11/Z1+ RQDX1 EXERCISER
RD51 RX50 CNRQAA0

AH-T760A-MC
FICHE 1 OF 3

APR 1984
digital
Made In USA

11/Z1+ RQDX1 EXERCISER
RD51 RX50 CNRQAA0

AH-T760A-MC
FICHE 2 OF 3

APR 1984
digital
Made In USA

11/Z1+ RQDX1 EXERCISER
RD51 RX50 CNRQAA0

AH-T760A-MC
FICHE 3 OF 3

APR 1984
digital
Made In USA

```
;          0001  module NRQAM1 (
;          0002
;          0003  %title  'RD/RX EXERCISER
;          0004                  ident = 'V01.0',
;          0005                  addressing mode (absolute),
;          0006                  environment (noe's)
;          0007                  )
;          0008
;          0009  begin
;          0010
;        C 0011  %(
;        C 0012                                  IDENTIFICATION
;        C 0013
;        C 0014
;        C 0015          PRODUCT CODE:           AC-T759A-MC
;        C 0016
;        C 0017          PRODUCT NAME:           CNRQAA0 RQDX1 EXERCISER
;        C 0018
;        C 0019          PRODUCT DATE:           DECEMBER 19, 1983
;        C 0020
;        C 0021          MAINTAINER:             ISS DIAGNOSTIC SERVICES
;        C 0022
;        C 0023          AUTHOR:                 JAMES S. DOUCETTE
;        C 0024
;        C 0026
;        C 0027
;        C 0028          COPYRIGHT (C) 1983
;        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          THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:
;        C 0047
;        C 0048          DIGITAL         PDP             UNIBUS          MASSBUS
;        C 0049          DEC             DECUS           DECTAPE
```

```
;       C 0050                          REVISION HISTORY
;       C 0051                          ---------------
;       C 0052
;       C 0053          The following changes were made to CZRQAB in producing CNRQAA for the
;       C 0054          FALCON PLUS project (SBC 11/21+).  Release date December 19, 1983.
;       C 0055          All changes are marked by "!JSD REV A".
;       C 0056
;       C 0057          1. Lowered the general operating priority of the program from level 7
;       C 0058                  to level 6 to allow the BREAK key to interrupt and invoke
;       C 0059                  ODT.
;       C 0060
;       C 0061          2. Set the ODT BREAK vector (location 140) to the starting address
;       C 0062                  of FALCON's ODT ROM (170000-octal).
;       C 0063
;       C 0064          3. Due to space limitations, removed all references to DBM's (debug
;       C 0065                  messages).
;       C 0066
;       C 0067          4. Changed the default IP address from 172150 to 176150.
;       C 0068
;       C 0069      -- -- ------- ------------------ ----------- -           -           --
;       C 0070
;       C 0071                          TABLE OF CONTENTS
;       C 0072
;       C 0073            1.0     GENERAL INFORMATION
;       C 0074            1.1     PROGRAM ABSTRACT
;       C 0075            1.2     SYSTEM REQUIREMENTS
;       C 0076            1.2.1   HARDWARE REQUIREMENTS
;       C 0077            1.2.2   SOFTWARE REQUIREMENTS
;       C 0078            1.3     RELATED DOCUMENTS AND STANDARDS
;       C 0079            1.4     DIAGNOSTIC HIERARCHY PREREQUISITES
;       C 0080            1.5     ASSUMPTIONS
;       C 0081            1.6     MEMORY MAP
;       C 0082
;       C 0083            2.0     OPERATING INSTRUCTIONS
;       C 0084            2.1     HARDWARE QUESTIONS
;       C 0085            2.2     SOFTWARE QUESTIONS
;       C 0086
;       C 0087            3.0     ERROR TYPES
;       C 0088            3.1     ERROR INFORMATION
;       C 0089            3.2     INITIALIZATION ERRORS
;       C 0090            3.3     EXERCISER ERRORS
;       C 0091            3.4     ERROR LOG MESSAGES
;       C 0092            3.5     MSCP ERRORS
;       C 0093            3.6     SAMPLE MSCP ERROR STATEMENT
;       C 0094            3.7     DUP ERRORS
;       C 0095            3.8     SAMPLE DUP ERROR STATEMENT
;       C 0096            4.0     PERFORMANCE AND PROGRESS REPORTS
;       C 0097
;       C 0098            5.0     TEST SUMMARY
;       C 0099            5.1     INITIALIZATION SUBTEST
;       C 0100            5.2     EXERCISER
;       C 0101            5.3     DROP UNIT SUMMARY
;       C 0102
;       C 0103            6.0     ERROR LIST
;       C 0104
;       C 0105            7.0     DATA PATTERNS
```

```
;       C 0106
;       C 0107          1.0   GENERAL INFORMATION
;       C 0108
;       C 0109
;       C 0110          1.1   PROGRAM ABSTRACT
;       C 0111                - ------- ----
;       C 0112
;       C 0113                This program will functionally verify and exercise RQDX1
;       C 0114                Controller/Disk Drive subsystems. It is designed to verify
;       C 0115                that the subsystem is functioning correctly and operating
;       C 0116                within design specifications.
;       C 0117
;       C 0118          1.2   SYSTEM REQUIREMENTS
;       C 0119                -------  --
;       C 0120
;       C 0121          1.2.1 HARDWARE REQUIREMENTS
;       C 0122                ---------------------
;       C 0123
;       C 0124                SBC-11/21+ processor with 28KW memory (jumpered for memory
;       C 0125                map 0), CNDP+ (XXDP+) load device (e.g., RX02), console
;       C 0126                device (eg. VT100), RQDX1 CONTROLLER board, and attached
;       C 0127                RD-51 WINCHESTER drive(s) and/or RX-50 FLOPPY drive(s).
;       C 0128
;       C 0129          1.2.2 SOFTWARE REQUIREMENTS
;       C 0130                --------------------
;       C 0131
;       C 0132                This diagnostic is designed to run with the Diagnostic
;       C 0133                Supervisor as described in paragraph 2.0.
;       C 0134
;       C 0135          1.3   RELATED DOCUMENTS AND STANDARDS
;       C 0136                -------------------------------
;       C 0137
;       C 0138                XXDP+ SUPERVISOR/USERS MANUAL   CHQUS
;       C 0139                UQSSP UNIBUS/Q-BUS STORAGE SYSTEMS PORT
;       C 0140                MSCP  MASS STORAGE SYSTEM PROTOCOL
;       C 0141                DUP   DIAGNOSTIC/UTILITIES PROTOCOL
;       C 0142
;       C 0143          1.4   DIAGNOSTIC HIERARCY PREREQUISITES
;       C 0144                ---------------------------------
;       C 0145
;       C 0146                NONE
;       C 0147
;       C 0148          1.5   ASSUMPTIONS
;       C 0149                -----------
;       C 0150
;       C 0151                The hardware, other than the subsystem being tested is
;       C 0152                assumed to work properly. False errors may be reported if
;       C 0153                the processor, memory, etc., do not function properly.
```

```
;       C 0154
;       C 0155          1.6     MEMORY MAP
;       C 0156
;       C 0157
;       C 0158                  Memory layout on 28k machine   CNDP+ environment
;       C 0159                  ------ - - ------ - ---- --
;       C 0160
;       C 0161                                                      ADDRESS
;       C 0162
;       C 0163                  +-------------------------+         160000
;       C 0164                  |                         |
;       C 0165                  |     CNDP+ MONITOR       |
;       C 0166                  |                         |
;       C 0167                  +- -  -      --- - --- -+
;       C 0168                  |                         |
;       C 0169                  | CNDP+   TTY I/O AREA    |
;       C 0170                  |                         |
;       C 0171                  +  -     -  -- ---- --- +
;       C 0172                  |                         |
;       C 0173                  |                         |
;       C 0174                  |         DRS             |
;       C 0175                  |                         |
;       C 0176                  |                         |
;       C 0177                  +---------  -- ----------+         121000
;       C 0178                  |                         |
;       C 0179                  |     EXERCISER           |
;       C 0180                  |                         |
;       C 0181                  +-------------------------+         2000
;       C 0182                  |                         |
;       C 0183                  |       STACKS            |
;       C 0184                  |                         |
;       C 0185                  +-------------------------+         400
;       C 0186                  |                         |
;       C 0187                  |     VECTOR AREA         |
;       C 0188                  |                         |
;       C 0189                  +-------------------------+         0
;       C 0190
;       C 0191          In a machine  with more memory,  free space will ocuur
;       C 0192          between the exerciser and the DRS.
```

```
;       C 0193
;       C 0194          2.0   OPERATING INSTRUCTIONS
;       C 0195
;       C 0196
;       C 0197          This is a Rev C  Supervisor  Diagnostic;  for operating
;       C 0198          instructions,  please see chapter 5 of XXDP+ operator's
;       C 0199          manual.  They are no longer  included in the diagnostic
;       C 0200          because  it is desired  that a change in those instruc-
;       C 0201          tions not require a re-assembly of all Supervisor Diag
;       C 0202          nostics.
;       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 ident'fy each disk subsystem.
;       C 0209
;       C 0210          Hardware Configuration Quest'ons
;       C 0211
;       C 0212
;       C 0213      The  program  will  ask  the  following  questions  in
;       C 0214      response to a START command (non-script).
;       C 0215
;       C 0216      1.  CHANGE HW (L) Y ?
;       C 0217
;       C 0218          Answer NO to use the pre-built answers for all  hardware
;       C 0219          questions.   This  program will be released pre-built to
;       C 0220          test three unit with default answers  shown below.   The
;       C 0221          pre-built  answers  may  be changed at any time with the
;       C 0222          setup utility.  Answer YES if you want all the  hardware
;       C 0223          questions to be asked.
;       C 0224
;       C 0225      2.  NUMBER OF UNITS (D) ?
;       C 0226
;       C 0227          No default.  Answer with the number of  disk drive units
;       C 0228          to be exercised or tested.   This  answer will determine
;       C 0229          how  many  times  the  following questions are asked.  A
;       C 0230          range  of 1 to 4 units  may be specified.  A unit number
;       C 0231          will be assigned  sequentially from 0  by the Diagnostic
;       C 0232          supervisor for each unit.
;       C 0233
;       C 0234      3.  IP ADDRESS (O) 176150 ?
;       C 0235
;       C 0236          Enter the address of the IP register of one RQDX1 as ad
;       C 0237          dressed  by  the processor with memory management turned
;       C 0238          off.  The program expects an even 16-bit address in  the
;       C 0239          range of 160000 to 177774.  176150 is the default.
;       C 0240
;       C 0241      4.  VECTOR ADDRESS (O) 154 ?
;       C 0242
;       C 0243          Answer  with the  interrupt vector of same  RQDX1 in  the
;       C 0244          above  question.   A vector  address in the range of 4 to
;       C 0245          774 may be specified.  154 is the default.
```

```
;       C 0246
;       C 0247          5.  BR LEVEL (D) 4 ?
;       C 0248
;       C 0249              Answer with the bus request interrupt level used by  the
;       C 0250              above RQDX1. Levels 4 through 7 are acceptable. 4 is the
;       C 0251              default.
;       C 0252
;       C 0253          6.  RQDX1 DRIVE NUMBER (D) 0 ?
;       C 0254
;       C 0255              Enter the  logical unit  number for one drive associated
;       C 0256              with the  IP address  above.  Drive  numbers are  in the
;       C 0257              range  of 0  through 3.  The  number  entered  here must
;       C 0258              match the unit plug on the front panel of the device.  0
;       C 0259              is the default answer.
;       C 0260
;       C 0261          7.  ENTER UNIT TYPE:  RD51 - YES  OR  RX50 - NO ?
;       C 0262
;       C 0263              Question  will  be  asked to  determine the type of disk
;       C 0264              that this particular unit is; either RD51, or RX50.  The
;       C 0265              software will  not check for the correctness of the con-
;       C 0266              figuration  as indicated by the answers to the UNIT TYPE
;       C 0267              question.
;       C 0268
;       C 0269          8.  ALSO EXERCISE DIAGNOSTIC AREA (-NON-CUSTOMER AREA) ?
