSR     PC,EMS.EL
000550  005726                          TST     (SP)+                   ;                       5226
000552  005726                  22$:    TST     (SP)+                   ;                       5157
000554  000207                          RTS     PC

; Routine Size:  183 words,     Routine Base:  $CODE$ + 24110
; Maximum stack depth per invocation:  12 words


000012                                  .PSECT  $PLIT$,  RO ,  D

                                P.AAB:                                  ; CASE Table for DATAGM+0374    5247
000012  000000                  11$:    .WORD   0                       ; [12$]
000014  000012                          .WORD   12                      ; [13$]
```

```
000016  000024                          .WORD   24              ; [14$]
000020  000036                          .WORD   36              ; [15$]
000022  000050                          .WORD   50              ; [16$]
```

```
;         5276  routine SOFT_ERROR (INDEX) : novalue =
;         5277
;         5278  !+
;         5279  !       THIS ROUTINE UPDATES THE SOFT ERROR COUNT IN THE TALLY TABLE FOR EACH
;         5280  !       ERROR LOG MESSAGE RECEIVED
;         5281  !
;         5282  !       IMPLICIT INPUTS:
;         5283  !               ICST_ADDR - ADDRESS OF THE INTERRUPTING CONTROLLER'S CST
;         5284  !-
;         5285
;         5286      begin
;         5287
;         5288      local
;         5289          FOUND: byte initial (byte (FALSE)),
;         5290          SOFT_OCCURED : byte initial (byte (FALSE)),
;         5291          UNIT: word,
;         5292          ERROR_CODE : byte,
;         5293          ERROR_SUB : word,
;         5294          TALLY_ADDR : ref block [TALLY_LEN, word] field (T_FIELDS),
;         5295          ELOG_ADDR : ref block [EP_LEN, word] field (EP_FIELDS);
;         5296
;         5297      ELOG_ADDR = ELOG_PKT + (.index * EP_LEN * 2);                          ! ADDRESS OF ERROR L
;         5298      ERROR_CODE = .ELOG_ADDR [EL_CODE];                                     ! ERROR CODE
;         5299      ERROR_SUB = .ELOG_ADDR [EL_SUBCODE];                                   ! ERROR SUBCODE
;         5300
;         5301      incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;         5302
;         5303          if (.ICST_ADDR [.OFFSET, D_PRES] eql PRESENT) and                  ! MAP DISK NUMBER TO
;         5304             (.ICST_ADDR [.OFFSET, D_DISK_NUM] eql .ELOG_ADDR [EL_DK_NUM])
;         5305          then
;         5306              begin
;         5307              FOUND = TRUE;
;         5308              UNIT = .ICST_ADDR [.OFFSET, D_UNIT];                           ! UNIT NUMBER OF DIS
;         5309              exitloop;
;         5310              end;
;         5311
;         5312  !   if (.ELOG_ADDR [EL_SUCCESS]) or
;         5313  !      (.ELOG_ADDR [EL_CONTINUE])
;         5314  !   then
;         5315          SOFT_OCCURED = TRUE;                                           ! SOFT ERROR FLAG
;         5316
;         5317      if .FOUND                                                          ! IF UNIT NUMBER FOU
;         5318      then
;         5319          begin
;         5320          TALLY_ADDR = TALLY + (.UNIT * TALLY_LEN * 2);                  ! ADDRESS OF TALLY T
;         5321
;         5322          if .SOFT_OCCURED                                               ! FOR SOFT ERRORRS
;         5323          then
;         5324              selectoneu .ERROR_CODE of
;         5325                  set
;         5326
;         5327                  [ST_MFE]:       TALLY_ADDR [ERR_SFT_SEK] = .TALLY_ADDR [ERR_SFT_SEK] + 1;       ! SOFT - MEDIA FORMA
;         5328
```