;       C 0270
;       C 0271              A "Yes" answer to this question will turn on the DUP
;       C 0272              (Diagnostic/Utilities Protocol) Exerciser.  The DUP Ex-
;       C 0273              erciser will read the DBNs (Diagnostic Blocks) in a
;       C 0274              sequential order from 0 to octal (217). DBNs can only
;       C 0275              be accessed throw DUP protocol making the blocks unreach-
;       C 0276              able through normal customer use.  The DUP Exerciser is
;       C 0277              contained in the middle of the MSCP (Mass Storage Control
;       C 0278              Protocol) Exerciser which reads LBNs (Logical blocks).
;       C 0279              So physically a large amount of LBNs will be READ and/or
;       C 0280              WRITTEN then the DUP Exerciser will turn on and READ
;       C 0281              and/or WRITE a few DBNs. The DUP Exerciser will then
;       C 0282              reinitialize the controller so the MSCP Exerciser can
;       C 0283              continue where it left off. The processes of reinitial-
;       C 0284              izing takes a few seconds.
;       C 0285
;       C 0286          9.  WRITE ON DIAGNOSTIC AREA?
;       C 0287
;       C 0288              A "yes" answer to this statement will allow the DUP Ex-
;       C 0289              erciser to WRITE/READ and compare the DBNs (Diagnostic
;       C 0290              Blocks).  Where a "NO" answer will allow the program to
;       C 0291              READ only the DBNs.
;       C 0292
;       C 0293              NOTE  **  IF yes ANSWER, DATA ON DBNs WILL BE DESTROYED.
;       C 0294
;       C 0295          10. STARTING LBN (D) 0 ?
;       C 0296
;       C 0297              Enter the starting logical block number of the  customer
;       C 0298              data area that you are going to test. LBNs  range from 0
;       C 0299              to  21599  (RD51),  or  0 to 799  (RX50),  with 0 as the
;       C 0300              default.
```

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

```
;       C 0329
;       C 0330          2.2     SOFTWARE QUESTIONS
;       C 0331
;       C 0332
;       C 0333                  Software Parameter Questions
;       C 0334
;       C 0335          The program will ask the following questions in response
;       C 0336          to the START, RESTART, and CONTINUE commands.
;       C 0337
;       C 0338            1.  CHANGE SW (L) Y ?
;       C 0339
;       C 0340              Answer NO to bypass the following questions in this sec
;       C 0341              tion.  This question should normally be answered NO when
;       C 0342              the Exerciser is first run.  A NO answer will cause  the
;       C 0343              Exerciser  to  select  the default parameters shown with
;       C 0344              each question below.  Then, depending on the errors  de-
;       C 0345              tected, it may be desirable to change this answer to YES
;       C 0346              to alter the test parameters  and  further  isolate  the
;       C 0347              problem.
;       C 0348
;       C 0349            2.  HARD ERROR LIMIT (D) 32 ?
;       C 0350
;       C 0351              Enter the number of hard errors allowed before a unit is
;       C 0352              dropped  from  testing.   A  number in the range of 1 tc
;       C 0353              65535 will be accepted.
;       C 0354
;       C 0355            3.  TRANSFER LIMIT IN MEGABYTES (0 FOR QUICK PASS) (D) 0 ?
;       C 0356
;       C 0357              When the specified number of bytes have been transferred
;       C 0358              to/from  a  unit, the unit will be dropped from testing.
;       C 0359              When all units are dropped, an end-of-pass will be indi-
;       C 0360              cated.   This  is  the method used to determine how long
;       C 0361              the Exerciser is to run.
;       C 0362
;       C 0363              The  only other  way the  Exerciser will declare end-of
;       C 0364              pass is if all units are dropped because the error limit
;       C 0365              on each is  exceeded.  However,  the operator can always
;       C 0366              abort the program at  ny time by typing CONTROL-C.
;       C 0367
;       C 0368            4.  PERCENTAGE OF RD51 OPERATIONS OUT OF TOTAL OPERATONS (D) 99 ?
;       C 0369
;       C 0370              In  order to maintain  typical usage for the devices of
;       C 0371              this  exercise,  a certain percentage of operations must
;       C 0372              be directed to the RD51s (the rest go to the RX50s).  It
;       C 0373              turns  out  that this  percentage is very high (as indi-
;       C 0374              cated by the 99% figure given as the default). It may be
;       C 0375              desirable in  some cases to  direct more activity to the
;       C 0376              RX50s.  This is easily  done by directing a smaller per-
;       C 0377              centage  of the  operations to  the  RD51s.  The numbers
;       C 0378              associated  with  usage  are  adjusted internally by the
;       C 0379              program according to device type and percentage.
```

```
;       C 0380
;       C 0381          5.  NUMBER OF DBNs READ AT ONE TIME (18) ?
;       C 0382
;       C 0383              This variable adjusts the amount of DBNs read in one pass
;       C 0384              of the DUP Exerciser.  The DUP Exerciser has to reinit
;       C 0385              the controller every time it goes back into the MSCP Exer
;       C 0386              ciser.  This Hard reinitialization takes a few seconds.
;       C 0387              The ratio of LBN tranfers to DBN tranfers DOESN'T change
;       C 0388              only the amount of DBNs read on one pass.  So the higher
;       C 0389              this number the less reinitializing the program does and
;       C 0390              the less time to run. The lower this number the more
;       C 0391              rein'tializations the longer amount of time to tranfer the
;       C 0392              same amount LBNs and DBNs.
;       C 0393
;       C 0394          6.  CLEAR STATISTICAL TABLES AFTER PRINTING (L) N ?
;       C 0395
;       C 0396              Answering YES causes the statistical fields to be clear
;       C 0397              ed to zero after the report is printed (either at end of
;       C 0398              pass,  or at  operator  request).  Otherwise, cumulative
;       C 0399              totals are maintained.
;       C 0400
;       C 0401          7.  RANDOM SEEK MODE (L) Y ?
;       C 0402
;       C 0403              Answer YES to cause block numbers to be chosen randomly.
;       C 0404              Answer  NO to cause block numbers to be selected sequen-
;       C 0405              tially.
;       C 0406
;       C 0407          8.  UNITS TO BE SELCTED AT RANDOM (NO, IMPLIES SEQUENTIAL) (L) N ?
;       C 0408
;       C 0409              This  question is  optionally asked if the answer to the
;       C 0410              previous  question is N[o].  The  selection of units for
;       C 0411              sequential  operations is affected by the answer to this
;       C 0412              question.  If the default  answer is chosen (N[o]), then
;       C 0413              units  shall be  selected in a  predetermined manner  in
;       C 0414              accordance  with the typical  seek time margins for each
;       C 0415              device. If the alternate answer is chosen (Y[es]),  then
;       C 0416              the  units will be chosen  at random  in accordance with
;       C 0417              the percentages specified in Software question 4.
;       C 0418
;       C 0419          9.  READ-COMPARES PERFORMED AT THE CONTROLLER (L) N ?
;       C 0420
;       C 0421              Answering YES causes all read commands  to  include  the
;       C 0422              "compare"  modifier.   This  essentially forces the con
;       C 0423              troller to perform two read operations on the same  disk
;       C 0424              address, and to compare the results.
;       C 0425
;       C 0426              The following message will appear after the operator has
;       C 0427              answered this question:
;       C 0428
;       C 0429              THE REMAINING QUESTIONS ONLY APPLY TO UNPROTECTED DISK UNITS.
```

```
;       C 0430
;       C 0431          10.  WRITE COMPARES PERFORMED AT THE CONTROLLER (L) N ?
;       C 0432
;       C 0433              Answering YES causes all write I/O requests to be
;       C 0434              changed to write-compare.  After each write, the con
;       C 0435              troller will read the data and compare it to data
;       C 0436              re obtained from the host.
;       C 0437
;       C 0438          11.  CHECK ALL WRITES AT HOST BY READING (L) N ?
;       C 0439
;       C 0440              This question will only be asked if the previous ques
;       C 0441              tion was answered NO.  Answering YES causes all writes
;       C 0442              to be checked by the host by reading the data immediate
;       C 0443              ly after the write operation.  This option consumes
;       C 0444              extra CPU time, and doubles the amount of storage re
;       C 0445              quired for writes.  Therefore, it is only recommended
;       C 0446              when device write-compare operations are suspect.
;       C 0447
;       C 0448          12. USER-DEFINED DATA PATTERN (L) N ?
;       C 0449
;       C 0450              An answer of YES allows the operator to define his/her
;       C 0451              own data pattern to be used in all write operations.  A
;       C 0452              NO answer will allow the operator to select a
;       C 0453              pre-defined data pattern in the next question.
;       C 0454
;       C 0455          13. SELECT PRE-DEFINED DATA PATTERN (0 FOR SEQUENTIAL SELEC-
;       C 0456              TION) (D) 0 ?
;       C 0457
;       C 0458              There are 21 predefined data patterns available, select-
;       C 0459              ed as 1 to 21 (see section 4.9).  A zero answer will
;       C 0460              cause patterns 1 to 21 to be sequentially selected for
;       C 0461              each write.  (Note that pattern 1 consists entirely of
;       C 0462              random numbers).
;       C 0463
;       C 0464          14. NUMBER OF WORDS IN DATA PATTERN (16 MAXIMUM) (D) 16 ?
;       C 0465              PATTERN VALUES (0) ?
;       C 0466
;       C 0467              These questions will only be asked if the operator has
;       C 0468              decided to define his/her own data pattern.  The actual
;       C 0469              bit patterns will be entered as octal.
```

```
;       C 0470
;       C 0471          3.0   ERROR TYPES
;       C 0472                --  -----  --
;       C 0473
;       C 0474                This program has four types of error  classifications;
;       C 0475                system fatal, dev'ce fatal, hard and soft.
;       C 0476
;       C 0477                SYSTEM FATAL ERRORS
;       C 0478                ---    -------- ----
;       C 0479
;       C 0480                System fatal errors are used to 'ndicate that an error
;       C 0481                was detected by the  Diagnostic Supervisor 'n relation
;       C 0482                to loading/controlling the diagnostic process.
;       C 0483
;       C 0484                The content  of each  error is  such that it should be
;       C 0485                self explanatory.  However,  the messages utilize some
;       C 0486                terms that are specific to the disk subsystem, and may
;       C 0487                require some getting use to.
;       C 0488
;       C 0489                DEVICE FATAL ERRORS
;       C 0490                -------- -------  -
;       C 0491
;       C 0492                Device fatal errors are a result of:
;       C 0493
;       C 0494                an error that is  considered fatal to the device,  but
;       C 0495                testing will continue.
;       C 0496
;       C 0497                HARD ERRORS
;       C 0498                -- -- -----
;       C 0499
;       C 0500                Hard errors are a result of:
;       C 0501
;       C 0502                        1.  retries of a soft error or *
;       C 0503                        2.  a non-recoverable error
;       C 0504                        3.  a soft error if retrys are not set.
;       C 0505
;       C 0506                        *  Note: Retries are executed in the controller
;       C 0507
;       C 0508                SOFT ERRORS
;       C 0509                ------ ----
;       C 0510
;       C 0511                Soft errors  are media related errors.  All soft errors
;       C 0512                will be retried by the controller.
;       C 0513
;       C 0514                Note: Soft errors are retrieved from the controller via
;       C 0515                the error log  capabilities of  MSCP.
```

```
;       C 0516
;       C 0517          3.1     ERROR INFORMATION
;       C 0518                  -- - ------ -----------
;       C 0519
;       C 0520          All general error messages will include the type of error
;       C 0521          (system-fatal, device-fatal, hard, soft) and a unit number.
;       C 0522          If the error applies to a controller, then only the first
;       C 0523          unit number of the controller will be g'ven. (The user w'll
;       C 0524          know the other unit numbers when subsequent "drop un't'
;       C 0525          messages are printed).
;       C 0526
;       C 0527              Basic error messages provide more details about the
;       C 0528          error. The Exerciser will print all basic error messages,
;       C 0529          along with the disk address, if applicable. In some
;       C 0530          cases where a device-fatal error applies to a controller,
;       C 0531          the controller's IP address will be printed.
;       C 0532
;       C 0533              Extended error messages will be used to print the
;       C 0534          relevant fields of command and end message packets, status
;       C 0535          codes, SA register contents, and error log messages. All
;       C 0536          values will be in octal.
;       C 0537
;       C 0538              The error messages in this section do not include
;       C 0539          errors detected and printed by the Diagnostic Supervisor.
;       C 0540
;       C 0541          3.2  INITIALIZATION ERRORS
;       C 0542               -- - ----  ---- ----
;       C 0543
;       C 0544              Two kinds of errors will be reported to the operator
;       C 0545          during the Initialization Test. The System-fatal error is -
;       C 0546          too many units specified. A system-fatal error will cause
;       C 0547          the Exerciser to abort.
;       C 0548
;       C 0549              Device-fatal errors only affect the unit(s) involved.
;       C 0550          Testing will continue on all other units. This class of
;       C 0551          errors includes, but is not limited to, the following:
;       C 0552
;       C 0553              1.  Register Existence Test failure (no device present)
;       C 0554
;       C 0555              2.  Vector Test failure
;       C 0556
;       C 0557              3.  BR Level Test failure
;       C 0558
;       C 0559              4.  Initialization sequence failure
;       C 0560
;       C 0561              5.  Online failed
;       C 0562
;       C 0563              6.  Access fa'led
```

```
;       C 0564
;       C 0565          3.3   EXERCISER ERRORS
;       C 0566                --
;       C 0567
;       C 0568              Most errors reported during this test will originate
;       C 0569          from MSCP end message packets.  The status code field will
;       C 0570          be converted to text and printed as part of a basic error
;       C 0571          message.  Any subcode value will follow if extended error
;       C 0572          messages are enabled.
;       C 0573
;       C 0574          The following list represents some of the error conditions
;       C 0575          reported via MSCP:
;       C 0576
;       C 0577              1.  Disk unit went offline (a sub-code may follow
;       C 0578                  detailing the reason)
;       C 0579
;       C 0580              2.  Compare error
;       C 0581
;       C 0582              3.  Data error (a sub-code may follow)
;       C 0583
;       C 0584              4.  Drive error (a sub-code may follow)
;       C 0585
;       C 0586              5.  Host buffer access error
;       C 0587
;       C 0588              6.  Media format error (a sub-code may follow)
;       C 0589
;       C 0590          3.4   ERROR LOG MESSAGES
;       C 0591                - ------------   -
;       C 0592
;       C 0593              The contents of the error-log messages received from the
;       C 0594          controller are printed as received, and should be deciphered
;       C 0595          using the MSCP specs.
;       C 0596
;       C 0597          3.5   MSCP ERRORS
;       C 0598                ----------
;       C 0599
;       C 0600              An MSCP error occurs when the host receives an Invalid
;       C 0601          Command End Message from the RQDX1.  In such cases, the host
;       C 0602          will print out the erroneous command followed by the reason
;       C 0603          for the error.  If extended printouts are enabled, then the
;       C 0604          entire contents of the end message will be displayed in octal
;       C 0605          without interpretation of the data.
```

```
;       C 0606
;       C 0607          3.6   SAMPLE MSCP ERROR STATEMENT
;       C 0608
;       C 0609
;       C 0610                  The errors listed by the exerciser are usually very des
;       C 0611          criptive and  are self explanatory.  The  following is an example
;       C 0612          error  statement.  This  error  statement  is  the extented error
;       C 0613          message.
;       C 0614
;       C 0615          (example)                           (comments)
;       C 0616
;       C 0617          * DISK: XXX                         !DISK UNIT NUMBER
;       C 0618          CRN: XXXXX                          !CONTROLLER PACKET NUMBER
;       C 0619          MESSAGE TYPE: - SEQUENTIAL          !THIS IS THE PORT MESSAGE TYPE
;       C 0620          COMMAND:-MSCP-READ-COMPARE          .CONNECTION ID AND COMMAND AND MODIFIER GIVEN TO DRIVE
;       C 0621          STATUS CODE:  UNIT OFFLINE          ! STATUS CODE OF COMMAND
;       C 0622          STATUS SUB-CODE:  NO VOLUME MOUNTED OR DRIVE DISABLED BY SWITCH ! SUB CODE
;       C 0623          BYTE COUNT IN COMMAND XXXXX         !NUMBER OF BYTES WANTED TO READ
;       C 0624          ACTUAL # OF BYTES TRANSFERED XXXXX !NUMBER OF BYTES ACTUALLY READ
;       C 0625          I/O BUFFER ADDRESS (32 BITS)        XXXXXX XXXXXX
;       C 0626          LBN: XXX                            ! LBN WANTED TO READ
;       C 0627          END MESSAGE FLAGS:                  ! THERE ARE NO END FLAGS FOR THIS ERROR
;       C 0628
;       C 0629                  The  status  code in  an end  messages is  broken into two
;       C 0630          pieces. The first 5 bits represent the major status which is given
;       C 0631          by the "invalid command" message.  The 11 remaining bits represent
;       C 0632          the sub-code,  which tells in greater detail the error in the con-
;       C 0633          troller.  The LBN is the logical  block on the disk the controller
;       C 0634          was trying to read.  The byte count refers  to the number of bytes
;       C 0635          the controller was going to read off the LBN. The actual number of
;       C 0636          bytes  transfered refers  to the  number of bytes  read before the
;       C 0637          error.  The end message flags give any flags that might have been
;       C 0638          set by the controller. It is pretty apparent that this error was
;       C 0639          caused by something physically switching the disk offline.
;       C 0640
;       C 0641          3.7   DUP ERRORS
;       C 0642                ----------
;       C 0643
;       C 0644                  A DUP error occurs when the host receives an  Invalid
;       C 0645          Command End Message from the RQDX1.  In such cases, the host
;       C 0646          will print out the erroneous command followed by the  reason
;       C 0647          for the error.
;       C 0648                  There are two major places where errors in  DUP will
;       C 0649          occur. The first being in the status code much the same as MSCP,
;       C 0650          and the second in the DUP I/O Buffer. Using the DUP sub-protocol
;       C 0651          (using Receive/Send commands to communicate with controller local
;       C 0652          programs (p. 25 of DUP.V05)) the controller may send an error
;       C 0653          message to the host by writting in the DUP I/O Buffer.
```

```
;       C 0654
;       C 0655          3.8   SAMPLE DUP ERROR STATEMENT
;       C 0656
;       C 0657
;       C 0658                  The errors listed by the exerc'ser are ussually very des
;       C 0659          criptive and  are self explanatory.  The  follow'ng is an example
;       C 0660          error  statement.  This  error  statement  's  the extented error
;       C 0661          message.
;       C 0662
;       C 0663          (example)                       (comments)
;       C 0664
;       C 0665          * DISK: XXX                     !DISK UNIT NUMBER
;       C 0666          CRN: XXXXX                      !CONTROLLER PACKET NUMBER
;       C 0667          MESSAGE TYPE: - SEQUENTIAL      !THIS IS THE PORT MESSAGE TYPE
;       C 0668          COMMAND:-DUP-RECEIVE DATA       !CONNECTION ID AND COMMAND GIVEN TO DRIVE
;       C 0669          STATUS CODE: -SUCCESS           ! STATUS CODE OF COMMAND
;       C 0670          ACTUAL # OF BYTES TRANSFERED XXXXX !NUMBER OF BYTES ACTUALLY READ
;       C 0671          I/O BUFFER ADDRESS (32 BITS)    XXXXXX XXXXXX
;       C 0672          MESSAGE TYPE:                   ! TYPE OF COMMUNICATION
;       C 0673                  ** FATAL ERROR
;       C 0674          MESSAGE NUMBER:                 ! MESSAGE TOBE GIVEN OR RESUL' EXPECTED
;       C 0675           -  SUCCESS/FAILURE CODE
;       C 0676          MESSAGE ERROR CODE:             ! ERROR LISTING
;       C 0677           - ILLEGAL UNIT NUMBER
;       C 0678          DBN: XXX
;       C 0679
;       C 0680                  The DUP Error messages are almost like the MSCP messages
;       C 0681          but they contain an extra three classifications.  These are the
;       C 0682          MESSAGE TYPE, MESSAGE NUMBER, MESSAGE ERROR CODE.  These are part
;       C 0683          of a DUP sub-protocal for communicating with a controller local
;       C 0684          program.  The I/O buffer address contains the address of the DUP
;       C 0685          I/O Buffer which contains the MESSAGE information.
;       C 0686                  The  status  code in  this end  message is succesful which
;       C 0687          means the controller was OK but it did not understand the host
;       C 0688          message or something on the host side caused the controller to error
;       C 0689          while running the controller local program.  This error was produced
;       C 0690          by an ILLEGAL UNIT NUMBER.  The disk went off line.
;       C 0691
```

```
;       C 0692
;       C 0693          4.0   PERFORMANCE AND PROGRESS REPORTS
;       C 0694                - - - - - - - - - - - - - - - - - -
;       C 0695
;       C 0696              A summary report is printed at the end of each pass of
;       C 0697          the Exerciser or upon demand by the operator. The fields may
;       C 0698          be cleared to zero after the report is printed depending on
;       C 0699          the operator's response to this option in the software ques
;       C 0700          tions. Any units added to the test cycle will also begin with
;       C 0701          cleared statistics.
;       C 0702              There are two basic listings one for LBNs and one for
;       C 0703          DBNs. The DBN listing will only contain RD51s for the simple
;       C 0704          reason that they contain DBNs and RX50s do not. The DBNs
;       C 0705          are READ and WRITTEN by blocks instead of by bytes. All units
;       C 0706          contain LBNs. The errors for each unit will be listed next to
;       C 0707          the LBNs.
;       C 0708              Errors are grouped into two basic categories: hard and
;       C 0709          soft. Each is sub divided into four more categories, depend
;       C 0710          ing on the most probable classification for that error.
;       C 0711
;       C 0712              The sub categories are:
;       C 0713
;       C 0714          1. disk related errors
;       C 0715
;       C 0716          2. seek (or format) related errors
;       C 0717
;       C 0718          3. controller or drive related errors
;       C 0719
;       C 0720              4. host (the CPU) related errors.
;       C 0721
;       C 0722                  All numeric values are in decimal radix.
;       C 0723
;       C 0724          UNT       # OF BYTS  # OF   BYTES    -- HRD ERS --      SFT ERS -
;       C 0725           #  TYPE READS READ WRITES WRITTEN DAT SEK DRV HST DAT SEK DRV HST
;       C 0726          --- ---- ---- ---- ------ ------ -- ------ -------   - ----- --- -
;       C 0727
;       C 0728           X  RD51 XXXX XXXX XXXXX XXXXXX X   X   X   X   X   X   X   X
;       C 0729           .  RX50  ..   ..    ...    ....  .   .   .   .   .   .   .   .
;       C 0730           .  RX50  ..   ..    ...    ....  .   .   .   .   .   .   .   .
;       C 0731
;       C 0732          UNIT   DISK          # OF    # BLKS       # OF     # BLKS
;       C 0733            #      #    TYPE  READS    READ       WRITES   WRITTEN
;       C 0734          ----   ----- ------- -----  ---- --     ------   -------
;       C 0735            X      X   DBNRD51 XXXXX   XXXXX      XXXXX    XXXXX
;       C 0736
```

```
;       C 0737
;       C 0738          5.0   TEST SUMMARY
;       C 0739                -- ------- -
;       C 0740
;       C 0741              The  RQDX1  functional  tester and exerciser consists of
;       C 0742          two  parts,  the  initialization subtest  and the performance
;       C 0743          exerciser.  The operator is not able to select which of these
;       C 0744          two parts he/she wishes to run; they both must be executed.
;       C 0745
;       C 0746           5.1  INITIALIZATION SUBTEST
;       C 0747                --------- ---   -- -
;       C 0748
;       C 0749              The  purpose of  this subtest  is to verify the hardware
;       C 0750          configuration as specified by the operator, and to bring each
;       C 0751          unit online.  The Initialization  Subtest will always precede
;       C 0752          the execution of any other test.
;       C 0753
;       C 0754              First,  the presence of each device register will be ve
;       C 0755          rified,  along with a check on  the BR level specified by the
;       C 0756          operator.  Then,  an  initialization  will be issued to  each
;       C 0757          controller configured for testing.  When  the initialization
;       C 0758          sequence has been completed, an attempt will be made to bring
;       C 0759          each unit online.  If this succeeds,  one  or two MSCP reads
;       C 0760          will be issued to the inner-most LBN of each selected disk to
;       C 0761          to ensure that each disk drive can seek and be read.