```
;         5329                    [ST_DAT]:       if .ERROR_SUB eql 2                                             ! SOFT - DATA
;         5330                                    then
;         5331                                        TALLY_ADDR [ERR_SFT_SEK] = .TALLY_ADDR [ERR_SFT_DAT] + 1
;         5332                                    else
;         5333                                        TALLY_ADDR [ERR_SFT_DAT] = .TALLY_ADDR [ERR_SFT_DAT] + 1;
;         5334
;         5335                    [ST_HST]:       TALLY_ADDR [ERR_SFT_HST] = .TALLY_ADDR [ERR_SFT_HST] + 1;       ! SOFT - HOST ACCESS
;         5336
;         5337                    [ST_CNT]:       C_ERR_TBL [.ICTLR, C_ERR_SFT] = .C_ERR_TBL [.ICTLR, C_ERR_SFT] + 1;
;         5338                                                                                                    ! SOFT - CONTROLLER
;         5339
;         5340                    [ST_DRV]:       if .ERROR_SUB eql 3                                             ! SOFT - DRIVE
;         5341                                    then
;         5342                                        TALLY_ADDR [ERR_SFT_SEK] = .TALLY_ADDR [ERR_SFT_SEK] + 1
;         5343                                    else
;         5344                                        TALLY_ADDR [ERR_SFT_DRV] = .TALLY_ADDR [ERR_SFT_DRV] + 1;
;         5345                    tes
;         5346            else
;         5347
;         5348                if (.ELOG_ADDR [EL_CRN_LO] eql 0) and
;         5349                   (.ELOG_ADDR [EL_CRN_HI] eql 0)
;         5350                then
;         5351                    selectoneu .ERROR_CODE of
;         5352                        set
;         5353
;         5354                        [ST_MFE]:   TALLY_ADDR [ERR_HRD_SEK] = .TALLY_ADDR [ERR_HRD_SEK] + 1;       ! HARD - MEDIA FORMA
;         5355
;         5356                        [ST_DAT]:   if .ERROR_SUB eql 2                                             ! HARD - DATA
;         5357                                    then
;         5358                                        TALLY_ADDR [ERR_HRD_SEK] = .TALLY_ADDR [ERR_HRD_SEK] + 1
;         5359                                    else
;         5360                                        TALLY_ADDR [ERR_HRD_DAT] = .TALLY_ADDR [ERR_HRD_DAT] + 1;
;         5361
;         5362                                [ST_HST]:   TALLY_ADDR [ERR_HRD_HST] = .TALLY_ADDR [ERR_HRD_HST] + 1;       ! HARD - HOST ACCESS
;         5363
;         5364                                [ST_CNT]:   C_ERR_TBL [.ICTLR, C_ERR_HRD] = .C_ERR_TBL [.ICTLR, C_ERR_HRD] + 1;
;         5365                                                                                                    ! HARD - CONTROLLER
;         5366
;         5367                                [ST_DRV]:   if .ERROR_SUB eql 3                                     ! HARD - DRIVE
;         5368                                            then
;         5369                                                TALLY_ADDR [ERR_HRD_SEK] = .TALLY_ADDR [ERR_HRD_SEK] + 1
;         5370                                            else
;         5371                                                TALLY_ADDR [ERR_HRD_DRV] = .TALLY_ADDR [ERR_HRD_DRV] + 1;
;         5372                                tes;
;         5373
;         5374                    end
;         5375
;         5376            else                                                                                    ! UNIT NOT FOUND
;         5377
;         5378                if .SOFT_OCCURED
;         5379                then
;         5380                    C_ERR_TBL [.ICTLR, C_ERR_SFT] = .C_ERR_TBL [.ICTLR, C_ERR_SFT] + 1
;         5381                else
```

```
;         5382              C_ERR_TBL [.ICTLR, C_ERR_HRD] = .C_ERR_TBL [.ICTLR, C_ERR_HRD] + 1;
;         5383
;         5384          end;                                                    ! ROUTINE SOFT_ERROR