;       C 0762
;       C 0763              Any device-fatal or hard errors encountered  during this
;       C 0764          test  will  cause the  appropriate unit(s) to be dropped.  If
;       C 0765          basic error messages are enabled, then the program will print
;       C 0766          out the specific reason for dropping the unit(s). Henceforth,
;       C 0767          the  failed  unit(s) will  not be  tested unless the operator
;       C 0768          intervenes (adds unit(s) or restarts Exerciser).
;       C 0769
;       C 0770              Upon succesful completion of the Initialization Subtest,
;       C 0771          the program  will  begin executing the Exerciser.
;       C 0772
;       C 0773        5.2  EXERCISER
;       C 0774             ---------
;       C 0775
;       C 0776           The purpose of this subtest is to exercise the  disk  drives
;       C 0777          in a manner similar to the typical usage under standard oper
;       C 0778          ating systems.  Execution of this test should give an indica
;       C 0779          tion of the operating performance of the disk drive subunits.
;       C 0780          This test will utilitize random disk addresses,  random  word
;       C 0781          counts,  and data  patterns,  all subject  to the  limits and
;       C 0782          specifications made by the operator. All protected disks will
;       C 0783          be subject to read-only  operations,  while unprotected disks
;       C 0784          may be read or written, depending on the answers given to the
;       C 0785          software parameter questions.  End-of-pass  will be declared
;       C 0786          when the specified number  of bytes have been transferred for
;       C 0787          all the disks taken as a whole.
```

```
;       C 0788
;       C 0789              If a read/write error occurs during this test,  then the
;       C 0790          RQDX1 CONTROLLER  will initiate an  appropriate number of re
;       C 0791          tries.  If all retries fail, then a hard error will be repor-
;       C 0792          ted to the host,  an  error message  will  be diplayed on the
;       C 0793          console  terminal and  the error will be tallied for the sum
;       C 0794          mary report. The unit will be dropped if the hard error count
;       C 0795          has exceeded the specified limit.
;       C 0796
;       C 0797              The Exerciser is actually two exercisers combined together.
;       C 0798          The main MSCP exerciser writes and reads LBNs while the less
;       C 0799          used DUP exerciser writes and reads DBNs. The DUP exerciser is
;       C 0800          used once per 25 LBN transfers or alot less than the MSCP
;       C 0801          exerciser.  The two Exercisers use a some what different
;       C 0802          protocol to tranfer I/O.  It is possible to go from MSCP
;       C 0803          protocol  to DUP protocol without reinitializing the controller
;       C 0804          but impossible to go from DUP to MSCP protocol.  Therfore after
;       C 0805          the DUP Exercise pass the controller is reinitialize and control
;       C 0806          given back to the MSCP Exerciser.  The reinitialization process
;       C 0807          takes a few seconds.  For this reason a variable is placed in
;       C 0808          the DUP Exerciser to allow it to transfer more than 1 DBN per pass.
;       C 0809          The variable,"X", is multiplied by 25 to give the amount of MSCP
;       C 0810          transfers before the DUP Exerciser can tranfer "X" amount of DBNs.
;       C 0811          The higher the variable the less reinits the controller must do.
;       C 0812
;       C 0813      5.3   DROP UNIT SUMMARY
;       C 0814              ---------------
;       C 0815
;       C 0816              During the Initialization Subtest,  individual units w ll
;       C 0817          be  dropped from  the test  sequence if they are unable  to be
;       C 0818          brought online or the operator specified device does not match
;       C 0819          the hardware.
;       C 0820
;       C 0821              During the Exercise, the program will drop a unit for one
;       C 0822          of three reasons. The normal path is for each unit to complete
;       C 0823          the transfer of N megabytes, where N is specified by the oper-
;       C 0824          ator during SW questioning and be soft-dropped.  Otherwise,  a
;       C 0825          unit will be hard-dropped if the number of hard errors encoun-
;       C 0826          tered exceeds the operator-specified limit, or if a fatal error
;       C 0827          is detected. Units hard-dropped may later be added  to the test
;       C 0828          cycle.  However,  statistics for  the  hard-added  unit will be
;       C 0829          cleared to zero;  if a transfer  limit was  specified, in which
;       C 0830          case  the unit was soft-dropped,  the statistics may or may not
;       C 0831          be  cleared  depending  on  the  operators  answer  to Software
;       C 0832          question 12.
```

```
;       C 0833
;       C 0834          6.0   ERROR CODES GENERATED BY NRQA EXERCISER
;       C 0835
;       C 0836             SYSTEM FATAL ERRORS
;       C 0837                     - --
;       C 0838
;       C 0839             1   More than 4 units specified
;       C 0840
;       C 0841             DEVICE FATAL ERRORS
;       C 0842                 -  ----- -
;       C 0843
;       C 0844             10  Controller couldn t be addressed   Wrong IP address selected
;       C 0845                 at the address given.
;       C 0846
;       C 0847             11  Controller didn't interrupt at    Wrong vector address sel
;       C 0848                 the interrupt vector given.       ected.
;       C 0849
;       C 0850             12  Controller didn't interrupt at    Wrong BR level selected.
;       C 0851                 the BR lelvel given.
;       C 0852
;       C 0853             13  Init sequence failed.             Either one of the four
;       C 0854                                                   initialization steps did
;       C 0855                                                   not receive the correct
;       C 0856                                                   response from the Con
;       C 0857                                                   troller, or one of the
;       C 0858                                                   steps timed-out.
;       C 0859
;       C 0860             14  Fatal Controller error.           The error bit (bit 15) in
;       C 0861                                                   the SA register was set.
;       C 0862
;       C 0863             15  Failed to bring unit on-line.     On-line response had an
;       C 0864                                                   error code.  (see also
;       C 0865                                                   #s 22 and 23.)
;       C 0866
;       C 0867             16  Write protect conflict.          The unit was hardware
;       C 0868                                                   write protected and write
;       C 0869                                                   operations were requested
;       C 0870                                                   on the unit.
;       C 0871
;       C 0872             17  Access to inner track failed.    Innermost track's header
;       C 0873                                                   may be corrupted.
;       C 0874
;       C 0875             18  Unit went off-line.              ---
;       C 0876
;       C 0877             20  Failed to send 'Set Controller   Either the unit is off-
;       C 0878                 Characteristics" command.         line or the Diagnostic is
;       C 0879                                                   corrupted because of any
;       C 0880                                                   problems with its RAM.
```

```
;       C 0681
;       C 0882              21  Controller  returned wrong  'end  Problem with the  Control
;       C 0883                  code'  for  the   Set Controller  ler microcode or the port/
;       C 0884                  Characteristics  command.         DMA intterface.
;       C 0885
;       C 0886              22  Failed to send 'On l'ne  command  Either  the  unit  is off
;       C 0887                                                    l'ne or  the diagnostic is
;       C 0888                                                    corrupted because  of  any
;       C 0889                                                    problems with its RAM.
;       C 0890
;       C 0891              23  Controller  returned  wrong 'end  Problem with the  Control
;       C 0892                  code' for the 'On line' command.  ler's  microcode  or   the
;       C 0893                                                    port/DMA interface.
;       C 0894
.       C 0895              24 Device went to available state     Fault switch or door mechanism
;       C 0896
;       C 0897              HARD ERRORS
.       C 0898              -    -----
;       C 0899
;       C 0900  MSCP ERRORS
;       C 0901
;       C 0902              31  Controller received  an  invalid  The  diagnostic is corrup-
;       C 0903                  command.                          ted  because  of any prob-
;       C 0904                                                    lems  with  its  RAM,   or
;       C 0905                                                    there  is  a  problem with
.       C 0906                                                    the   Controller   microde
;       C 0907                                                    (RAM or ROM)  or  there is
;       C 0908                                                    problem with  the port/DMA
;       C 0909                                                    interface.
;       C 0910
;       C 0911              32  Command aborted by the  Control-  Command  timed-out  in  the
;       C 0912                  ler.                              Controller.
;       C 0913
;       C 0914              35  Media format error.               ---
;       C 0915
;       C 0916              36  Device write protected.           --
;       C 0917
;       C 0918              37  Controller  read  or  write com-  ---
;       C 0919                  pare error.
;       C 0920
.       C 0921              38  Data error.                       CRC  error  in  the   data
;       C 0922                                                    field of a disk block.
;       C 0923
;       C 0924              39  Host buffer access error         ---
;       C 0925
;       C 0926              40  Controller error.                Difficult  to   catagorize
;       C 0927                                                    without  looking  at   the
.       C 0928                                                    error   sub-code   or   any
;       C 0929                                                    associated  error-log mes
;       C 0930                                                    sage.
```

```
;       C 0931
;       C 0932              41  Dr've error.                      See #40.
;       C 0933
;       C 0934              42  Host write compare error.         Error detected  when  Host
;       C 0935                                                    CPU  compared   the   data
;       C 0936                                                    written and read back. May
;       C 0937                                                    be a problem with the Host
;       C 0938                                                    or Controller RAM.
;       C 0939
;       C 0940              43 Message from 'nternal diagnostics  See #40.
;       C 0941
;       C 0942              44  Duplicate  unit number  detected   -
;       C 0943                  by the Controller.
;       C 0944
;       C 0945              45  Unknown end code received.        Problem with the  Control
;       C 0946                                                    ler microcode or the port/
;       C 0947                                                    DMA interface.
;       C 0948
;       C 0949  DUP ERRORS
;       C 0950  Host found error
;       C 0951              46 DBN compare error.                 see # 42
;       C 0952
;       C 0953  Message errors
;       C 0954              47 No local media                     Controller local program on
;       C 0955                                                    RAM may be corrupt
;       C 0956
;       C 0957              48 Illegal Unit #                     Unit went offline
;       C 0958
;       C 0959              49 Illegal relative or physical BLK #      see # 31
;       C 0960
;       C 0961              50 Device Error                       Possible write protected
;       C 0962
;       C 0963              51 Zero lenght message                see # 31
;       C 0964
;       C 0965  Status errors
;       C 0966              52 Invalid Command                    see # 31
;       C 0967
;       C 0968              53 No region available                see # 31
;       C 0969
;       C 0970              54 No region suitable                 see # 31
;       C 0971
;       C 0972              55 Program not known                  see # 47
;       C 0973
;       C 0974              56 Load failure                       ---
;       C 0975
;       C 0976              57 Stand alone type program           see # 31
;       C 0977
;       C 0978              58 DUP unkown status code             see # 31
```

```
;       C 0979
;       C 0980              SOFT ERRORS
;       C 0981              - - ---- -
;       C 0982
;       C 0983              60  Controller error.                 See  error-log  packet for
;       C 0984                                                    details as the exact cause
;       C 0985                                                    may not be evident.
;       C 0986
;       C 0987              61  Host memory access error.         See #50.
;       C 0988
;       C 0989              62  Disk transfer error.              See #50.
;       C 0990
;       C 0991              63  'Standard    Disk    Inteconnect'  See #50.
;       C 0992                  error.
;       C 0993
;       C 0994              64  'Small Disk' error.               See #50.
```

```
;       C 0995
;       C 0996          7.0   DATA PATTERNS
;       C 0997                - --------
;       C 0998
;       C 0999                        HEX      OCTAL                BINARY
;       C 1000                        --       ----                 ------
;       C 1001
;       C 1002      Pattern 1             R A N D O M   N U M B E R S
;       C 1003
;       C 1004      Pattern 2         0000     000000          0 000 000 000 000 000
;       C 1005
;       C 1006      Pattern 3         FFFF     177777          1 111 111 111 111 111
;       C 1007
;       C 1008      Pattern 4         8888     105613          1 000 101 110 001 011
;       C 1009
;       C 1010      Pattern 5         3333     031463          0 011 001 100 110 011
;       C 1011
;       C 1012      Pattern 6         3091     030221          0 011 000 010 010 001
;       C 1013
;       C 1014      Pattern 7         0001     000001          0 000 000 000 000 001
;       C 1015                        0003     000003          0 000 000 000 000 011
;       C 1016                        0007     000007          0 000 000 000 000 111
;       C 1017                        000F     000017          0 000 000 000 001 111
;       C 1018                        001F     000037          0 000 000 000 011 111
;       C 1019                        003F     000077          0 000 000 000 111 111
;       C 1020                        007F     000177          0 000 000 001 111 111
;       C 1021                        00FF     000377          0 000 000 011 111 111
;       C 1022                        01FF     000777          0 000 000 111 111 111
;       C 1023                        03FF     001777          0 000 001 111 111 111
;       C 1024                        07FF     003777          0 000 011 111 111 111
;       C 1025                        0FFF     007777          0 000 111 111 111 111
;       C 1026                        1FFF     017777          0 001 111 111 111 111
;       C 1027                        3FFF     037777          0 011 111 111 111 111
;       C 1028                        7FFF     077777          0 111 111 111 111 111
;       C 1029                        FFFF     177777          1 111 111 111 111 111
;       C 1030
;       C 1031      Pattern 8         FFFE     177776          1 111 111 111 111 110
;       C 1032                        FFFC     177774          1 111 111 111 111 100
;       C 1033                        FFF8     177770          1 111 111 111 111 000
;       C 1034                        FFF0     177760          1 111 111 111 110 000
;       C 1035                        FFE0     177740          1 111 111 111 100 000
;       C 1036                        FFC0     177700          1 111 111 111 000 000
;       C 1037                        FF80     177600          1 111 111 110 000 000
;       C 1038                        FF00     177400          1 111 111 100 000 000
;       C 1039                        FE00     177000          1 111 111 000 000 000
;       C 1040                        FC00     176000          1 111 110 000 000 000
;       C 1041                        F800     174000          1 111 100 000 000 000
;       C 1042                        F000     170000          1 111 000 000 000 000
;       C 1043                        E000     160000          1 110 000 000 000 000
;       C 1044                        C000     140000          1 100 000 000 000 000
;       C 1045                        8000     100000          1 000 000 000 000 000
;       C 1046                        0000     000000          0 000 000 000 000 000
```

```
;       C 1048          Pattern 9       0000    000000          0 000 000 000 000 000
;       C 1049                          0000    000000          0 000 000 000 000 000
;       C 1050                          0000    000000          0 000 000 000 000 000
;       C 1051                          FFFF    177777          1 111 111 111 111 111
;       C 1052                          FFFF    177777          1 111 111 111 111 111
;       C 1053                          FFFF    177777          1 111 111 111 111 111
;       C 1054                          0000    000000          0 000 000 000 000 000
;       C 1055                          0000    000000          0 000 000 000 000 000
;       C 1056                          FFFF    177777          1 111 111 111 111 111
;       C 1057                          FFFF    177777          1 111 111 111 111 111
;       C 1058                          0000    000000          0 000 000 000 000 000
;       C 1059                          FFFF    177777          1 111 111 111 111 111
;       C 1060                          0000    000000          0 000 000 000 000 000
;       C 1061                          FFFF    177777          1 111 111 111 111 111
;       C 1062                          0000    000000          0 000 000 000 000 000
;       C 1063                          FFFF    177777          1 111 111 111 111 111
;       C 1064
;       C 1065          Pattern 10      B6D9    133331          1 011 011 011 011 001
;       C 1066
;       C 1067          Pattern 11      5555    052525          0 101 010 101 010 101
;       C 1068                          5555    052525          0 101 010 101 010 101
;       C 1069                          5555    052525          0 101 010 101 010 101
;       C 1070                          AAAA    125252          1 010 101 010 101 010
;       C 1071                          AAAA    125252          1 010 101 010 101 010
;       C 1072                          AAAA    125252          1 010 101 010 101 010
;       C 1073                          5555    052525          0 101 010 101 010 101
;       C 1074                          5555    052525          0 101 010 101 010 101
;       C 1075                          AAAA    125252          1 010 101 010 101 010
;       C 1076                          AAAA    125252          1 010 101 010 101 010
;       C 1077                          5555    052525          0 101 010 101 010 101
;       C 1078                          AAAA    125252          1 010 101 010 101 010
;       C 1079                          5555    052525          0 101 010 101 010 101
;       C 1080                          AAAA    125252          1 010 101 010 101 010
;       C 1081                          5555    052525          0 101 010 101 010 101
;       C 1082                          AAAA    125252          1 010 101 010 101 010
;       C 1083
;       C 1084          Pattern 12      2D2D    026455          0 010 110 100 101 101
;       C 1085                          2D2D    026455          0 010 110 100 101 101
;       C 1086                          2D2D    026455          0 010 110 100 101 101
;       C 1087                          D2D2    151322          1 101 001 011 010 010
;       C 1088                          D2D2    151322          1 101 001 011 010 010
;       C 1089                          D2D2    151322          1 101 001 011 010 010
;       C 1090                          2D2D    026455          0 010 110 100 101 101
;       C 1091                          2D2D    026455          0 010 110 100 101 101
;       C 1092                          D2D2    151322          1 101 001 011 010 010
;       C 1093                          D2D2    151322          1 101 001 011 010 010
;       C 1094                          2D2D    026455          0 010 110 100 101 101
;       C 1095                          2D2D    026455          0 010 110 100 101 101
;       C 1096                          D2D2    151322          1 101 001 011 010 010
;       C 1097                          2D2D    026455          0 010 110 100 101 101
;       C 1098                          D2D2    151322          1 101 001 011 010 010
;       C 1099                          2D2D    026455          0 010 110 100 101 101
;       C 1100                          D2D2    151322          1 101 001 011 010 010
;       C 1101                          2D2D    026455          0 010 110 100 101 101
;       C 1102                          D2D2    151322          1 101 001 011 010 010
;       C 1103                          2D2D    026455          0 010 110 100 101 101
```

```
;       C 1104
;       C 1105      Pattern 13      6DB6    066666          0 110 110 110 110 110
;       C 1106
;       C 1107      Pattern 14      0001    000001          0 000 000 000 000 001
;       C 1108                      0002    000002          0 000 000 000 000 010
;       C 1109                      0004    000004          0 000 000 000 000 100
;       C 1110                      0008    000010          0 000 000 000 001 000
;       C 1111                      0010    000020          0 000 000 000 010 000
;       C 1112                      0020    000040          0 000 000 000 100 000
;       C 1113                      0040    000100          0 000 000 001 000 000
;       C 1114                      0080    000200          0 000 000 010 000 000
;       C 1115                      0100    000400          0 000 000 100 000 000
;       C 1116                      0200    001000          0 000 001 000 000 000
;       C 1117                      0400    002000          0 000 010 000 000 000
;       C 1118                      0800    004000          0 000 100 000 000 000
;       C 1119                      1000    010000          0 001 000 000 000 000
;       C 1120                      2000    020000          0 010 000 000 000 000
;       C 1121                      4000    040000          0 100 000 000 000 000
;       C 1122                      8000    100000          1 000 000 000 000 000
;       C 1123
;       C 1124      Pattern 15      FFFE    177776          1 111 111 111 111 110
;       C 1125                      FFFD    177775          1 111 111 111 111 101
;       C 1126                      FFFB    177773          1 111 111 111 111 011
;       C 1127                      FFF7    177767          1 111 111 111 110 111
;       C 1128                      FFEF    177757          1 111 111 111 101 111
;       C 1129                      FFDF    177737          1 111 111 111 011 111
;       C 1130                      FFBF    177677          1 111 111 110 111 111
;       C 1131                      FF7F    177577          1 111 111 101 111 111
;       C 1132                      FEFF    177377          1 111 111 011 111 111
;       C 1133                      FDFF    176777          1 111 110 111 111 111
;       C 1134                      FBFF    175777          1 111 101 111 111 111
;       C 1135                      F7FF    173777          1 111 011 111 111 111
;       C 1136                      EFFF    167777          1 110 111 111 111 111
;       C 1137                      DFFF    157777          1 101 111 111 111 111
;       C 1138                      BFFF    137777          1 011 111 111 111 111
;       C 1139                      7FFF    077777          0 111 111 111 111 111
;       C 1140
;       C 1141      Pattern 16      B6D9    133331          1 011 011 011 011 001
;       C 1142                      B6D9    133331          1 011 011 011 011 001
;       C 1143                      B6D9    133331          1 011 011 011 011 001
;       C 1144                      DB6C    155554          1 101 101 101 101 100
;       C 1145                      DB6C    155554          1 101 101 101 101 100
;       C 1146                      DB6C    155554          1 101 101 101 101 100
;       C 1147                      B6D9    133331          1 011 011 011 011 001
;       C 1148                      B6D9    133331          1 011 011 011 011 001
;       C 1149                      DB6C    155554          1 101 101 101 101 100
;       C 1150                      DB6C    155554          1 101 101 101 101 100
;       C 1151                      B6D9    133331          1 011 011 011 011 001
;       C 1152                      DB6C    155554          1 101 101 101 101 100
;       C 1153                      B6D9    133331          1 011 011 011 011 001
;       C 1154                      DB6C    155554          1 101 101 101 101 100
;       C 1155                      B6D9    133331          1 011 011 011 011 001
;       C 1156                      DB6C    155554          1 101 101 101 101 100
```

```
;       C 1157
;       C 1158          Pattern 17      LBN     LBN             LBN
;       C 1159                          8D36    106466      1 000 110 100 110 110
;       C 1160                          8D36    106466      1 000 110 100 110 110
;       C 1161                          72C9    071311      0 111 001 011 001 001
;       C 1162                          72C9    071311      0 111 001 011 001 001
;       C 1163                          72C9    071311      0 111 001 011 001 001
;       C 1164                          8D36    106466      1 000 110 100 110 110
;       C 1165                          8D36    106466      1 000 110 100 110 110
;       C 1166                          8D36    106466      1 000 110 100 110 110
;       C 1167                          8D36    106466      1 000 110 100 110 110
;       C 1168                          72C9    071311      0 111 001 011 001 001
;       C 1169                          72C9    071311      0 111 001 011 001 001
;       C 1170                          72C9    071311      0 111 001 011 001 001
;       C 1171                          72C9    071311      0 111 001 011 001 001
;       C 1172                          72C9    071311      0 111 001 011 001 001
;       C 1173                          8D36    106466      1 000 110 100 110 110
;       C 1174                          8D36    106466      1 000 110 100 110 110
;       C 1175                          8D36    106466      1 000 110 100 110 110
;       C 1176                          8D36    106466      1 000 110 100 110 110
;       C 1177                          8D36    106466      1 000 110 100 110 110
;       C 1178                          8D36    106466      1 000 110 100 110 110
;       C 1179
;       C 1180          Pattern 18      8D36    106466      1 000 110 100 110 110
;       C 1181                          LBN     LBN             LBN
;       C 1182                          72C9    071311      0 111 001 011 001 001
;       C 1183                          8D36    106466      1 000 110 100 110 110
;       C 1184                          8D36    106466      1 000 110 100 110 110
;       C 1185                          8D36    106466      1 000 110 100 110 110
;       C 1186                          72C9    071311      0 111 001 011 001 001
;       C 1187                          72C9    071311      0 111 001 011 001 001
;       C 1188                          72C9    071311      0 111 001 011 001 001
;       C 1189                          72C9    071311      0 111 001 011 001 001
;       C 1190                          8D36    106466      1 000 110 100 110 110
;       C 1191                          8D36    106466      1 000 110 100 110 110
;       C 1192                          8D36    106466      1 000 110 100 110 110
;       C 1193                          8D36    106466      1 000 110 100 110 110
;       C 1194                          8D36    106466      1 000 110 100 110 110
;       C 1195                          72C9    071311      0 111 001 011 001 001
;       C 1196                          72C9    071311      0 111 001 011 001 001
;       C 1197                          72C9    071311      0 111 001 011 001 001
;       C 1198                          72C9    071311      0 111 001 011 001 001
;       C 1199                          72C9    071311      0 111 001 011 001 001
;       C 1200                          72C9    071311      0 111 001 011 001 001
```

```
;       C 1201
;       C 1202          Pattern 19      LBN     LBN             LBN
;       C 1203                          8999    134631          1 011 100 110 011 001
;       C 1204                          8999    134631          1 011 100 110 011 001
;       C 1205                          4666    043146          0 100 011 001 100 110
;       C 1206                          4666    043146          0 100 011 001 100 110
;       C 1207                          4666    043146          0 100 011 001 100 110
;       C 1208                          8999    134631          1 011 100 110 011 001
;       C 1209                          8999    134631          1 011 100 110 011 001
;       C 1210                          8999    134631          1 011 100 110 011 001
;       C 1211                          8999    134631          1 011 100 110 011 001
;       C 1212                          4666    043146          0 100 011 001 100 110
;       C 1213                          4666    043146          0 100 011 001 100 110
;       C 1214                          4666    043146          0 100 011 001 100 110
;       C 1215                          4666    043146          0 100 011 001 100 110
;       C 1216                          4666    043146          0 100 011 001 100 110
;       C 1217                          8999    134631          1 011 100 110 011 001
;       C 1218                          8999    134631          1 011 100 110 011 001
;       C 1219                          8999    134631          1 011 100 110 011 001
;       C 1220                          8999    134631          1 011 100 110 011 001
;       C 1221                          8999    134631          1 011 100 110 011 001
;       C 1222                          8999    134631          1 011 100 110 011 001
;       C 1223
;       C 1224          Pattern 20      8999    134631          1 011 100 110 011 001
;       C 1225                          LBN     LBN                     LBN
;       C 1226                          4666    043146          0 100 011 001 100 110
;       C 1227                          8999    134631          1 011 100 110 011 001
;       C 1228                          8999    134631          1 011 100 110 011 001
;       C 1229                          8999    134631          1 011 100 110 011 001
;       C 1230                          4666    043146          0 100 011 001 100 110
;       C 1231                          4666    043146          0 100 011 001 100 110
;       C 1232                          4666    043146          0 100 011 001 100 110
;       C 1233                          4666    043146          0 100 011 001 100 110
;       C 1234                          8999    134631          1 011 100 110 011 001
;       C 1235                          8999    134631          1 011 100 110 011 001
;       C 1236                          8999    134631          1 011 100 110 011 001
;       C 1237                          8999    134631          1 011 100 110 011 001
;       C 1238                          8999    134631          1 011 100 110 011 001
;       C 1239                          4666    043146          0 100 011 001 100 110
;       C 1240                          4666    043146          0 100 011 001 100 110
;       C 1241                          4666    043146          0 100 011 001 100 110
;       C 1242                          4666    043146          0 100 011 001 100 110
;       C 1243                          4666    043146          0 100 011 001 100 110
;       C 1244                          4666    043146          0 100 011 001 100 110
;       C 1245
;       C 1246          Pattern 21      LBN     LBN                     LBN
;       C 1247
;         1248  )%
```

```
;        1249  %sbttl  PROGRAM HEADER'
;        1250
;        1251  library 'CNRQAA.L16 ;                    ! RDRX EXERCISER GLOBAL LIBRARY
;        1252
;        1253  require  BLSMAC.REQ ;                    ! DIAGNOSTIC SUPERVISOR LIBRARY
;        2742
;        2743  literal
;        2744      DS$NBR_OF_TESTS = 1;                 ! NUMBER OF TESTS IN THIS DIAGNOSTIC
;        2745
;        2746  EQUALS;
;        2747
;        2748  POINTER (ALL);
;        2749
;        2750  !+
;        2751  ! THE PROGRAM HEADER IS THE INTERFACE BETWEEN
;        2752  ! THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
;        2753  !