                                        .SBTTL  SOFT.ERROR RDRX INTERRUPT SERVICE ROUTINES
024666                                  .PSECT  $CODE$,  RO

000000  004137  000000G          SOFT.ERROR:
                                        JSR     R1,$SAVE5                       ;                                       5276
000004  024646                          CMP     -(SP),-(SP)
000006  105001                          CLRB    R1                              ; FOUND                                 5286
000010  105066  000002                  CLRB    2(SP)                           ; SOFT.OCCURED
000014  016646  000022                  MOV     22(SP),-(SP)                    ; INDEX,*                               5297
000020  012746  000102                  MOV     #102,-(SP)
000024  004737  000000G                 JSR     PC,BL$MUL
000030  062700  000000G                 ADD     #ELOG.PKT,RO
000034  010004                          MOV     RO,R4                           ; *,ELOG.ADDR
000036  116400  000020                  MOVB    20(R4),RO                       ; *(ELOG.ADDR),*                        5298
000042  042700  177740                  BIC     #177740,RO
000046  105002                          CLRB    R2                              ; ERROR.CODE
000050  050002                          BIS     RO,R2                           ; *,ERROR.CODE
000052  016403  000020                  MOV     20(R4),R3                       ; *(ELOG.ADDR),ERROR.SUB                5299
000056  006203                          ASR     R3                              ; ERROR.SUB
000060  006203                          ASR     R3                              ; ERROR.SUB
000062  006203                          ASR     R3                              ; ERROR.SUB
000064  006203                          ASR     R3                              ; ERROR.SUB
000066  006203                          ASR     R3                              ; ERROR.SUB
000070  042703  174000                  BIC     #174000,R3                      ; *,ERROR.SUB
000074  012700  000006                  MOV     #6,RO                           ; *,OFFSET                              5301
000100  010005                  1$:     MOV     RO,R5                           ; OFFSET,*                              5303
000102  063705  000106'                 ADD     ICST.ADDR,R5
000106  032715  040000                  BIT     #40000,(R5)
000112  001420                          BEQ     2$
000114  016446  000012                  MOV     12(R4),-(SP)                    ; *(ELOG.ADDR),*                        5304
000120  111546                          MOVB    (R5),-(SP)
000122  042716  177774                  BIC     #177774,(SP)
000126  022626                          CMP     (SP)+,(SP)+
000130  001011                          BNE     2$
000132  112701  000001                  MOVB    #1,R1                           ; *,FOUND                               5307
000136  011546                          MOV     (R5),-(SP)                      ;                                       5308
000140  000316                          SWAB    (SP)
000142  042716  177760                  BIC     #177760,(SP)
000146  012666  000004                  MOV     (SP)+,4(SP)                     ; *,UNIT
000152  000405                          BR      3$                              ;                                       5306
000154  062700  000012          2$:     ADD     #12,RO                          ; *,OFFSET                              5301
000160  020027  000044                  CMP     RO,#44                          ; OFFSET,*
000164  003745                          BLE     1$
000166  112766  000001  000006  3$:     MOVB    #1,6(SP)                        ; *,SOFT.OCCURED                        5315
000174  006001                          ROR     R1                              ; FOUND                                 5317
000176  103154                          BCC     17$
000200  016616  000004                  MOV     4(SP),(SP)                      : UNIT,*                                5320
000204  012746  000070                  MOV     #70,-(SP)
```

```
000210  004737  000000G                        JSR     PC,BL$MUL
000214  062700  000000G                        ADD     #TALLY,R0
000220  032766  000001  000010                BIT     #1,10(SP)               ; *,SOFT.OCCURED                5322
000226  001455                                BEQ     9$
000230  120227  000005                        CMPB    R2,#5                   ; ERROR.CODE,*                  5324
000234  001444                                BEQ     7$                      ;                               5327
000236  120227  000010                        CMPB    R2,#10                  ; ERROR.CODE,*                  5324
000242  001014                                BNE     4$
000244  012701  000064                        MOV     #64,R1                  ;                               5331
000250  060001                                ADD     R0,R1                   ; TALLY.ADDR,*
000252  020327  000002                        CMP     R3,#2                   ; ERROR.SUB,*                   5329
000256  001065                                BNE     10$
000260  005004                                CLR     R4                      ;                               5331
000262  156104  000001                        BISB    1(R1),R4
000266  005204                                INC     R4
000270  110411                                MOVB    R4,(R1)
000272  000514                                BR      16$                     ;                               5329
000274  120227  000011          4$:           CMPB    R2,#11                  ; ERROR.CODE,*                  5324
000300  001003                                BNE     5$
000302  105260  000067                        INCB    67(R0)                  ; *(TALLY.ADDR)                 5335
000306  000506                                BR      16$                     ;                               5324
000310  120227  000012          5$:           CMPB    R2,#12                  ; ERROR.CODE,*
000314  001006                                BNE     6$
000316  013701  000156'                       MOV     ICTLR,R1                ;                               5337
000322  006301                                ASL     R1
000324  105261  000001G                       INCB    C.ERR.TBL+1(R1)
000330  000475                                BR      16$                     ;                               5324
000332  120227  000013          6$:           CMPB    R2,#13                  ; ERROR.CODE,*
000336  001072                                BNE     16$
000340  020327  000003                        CMP     R3,#3                   ; ERROR.SUB,*                   5340
000344  001003                                BNE     8$
000346  105260  000064          7$:           INCB    64(R0)                  ; *(TALLY.ADDR)                 5342
000352  000464                                BR      16$                     ;                               5340
000354  105260  000066          8$:           INCB    66(R0)                  ; *(TALLY.ADDR)                 5344
000360  000461                                BR      16$                     ;                               5324
000362  005764  000006          9$:           TST     6(R4)                   ; *(ELOG.ADDR)                  5348
000366  001056                                BNE     16$
000370  005764  000010                        TST     10(R4)                  ; *(ELOG.ADDR)                  5349
000374  001053                                BNE     16$
000376  120227  000005                        CMPB    R2,#5                   ; ERROR.CODE,*                  5351
000402  001443                                BEQ     14$                     ;                               5354
000404  120227  000010                        CMPB    R2,#10                  ; ERROR.CODE,*                  5351
000410  001013                                BNE     11$
000412  012701  000060                        MOV     #60,R1                  ;                               5358
000416  060001                                ADD     R0,R1                   ; TALLY.ADDR,*
000420  020327  000002                        CMP     R3,#2                   ; ERROR.SUB,*                   5356
000424  001002                                BNE     10$
000426  105211                                INCB    (R1)                    ;                               5358
000430  000435                                BR      16$                     ;                               5356
000432  105261  000001          10$:          INCB    1(R1)                   ;                               5360
000436  000432                                BR      16$                     ;                               5351
000440  120227  000011          11$:          CMPB    R2,#11                  ; ERROR.CODE,*
000444  001003                                BNE     12$
```