;        2754
;        2755  HEADER (%ascii' CNRQA', %ascii'A', %ascii'0', 32767, 1, PRI00);
;        2756
;        2757  !+
;        2758  ! THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
;        2759  ! IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
;        2760  !
;        2761
;        2762  DISPATCH (DS$NBR_OF_TESTS);
```

```
;          2763  %sbttl 'GLOBAL DATA SECTION
;          2764
;          2765  !+
;          2766  ! THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
;          2767  ! IN MORE THAN ONE TEST.
;          2768  !
;          2769
;          2770  psect
;          2771      global = $FFF$ (read, nowrite, execute, local, concatenate);
;          2772
;          2773  global
;          2774      CST : blockvector [MAX_CTLR, CST_LEN, word] field (CST_FIELDS),
;          2775                          ! RUN-TIME CONTROLLER STATUS TABLES
;          2776      CST_ADDR : ref block [CST_LEN, word] field (CST_FIELDS),
;          2777                          ! CONTROLLER STATUS TABLE ADDRESS OF "CURRENT" CONTROLLER
;          2778      DCT : blockvector [MAX_CTLR, DCT_LEN, word] field (DCT_FIELDS),
;          2779                          ! DRIVER CONTROLLER TABLES
;          2780      DCT_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS),
;          2781                          ! ADDRESS OF "CURRENT" DRIVER CONTROLLER TABLE
;          2782      RDRX_ADDR : ref rdrx field (RC_REG),
;          2783                          ! DEVICE ADDRESS OF "CURRENT" CONTROLLER
;          2784      IRDRX_ADDR : ref rdrx field (RC_REG),
;          2785                          ! DEVICE ADDRESS OF INTERRUPTING CONTROLLER
;          2786      DUPPKT : BLOCK [257, WORD] field (DP_FIELDS),
;          2787                          ! BUFFER CONTAINING DUP INFORMATION FROM RECEIVE AND SEND COMMANDS
;          2788      TALLY : vector [MAX_UNITS * TALLY_LEN, word] field (T_FIELDS),
;          2789                          ! STATISTICS TABLES
;          2790      T_ADDR : ref block [TALLY_LEN, word] field (T_FIELDS),
;          2791                          ! ADDRESS OF STATISTICS TABLE (TALLY) FOR CURRENT UNIT
;          2792      C_ERR_TBL : blockvector [MAX_CTLR, C_ERR_LEN, word] field (C_ERR_FIELDS),
;          2793                          ! STATISTICS TABLE FOR CONTROLLER ERRORS
;          2794      MSCP_PKT : blockvector [PKT_CNT, PKT_LEN, word] field (PKT_FIELDS),
;          2795                          ! MSCP PACKET POOL
;          2796      IPKT_ADDR : ref block [PKT_LEN, word] field (PKT_FIELDS),
;          2797                          ! ADDRESS OF AN MSCP PACKET (INTERUPT PROCESSING)
;          2798      PKT_USE : vector [PKT_CNT, byte, signed],
;          2799                          ! MSCP PACKET POOL ALLOCATION TABLE
;          2800      RETPKT : blockvector [RP_CNT, RP_LEN, word] field (RP_FIELDS),
;          2801                          ! RETURN PACKET POOL
;          2802      RP_USE : vector [RP_CNT, byte, signed],
;          2803                          ! RETURN PACKET POOL ALLOCATION TABLE
;          2804      RP_INDX : word,     ! CURRENT RETURN PACKET INDEX
;          2805      RP_ADDR : ref block [RP_LEN, word] field (RP_FIELDS),
;          2806                          ! CURRENT RETURN PACKET ADDRESS
;          2807      ELOG_PKT : blockvector [EP_CNT, EP_LEN, word] field (EP_FIELDS),
;          2808                          ! ERROR-LOG PACKET SAVE AREA
;          2809      BUFF_ADDR : vector [MAX_BUF_CNT],                       ! TABLE OF I/O BUFFER DESCRIPTORS
;          2810      BUFF_OWN : vector [MAX_BUF_CNT, byte, signed],          ! I/O BUFFER OWNERSHIP (CONTROLLER NUMBER)
;          2811      IODQ : vector [IODQ_LEN, byte],
;          2812                                      ! I/O DONE QUEUE - CIRCULAR QUEUE OF RETPKT INDECES
;          2813      IODQ_IN : word,                 ! I/O DONE QUEUE IN POINTER
;          2814      IODQ_OUT : word,                ! I/O DONE QUEUE OUT POINTER
;          2815      ENTRY_REASON : byte,            ! CURRENT OPERATOR COMMAND
;          2816      EOP_FLAG : byte,                ! END-OF-PASS FLAG
;          2817      DUP_FLAGS : WORD,               ! DUP FLAGS
;          2818      CCTLR : word,                   ! NUMBER OF "CURRENT" CONTROLLER
```

```
;         2819      CDISK : word,                          ! CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;         2820      CUOFF : word,                          ! CURRENT UNIT CST OFFSET
;         2821      CTLR_CNT : word,                       ! TOTAL NUMBER OF CONFIGURED CONTROLLERS
;         2822      DUR : vector [MAX_UNITS, byte],        ! DROP UNIT REASON
;         2823      QIO : vector [MAX_CTLR, byte],         ! NUMBER OF OUTSTANDING QIOS PER CONTROLLER
;         2824      FREE_MEM_ADDR,                         ! START OF FREE MEMORY
;         2825      BYTS_PER_QIO : word,                   ! SIZE (BYTES) OF AN I/O BUFFER
;         2826      ST_CODE : word,                        ! CURRENT STATUS CODE
;         2827      SB_CODE : word,                        ! CURRENT SUB-CODE
;         2828      STEP : word,                           ! CURRENT STEP IN HARD_INIT
;         2829      OF_RC : signed word,                   ! OFFSET (0 OR 2) TO READ IP OR SA
;         2830      SA_REG : word,                         ! STORAGE FOR SA REGISTER READS AND WRITES
;         2831      CMD_TIME : word,                       ! COMMAND TIMEOUT VALUE (IN SECONDS)
;         2832      NEX : word,                            ! NON-EXISTENT MEMORY TRAP INDICATOR
;         2833      CRN_LOW : word,                        ! COMMAND REF NUMBER OF LAST COMMAND SENT
;         2834      CRN_HIGH : word,                       ! COMMAND REF NUMBER (HI ORDER)
;         2835      P_INDEX : signed word,       ! CURRENT message PACKET INDEX
;         2836      S_DUPPKT : WORD,             ! DBN BYTE COUNTER
;         2837      S_PATTERN : WORD,            ! THE PATTERN WRITTEN TO DBN'S
;         2838      CREDIT_BAL : word,                     ! CREDIT BALANCE
;         2839      NEXT_PKT_USE : byte;                   ! POINTER TO NEXT ENTRY IN PKT_USE TABLE
;         2840
;         2841  ERRTBL;
```

```
;          2842  %sbttl 'GLOBAL TEXT SECTION'
;          2843
;          2844  !+
;          2845  ! THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
;          2846  ! MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
;          2847  ! MORE THAN ONE TEST.
;          2848  !-
;          2849
;          2850  global bind
;          2851  !
;          2852  ! HARDWARE DIALOG
;          2853  !
;          2854      HWQ1 = uplit (%asciz'IP ADDRESS'),
;          2855      HWQ2 = uplit (%asciz'VECTOR'),
;          2856      HWQ3 = uplit (%asciz'BR LEVEL'),
;          2857      HWQ4 = uplit (%asciz'RD/RX DRIVE NUMBER'),
;          2858      HWQ5 = uplit (%asciz'IS DISK AN RD51 WINCHESTER (NO, IMPLIES AN RX50 FLOPPY) '),
;          2859      HWQ6 = uplit (%asciz'STARTING LBN'),
;          2860      HWQ7 = uplit (%asciz'ENDING LBN (MAXIMUM -  RX50: 799, RD51: 21599)'),
;          2861      HWQ8 = uplit (%asciz'WRITE ON CUSTOMER DATA AREA ON THIS DISK'),
;          2862      HWQ9 = uplit (%asciz'** WARNING - CUSTOMER DATA AREA MAY BE OVERWRITTEN! ... CONFIRM'),
;          2863      HWQ10 = uplit (%asciz'ALSO EXERCISE ON DIAGNOSTIC AREA ( NON-CUSTOMER AREA)'),
;          2864      HWQ11 = uplit (%asciz'WRITE ON DIAGNOSTIC AREA'),
;          2865
;          2866  !
;          2867  ! SOFTWARE DIALOG
;          2868  !
;          2869      SWQ1  = uplit (%asciz'HARD ERROR LIMIT'),
;          2870      SWQ2  = uplit (%asciz'TRANSFER LIMIT IN MEGABYTES (0 FOR "QUICK PASS")'),
;          2871      SWQ4  = uplit (%asciz'RANDOM SEEK MODE'),
;          2872      SWQ7  = uplit (%asciz READ-COMPARES PERFORMED AT THE CONTROLLER'),
;          2873      SWQ9  = uplit (%asciz'WRITE-COMPARES PERFORMED AT THE CONTROLLER'),
;          2874      SWQ10 = uplit (%asciz'CHECK ALL WRITES AT HOST BY READING'),
;          2875      SWQ11 = uplit (%asciz'USER-DEFINED DATA PATTERN'),
;          2876      SWQ12 = uplit (%asciz'SELECT PRE-DEFINED DATA PATTERN (0 FOR SEQUENTIAL SELECTION)'),
;          2877      SWQ13 = uplit (%asciz'NUMBER OF WORDS IN DATA PATTERN (16 MAXIMUM)'),
;          2878      SWQ14 = uplit (%asciz'PATTERN VALUE'),
;          2879      SWQ15 = uplit (%asciz'CLEAR STATISTICAL TABLES AFTER PRINTING'),
;          2880      SWQ17 = uplit (%asciz'PERCENTAGE OF RD51 OPERATIONS OUT OF TOTAL OPERATIONS'),
;          2881      SWQ19 = uplit (%asciz'UNITS TO BE SELECTED AT RANDOM (NO, IMPLIES SEQUENTIAL)'),
;          2882  !   SWQ20 = uplit (%asciz'WANT TO REWRITE BLOCKS WHEN "FORCED ERROR" DETECTED ON READS '),
;          2883  !   SWQ21 = uplit (%asciz'IF "HALT ON ERROR" FLAG SET, WANT TO HALT ON HARD/SOFT ERRORS ALSO '),
;          2884      SWQ22 = uplit (%asciz'NUMBER OF DBNs READ AT ONE TIME (effects RD51s only)'),
;          2885      SWM1  = uplit (%asciz'THE REMAINING QUESTIONS ONLY APPLY TO UNPROTECTED DISKS'),
;          2886      NULL  = uplit (%asciz''),
;          2887
;          2888  !++
;          2889  !  THE FOLLOWING DBMs ARE DEBUG MESSAGES, AND SHOULD BE REMOVED BEFORE
;          2890  !  RELEASING THE PROGRAM.  THEY INCLUDE THE NAMES OF EACH ROUTINE, PLUS
;          2891  !  FORMAT STATEMENTS FOR PRINTING OUT OTHER INFORMATION.
;          2892  ! -
;          2893
;          2894  !       !JSD REV A --
;          2895  !       REMOVED DEBUG MESSAGES MARKED BY "!+"
;          2896  !       THE REST WERE ALREADY COMMENTED OUT.
;          2897  !
```

```
;        2898  !*    D8M5   = uplit (%asciz'%N%A** DROP UNIT %D2'),
;        2899  !*    D8M12  = uplit (%asciz'%N%A** PROC RETPKT: CONN ID = %D5%A RECEIVED'),
;        2900  !    D8M15  = uplit (%asciz'%N%A** MULTI DRIVE TEST'),
;        2901  !*    D8M18  = uplit (%asciz'%N%A** FATAL_ERROR: RETPKT NOT AVAILABLE'),
;        2902  !*    D8M19  = uplit (%asciz'%N%A** FSET_UPAR: CAN''T FIND DISK %D3%A IN CST %D1'),
;        2903  !*    D8M20  = uplit (%asciz'%N%A** BAD CONN ID = %D5%A RECEIVED FROM %O6'),
;        2904  !*    D8M21  = uplit (%asciz'%N%A** MESSAGE TYPE %D2%A RECEIVED IN MSCP PACKET'),
;        2905  !    D8M22  = uplit (%asciz'%N%A** SEQUEN: RETPKT NOT AVAILABLE'),
;        2906  !*    D8M23  = uplit (%asciz'%N%A** ERROR IN SET_CTLR_CHAR'),
;        2907  !*    D8M25  = uplit (%asciz'%N%A** CTLR TIMEOUT = %D3%A SECONDS'),
;        2908  !*    D8M26  = uplit (%asciz'%N%A** ERROR IN UNIT_INIT'),
;        2909  !    D8M27  = uplit (%asciz'%N%A** UNIT_INIT: RETPKT HAS BAD ENDCODE'),
;        2910  !    D8M28A = uplit (%asciz'%N%A** UNIT SIZE (LO) = %D5'),
;        2911  !    D8M28B = uplit (%asciz'%N%A** UNIT SIZE (HI) = %D5'),
;        2912  !*    D8M29  = uplit (%asciz'%N%A** ACCESS: RETPKT HAS BAD ENDCODE'),
;        2913   !   D8M32  = uplit (%asciz'%N%A** QIO_UNIT: CST %D1%A NO UNIT SELECTED'),
;        2914   !   D8M101 = uplit (%asciz'%N%A** UNIT # IS: %O6'),
;        2915   !   D8M104 = uplit (%asciz'%N%A** RX50 IS SELECTED'),
;        2916   !   D8M105 = uplit (%asciz'%N%A** RD51 IS SELECTED'),
;        2917  !*    D8M107 = uplit (%asciz'%N%A** ILLEGAL FUNCTION: %O6'),
;        2918  !*    D8M108 = uplit (%asciz'%N%A** COMMAND REF # %D5%A. NOT SENT BY HOST'),
;        2919  !*    D8M109 = uplit (%asciz'%N%A** UNKNOWN ERROR LOG FORMAT %D3%A. RECEIVED'),
;        2920  !*    D8M110 = uplit (%asciz'%N%A** ERROR-LOG SAVE AREA FULL'),
;        2921    ! D8M111 = uplit (%asciz'%N%A** REINIT MSCP EXERCISER'),
;        2922  !*    D8M112  = uplit (%asciz'%N%A** DUP_RSP: RETPKT NOT AVAILABLE'),
;        2923  !
;        2924  ! DROP UNIT MESSAGES
;        2925  !
;        2926      DU_MSG = uplit (%asciz'%N%AUNIT%D2%A DROPPED - '),
;        2927      DU_RSN = uplit (
;        2928                uplit (%asciz'%AUSER COMMAND%N'),
;        2929                uplit (%asciz'%ACONFIGURATION ERROR%N'),
;        2930                uplit (%asciz'%AINIT ERROR%N'),
;        2931                uplit (%asciz'%ATRANSFER LIMIT REACHED%N'),
;        2932                uplit (%asciz'%AERROR LIMIT REACHED%N'),
;        2933                uplit (%asciz'%AUNRECOVERABLE DEVICE ERROR%N'),
;        2934                uplit (%asciz'%AUNRECOVERABLE CONTROLLER ERROR%N'),
;        2935                uplit (%asciz'%AFAILED TO COME ONLINE%N'),
;        2936                uplit (%asciz'%AFAILED TO ACCESS LAST TRACK DURING INIT%N'),
;        2937                uplit (%asciz'%ADISK WRITE PROTECTED%N'),
;        2938                uplit (%asciz'%ACOMMAND TIME OUT%N'),
;        2939                uplit (%asciz'%AUNIT WENT TO AVAILABLE STATE%N')) : vector [11],
;        2940  !
;        2941  ! SYSTEM MESSAGES (PRINTF)
;        2942  !
;        2943      MSG_01 = uplit (%asciz'%N%APOWER DELAY - WAITING'),
;        2944      MSG_02 = uplit (%asciz'%N%AFUNCTIONAL TEST STARTED'),
;        2945      MSG_03 = uplit (%asciz'%N%N%AEXERCISER STARTED%N'),
;        2946  !
;        2947  ! REPORT MESSAGES (PRINTS)
;        2948  !
;        2949      RPT1   = uplit (%asciz'%N%N%AUNT DSK%S8%A# OF    # BYTES    # OF     # BYTES'),
;        2950      RPT2   = uplit (%asciz'%A  --HARD ERRORS-- --SOFT ERRORS--'),
;        2951      RPT3   = uplit (%asciz'%N%A #    #  TYPE  READS     READ   WRITES   WRITTEN'),
;        2952      RPT4   = uplit (%asciz'%A  SEK DAT DRV HST SEK DAT DRV HST'),
;        2953      RPT5   = uplit (%asciz'%N%A- -        -  - -       -- --   -  -   - - - - -'),
```

```
;        2954      RPT6  = uplit (%asciz'%A                -                  -          '),
;        2955      RPT7  = uplit (%asciz'%N%D2%D4%A  RX50 '),
;        2956      RPT8  = uplit (%asciz'%N%D2%D4%A  RD51 '),
;        2957      RPT9  = uplit (%asciz'%D4%Z3%D3%A,%Z3%A,%Z3'),
;        2958      RPT10 = uplit (%asciz'%D4%D4%D4%D4%D4%D4%D4%D4'),
;        2959      RPT11 = uplit (%asciz'%N%A .   .  CNTR      .  .........  .  ......... '),
;        2960      RPT12 = uplit (%asciz'%A   .  .%D4%A   .   .  .%D4%A   .'),
;        2961      RPT13 = UPLIT(%ASCIZ'%N%N%AUNIT  DISK          # OF   # BLKS       # OF    # BLKS '),
;        2962      RPT14 = UPLIT(%ASCIZ'%N%A  #    #     TYPE   READS     READ      WRITES  WRITTEN '),
;        2963      RPT15 = UPLIT(%ASCIZ'%N%A ---  - --                 -- --   --        --        -    '),
;        2964      RPT16 = UPLIT(%ASCIZ'%N%S1%D2%S4%D2%A   DBNRD51  %D6%S3%D6%S5%D6%S3%D6'),
;        2965
;        2966  !
;        2967  ! GENERAL ERROR MESSAGES
;        2968  !
;        2969  !       SYSTEM FATAL (ERRSF)
;        2970  !
;        2971      EGS_01 = uplit (%asciz'TOO MANY UNITS'),
;        2972      EGS_02 = uplit (%asciz'NOT ENOUGH FREE MEMORY FOR ALLOCATING READ/WRITE BUFFERS'),
;        2973  !
;        2974  !       DEVICE FATAL (ERRDF)
;        2975  !
;        2976      EGD_10 = uplit (%asciz'REGISTER EXISTENCE TEST FAILED'),
;        2977      EGD_11 = uplit (%asciz'VECTOR TEST FAILED'),
;        2978      EGD_12 = uplit (%asciz'BR LEVEL TEST FAILED'),
;        2979      EGD_13 = uplit (%asciz'INIT SEQUENCE FAILED'),
;        2980      EGD_14 = uplit (%asciz'FATAL CONTROLLER ERROR'),
;        2981      EGD_15 = uplit (%asciz'ONLINE FAILED'),
;        2982      EGD_16 = uplit (%asciz'WRITE-PROTECT CONFLICT'),
;        2983      EGD_17 = uplit (%asciz'ACCESS FAILED'),
;        2984      EGD_18 = uplit (%asciz'FATAL I/O ERROR'),
;        2985  !   EGD_19 = uplit (%asciz'CONTROLLER TIMEOUT'),
;        2986      EGD_20 = uplit (%asciz'FAILED TO SEND SET-CONTROLLER-CHARACTERISTICS COMMAND'),
;        2987      EGD_21 = uplit (%asciz'SET-CONTROLLER-CHARACTERISTICS RESPONSE HAS BAD ENDCODE OR FLAGS IN ERROR'),
;        2988      EGD_22 = uplit (%asciz'FAILED TO SEND ON-LINE COMMAND'),
;        2989      EGD_23 = uplit (%asciz'ON-LINE RESPONSE HAS BAD ENDCODE '),
;        2990  !
;        2991  !       HARD or SOFT (ERRHRD or ERRSOFT)
;        2992  !
;        2993      EGH_30 = uplit (%asciz'I/O REQUEST FAILED'),
;        2994  !
;        2995  ! BASIC ERROR MESSAGES (PRINTB)
;        2996  !
;        2997  !       SYSTEM FATAL (ERRSF)
;        2998  !
;        2999      EBS_01 = uplit (%asciz'%AMORE THAN %D2%A UNITS SPECIFIED'),
;        3000  !
;        3001  !       DEVICE FATAL (ERRDF)
;        3002  !