```
000446  105260  000063                          INCB    63(R0)                  ; *(TALLY.ADDR)                         5362
000452  000424                                  BR      16$                     ;                                       5351
000454  120227  000012                  12$:    CMPB    R2,#12                  ; ERROR.CODE,*
000460  001006                                  BNE     13$
000462  013701  000156'                          MOV     ICTLR,R1                ;                                       5364
000466  006301                                  ASL     R1
000470  105261  000000G                         INCB    C.ERR.TBL(R1)
000474  000413                                  BR      16$                     ;                                       5351
000476  120227  000013                  13$:    CMPB    R2,#13                  ; ERROR.CODE,*
000502  001010                                  BNE     16$
000504  020327  000003                          CMP     R3,#3                   ; ERROR.SUB,*                           5367
000510  001003                                  BNE     15$
000512  105260  000060                  14$:    INCB    60(R0)                  ; *(TALLY.ADDR)                         5369
000516  000402                                  BR      16$                     ;                                       5367
000520  105260  000062                  15$:    INCB    62(R0)                  ; *(TALLY.ADDR)                         5371
000524  005726                          16$:    TST     (SP)+                   ;                                       5319
000526  000415                                  BR      19$                     ;                                       5317
000530  013700  000156'                  17$:    MOV     ICTLR,R0                ;                                       5380
000534  006300                                  ASL     R0
000536  062700  000000G                         ADD     #C.ERR.TBL,R0
000542  032766  000001  000006                  BIT     #1,6(SP)                ; *,SOFT.OCCURED                        5378
000550  001403                                  BEQ     18$
000552  105260  000001                          INCB    1(R0)                   ;                                       5380
000556  000401                                  BR      19$                     ;                                       5378
000560  105210                          18$:    INCB    (R0)                    ;                                       5382
000562  062706  000010                  19$:    ADD     #10,SP                  ;                                       5276
000566  000207                                  RTS     PC

; Routine Size:  188 words,    Routine Base:  $CODE$ + 24666
; Maximum stack depth per invocation:  12 words


;        5385
;        5386  end
;        5387
;        5388  eludom


;                                       OTS external references
                                                .GLOBL  $SAVE5, $SAVE4, $SAVE3, $SAVE2
                                                .GLOBL  BL$SHF, BL$DIV, BL$MOD, BL$MUL


;                                       PSECT SUMMARY
;
;       Psect Name                      Words     Attributes
;        $GGG$                            343       RO ,  I  ,  LCL,  REL,  CON
;        $CODE$                          5527       RO ,  I  ,  LCL,  REL,  CON
;        $PLIT$                            10       RO ,  D  ,  LCL,  REL,  CON
```

```
;                               LIBRARY STATISTICS
;
;                                        -------- Symbols --------    Blocks
;       File                             Total    Loaded   Percent      Read
;
;  DISK$USER2:[DIETZ.RELEASE]ZRQAB0.L16;27
;                                           455       361        79       125
;
;
;
;
;                                   COMMAND QUALIFIERS
```

```
;         5389  module ZRQAM4 (
;         5390
;         5391  %title 'RD/RX EXERCISER'
;         5392                  ident = 'V01.2',
;         5393                  addressing_mode (absolute),
;         5394                  environment (noeis)
;         5395                  ) =
;         5396
;         5397  begin
;         5398
;         5399  %sbttl 'LASTAD AND SETUP'
;         5400
;         5401  library 'ZRQAB0.L16';
;         5402
;         5403  require 'BLSMAC.REQ';                          ! DIAGNOSTIC SUPERVISOR LIBRARY
;         6894
;         6895  LASTAD
;         6896
;         6897  BGNSETUP (3)
;         6898
;       P 6899          BGNPTAB
;       P 6900                  INIT_IP_ADDR, INIT_INTR_VECT, INIT_BR_LEVEL, %o'100034', 0, RD_MAX_LBN
;       P 6901                          ! IP, VECTOR, BR, DISK ADDR, START BLOCK, END BLOCK
;         6902          ENDPTAB
;         6903
;       P 6904          BGNPTAB
;       P 6905                  INIT_IP_ADDR, INIT_INTR_VECT, INIT_BR_LEVEL, %o'100001', 0, RX_MAX_LBN
;       P 6906                          ! IP, VECTOR, BR, DISK ADDR, START BLOCK, END BLOCK
;         6907          ENDPTAB
;         6908
;       P 6909          BGNPTAB
;       P 6910                  INIT_IP_ADDR, INIT_INTR_VECT, INIT_BR_LEVEL, %o'100002', 0, RX_MAX_LBN
;       P 6911                          ! IP, VECTOR, BR, DISK ADDR, START BLOCK, END BLOCK
;         6912          ENDPTAB
;         6913
;         6914  ENDSETUP


                                        .TITLE  ZRQAM4 RD/RX EXERCISER
                                        .IDENT  /V01.2/
                                        .ENABL  AMA


000000                                  .PSECT  $XYZ$,  RO
000000  000064'                BL$LAS::.WORD   T$FREE
000002  000000C                        .WORD   <<T$FREE-<BL$LAS+4>>/2>
000004  000030'                P.AAA:  .WORD   L$LAST+24
000006  000006                        .WORD   6                               ; Plit count word
000010  172150                P.AAB:  .WORD   -5630
000012  000154                        .WORD   154
000014  000004                        .WORD   4
000016  100034                        .WORD   -77744
000020  000000                        .WORD   0
```

```
000022  052137                          .WORD   52137
000024  000050'                P.AAC:  .WORD   L$LAST+44
000026  000006                          .WORD   6                       ; Plit count word
000030  172150                P.AAD:  .WORD   -5630
000032  000154                          .WORD   154
000034  000004                          .WORD   4
000036  100001                          .WORD   -77777
000040  000000                          .WORD   0
000042  001437                          .WORD   1437
000044  000000                P.AAE:  .WORD   0
000046  000006                          .WORD   6                       ; Plit count word
000050  172150                P.AAF:  .WORD   -5630
000052  000154                          .WORD   154
000054  000004                          .WORD   4
000056  100002                          .WORD   -77776
000060  000000                          .WORD   0
000062  001437                          .WORD   1437
000064  000000                T$FREE::.WORD   0


        000004'                 L$LAST==        BL$LAS+4
        000003                  T$PTHV==        3
        000004'                 $LAS4=          P.AAA
        000010'                 $REM4=          P.AAB
        000024'                 $LAS3=          P.AAC
        000030'                 $REM3=          P.AAD
        000044'                 $$LAS1=         P.AAE
        000050'                 $REM2=          P.AAF


                                        .SBTTL  $END.LINK LASTAD AND SETUP
000000  000207                  $END.LINK::
                                        RTS     PC                      ;                                       6893

; Routine Size:  1 word,        Routine Base:  $XYZ$ + 0066
; Maximum stack depth per invocation:  0 words


;         6915  end
;         6916
;         6917  eludom


;                               PSECT SUMMARY
;
;       Psect Name              Words     Attributes
;        $XYZ$                    28       RO ,  I  ,  LCL,  REL,  CON


;                           LIBRARY STATISTICS
```

```
;
;                                        -------- Symbols --------     Blocks
;       File                             Total    Loaded   Percent      Read
;
;  DISK$USER2:[DIETZ.RELEASE]ZRQAB0.L16;27
;                                          455         7         1        17
;
;
;
;
;
;                                    COMMAND QUALIFIERS
;
;
;         6918
; Size:           5528 code + 380 data words
; Run Time:         03:10.5
; Elapsed Time:     05:38.6
; Memory Used:  413 pages
; Compilation Complete
```