;        3003      EBD_10 = uplit (%asciz'%A* NO RESPONSE AT ADDRESS %O6'),
;        3004      EBD_12 = uplit (%asciz'%A* INCORRECT BR LEVEL FOR DEVICE %O6'),
;        3005      EBD_13 = uplit (%asciz'%A* STEP %D1%A READ ERROR'),
;        3006      EBD_14 = uplit (%asciz'%A* BAD SA CODE FROM DEVICE %O6'),
;        3007      EBD_18 = uplit (%asciz'%A* DISK%D2%A WENT OFFLINE'),
;        3008      EBD_19 = uplit (%asciz'%A* DEVICE %O6%A NOT PROCESSING COMMAND PACKETS'),
;        3009  !
```

```
;          3010  !       HARD or SOFT (ERRHRD or ERRSOFT)
;          3011  !
;          3012      EH_0 = UPLIT (%ASCIZ' - unrecognized MESSAGE TYPE'),
;          3013      EH_1 = UPLIT (%ASCIZ' - unrecognized connection id'),
;          3014      EH_2 = UPLIT (%ASCIZ' - unrecognized RETURN message'),
;          3015      EH_3  = UPLIT (%ASCIZ' - unrecognized RETURN PACKET'),
;          3016      EH_4 = UPLIT (%ASCIZ' - unrecognized CRN'),
;          3017      EH_5 = UPLIT (%ASCIZ' - UNRECOGNIZED OPCODE'),
;          3018      EH_6 = UPLIT (%ASCIZ' - MSCP STATUS CODE ERROR'),
;          3019      EH_7 = UPLIT (%ASCIZ' - DUP STATUS CODE ERROR'),
;          3020      EH_8 = UPLIT (%ASCIZ' - unrecognized STATUS CODE'),
;          3021      EH_9 = UPLIT (%ASCIZ' - LBN HOST COMPARE ERROR'),!
;          3022      EH_10 = UPLIT (%ASCIZ' - DBN HOST COMPARE ERROR'),
;          3023      EH_12 = UPLIT (%ASCIZ' - UNABLE TO LOAD DUP MEDIA '),
;          3024      EH_13 = UPLIT (%ASCIZ' - ERROR IN DUP-PKT WHEN USING CTLR LC PRG'),
;          3025      ERR_COD = uplit (
;          3026                uplit (%esciz'%AINVALID COMMAND'),
;          3027                uplit (%esciz'%ACOMMAND ABORTED'),
;          3028                uplit (%esciz'%AUNIT OFFLINE'),
;          3029                uplit (%esciz'%ATRANSITION TO AVAILABLE STATE'),
;          3030                uplit (%esciz'%AMEDIA FORMAT ERROR'),
;          3031                uplit (%esciz'%AWRITE-PROTECTED'),
;          3032                uplit (%esciz'%ADEVICE COMPARE ERROR'),
;          3033                uplit (%esciz'%ADATA ERROR'),
;          3034                uplit (%esciz'%AHOST BUFFER ACCESS ERROR'),
;          3035                uplit (%esciz'%ACONTROLLER ERROR'),
;          3036                uplit (%esciz'%ADRIVE ERROR'),
;          3037                uplit (%esciz'%AMESSAGE FROM INTERNAL DIAGNOSTICS'),
;          3038                uplit (%esciz'%AHOST COMPARE ERROR'),
;          3039                uplit (%esciz'%ACOMMAND TIMEOUT')) : vector [14],
;          3040  !
;          3041  !       ERROR LOG MESSAGE (ERRSOFT)
;          3042  !
;          3043      ELG_00  = uplit (%esciz'%N%AERROR LOG MESSAGE RECEIVED:%N'),
;          3044      ELG_FMT = uplit (
;          3045                uplit (%esciz'%A* CONTROLLER ERROR=%N'),
;          3046                uplit (%esciz'%A* HOST MEMORY ACCESS ERROR=%N'),
;          3047                uplit (%esciz'%A* DISK=%D2=%A - DISK TRANSFER ERROR=%N'),
;          3048                uplit (%esciz'%A* DISK=%D2=%A - "STANDARD DISK INTERCONNECT" ERROR=%N'),
;          3049                uplit (%esciz'%A* DISK=%D2=%A - "SMALL DISK" ERROR=%N')) : vector [5],
;          3050  !
;          3051  ! EXTENDED ERROR MESSAGES (PRINTX)
;          3052  !
;          3053
;          3054      EX_BDR = uplit (%esciz'%N%AI/O BUFFER ADDRESS FOR READ (32 BITS): %O6%A %O6%N'),
;          3055      EX_BDW = uplit (%esciz'%N%AI/O BUFFER ADDRESS FOR WRITE (32 BITS): %O6%A %O6'),
;          3056      EX_LBR = uplit (%esciz'%N%ALBN: (READ) %D5%A. (OCT %O6%A)'),
;          3057      EX_LBW = uplit (%esciz'%N%ALBN: (WRITE) %D5%A. (OCT %O6%A)'),
;          3058      EX_RBN = uplit (%esciz'%N%AREPLACEMENT BLOCK NO. %D5%A. (OCT %O6%A)'),
;          3059      EX_CBR = uplit (%esciz'%N%ABYTE COUNT IN READ COMMAND: %D5%A.'),
;          3060      EX_CBW = uplit (%esciz'%N%ABYTE COUNT IN WRITE COMMAND: %D5%A.'),
;          3061
;          3062  XX13 = UPLIT (%ASCIZ'%N%A * DISK : %D2'),
;          3063  XX14 = uplit (%esciz'%N%ASA: %O6'),
;          3064  XX15 = uplit (%esciz'%N%ASTATUS CODE: '),
;          3065  XX16 = uplit (%esciz'%N%ASTATUS SUB-CODE: '),
```

```
;        3066  XX17 = uplit (%asciz'%N%ACOMMAND:   '),
;        3067  XX18 = uplit (%asciz'%A-DUP-'),
;        3068  XX19 = uplit (%asciz'%A-MSCP-'),
;        3069  XX20 = uplit (%asciz'%A-COMPARE'),
;        3070  XX21 = uplit (%asciz'%N%ABAD BLOCK REPORTED: %O5%A.'),
;        3071  XX22 = uplit (%asciz'%N%ALBN: %D5%A. (OCT %O6%A)'),
;        3072  XX23 = UPLIT (%ASCIZ'%N%ADBN: %D5%A. (OCT %O6%A)'),
;        3073  XX24 = uplit (%asciz'%N%ABYTE COUNT IN COMMAND: %D5'),
;        3074  XX25 = uplit (%asciz'%N%AACTUAL # OF BYTES TRANSFERRED: %D5'),
;        3075  XX26 = uplit (%asciz'%N%AI/O BUFFER ADDRESS (32 BITS): %O6%A %O6'),
;        3076  XX27 = uplit (%asciz'%N%ACONTENTS OF RETURN PACKET:%N'),
;        3077  XX29 = UPLIT (%ASCIZ'%N%AMESSAGE TYPE: '),
;        3078  XX30 = UPLIT (%ASCIZ'%N%AMESSAGE NUMBERS: '),
;        3079  XX31 = UPLIT (%ASCIZ'%N%AMESSAGE ERROR CODES: '),
;        3080  XX32 = UPLIT (%ASCIZ'%N%ABYTE NUMBER: %D3'),
;        3081  XX33 = UPLIT (%ASCIZ'%N%ARANDOM WRITTEN WORD :%B16'),
;        3082  XX34 = UPLIT (%ASCIZ'%N%ARANDOM READ WORD bin:%B16%A oct:%O6'),
;        3083  XX35 = UPLIT (%ASCIZ'%N%ACRN : %O6'),
;        3084  !XX36 = UPLIT (%ASCIZ'%N%ATHE EXPECTED CRN : %O6'),
;        3085  XX37 = UPLIT (%ASCIZ'%A - UNKNOWN : %D2'),
;        3086  XX38 = UPLIT (%ASCIZ'%A - UNKNOWN CONNECTION ID: %D3%A - '),
;        3087  XX39 = UPLIT (%ASCIZ'%N%ACONTROLLER FLAGS:'),
;        3088  XX40 = UPLIT (%ASCIZ'%N%AUNIT FLAGS:'),
;        3089  XX41 = UPLIT (%ASCIZ'%N%AEND MESSAGE FLAGS:'),
;        3090
;        3091  !
;        3092  ! UNKNOWN RETURN MESSAGES
;        3093  !
;        3094        EB_DCT = UPLIT (%ASCIZ'%N%A DRIVER CONTROLLER TABLE = ADDR: %D6%N'),
;        3095        EB_COMM = UPLIT (%ASCIZ'%N%A CMD INT, RSP INT, COMMAND RING = ADDR: %D6%N'),
;        3096        EB_PKT = UPLIT (%ASCIZ'%N%A ALL PACKETS IN MESSAGE AREA'),
;        3097        EB_RAL = UPLIT (%ASCIZ'%N%A ALL RETURN PACKETS IN AREA'),
;        3098        EB_ADDR = UPLIT (%ASCIZ'%N%A ADDR: %D6%A     PACKET = %N'),
;        3099        EBNEX1 = UPLIT (%ASCIZ'%N%A ADDR OF RESPONSE RING TO BE POLLED %D6'),
;        3100        EB_NEX2 = UPLIT (%ASCIZ'%N%A ADDR OF MESSAGE PACKET RESPONSE RING SLOT POINTS TO %D6'),
;        3101        EBNEX3 = UPLIT (%ASCIZ'%N%A ADDR OF MESSAGE PACKET COMMAND RING SLOT POINTS TO %D6'),
;        3102  !
;        3103  ! CONFIGURATION ERROR MESSAGES (PRINTF)
;        3104  !
;        3105     CER_01 = uplit (%asciz'%N%ADUPLICATE UNIT:%D2%A AT IP: %O6'),
;        3106     CER_02 = uplit (%asciz'%N%AMORE THAN %D1%A DIFFERENT IP ADDRESSES'),
;        3107
;        3108  !
;        3109  ! MESSAGE TYPES
;        3110  !
;        3111     EX_SEQ = UPLIT (%ASCIZ'%A- SEQUENTIAL'),
;        3112     EX_CRD = UPLIT (%ASCIZ'%A- CREDIT NOTIFICATION'),
;        3113     EX_MTN = UPLIT (%ASCIZ'%A- MAINTENANCE'),
;        3114     EX_DGM = UPLIT (%ASCIZ'%A- DATAGRAM'),
;        3115  !
;        3116  ! commands
;        3117  !mscp
;        3118     EX_RD = uplit (%asciz'%AREAD'),
;        3119     EX_WRT = uplit (%asciz'%AWRITE'),
;        3120     EX_ACC = uplit (%asciz'%AACCESS'),
;        3121     EX_ONL = uplit (%asciz'%AON LINE'),
```

```
;        3122      EX_SCC = uplit (%asciz'%ASET CONTROL CHAR.'),
;        3123  !dup
;        3124      EX_GDS = uplit (%asciz'%AGET DUST STATUS'),
;        3125      EX_ESP = uplit (%asciz'%AEXECUTE SUPPLIED PRG'),
;        3126      EX_ELP = uplit (%asciz'%AEXECUTE LOCAL PRG'),
;        3127      EX_SDD = uplit (%asciz'%ASEND DATA'),
;        3128      EX_RCD = uplit (%asciz'%ARECEIVE DATA'),
;        3129      EX_ABP = uplit (%asciz'%AABORT'),
;        3130  !
;        3131  ! ERROR/EVENT SUB CODES (PRINTX)
;        3132  !
;        3133      SC_SDI = uplit (%asciz'%ASPIN-DOWN IGNORED'),
;        3134      SC_CON = uplit (%asciz'%ASTILL CONNECTED'),
;        3135      SC_DUP = uplit (%asciz'%ADUPLICATE UNIT NUMBER'),
;        3136      SC_ONL = uplit (%asciz'%AALREADY ONLINE'),
;        3137      SC_SON = uplit (%asciz'%ASTILL ONLINE'),
;        3138      SC_UNK = uplit (%asciz'%AUNIT UNKNOWN OR ONLINE TO ANOTHER CONTROLLER'),
;        3139      SC_VOL = uplit (%asciz'%ANO VOLUME MOUNTED OR DRIVE DISABLED BY SWITCH'),
;        3140      SC_IOP = uplit (%asciz'%AUNIT INOPERATIVE (RD51 WRITE FAULT'),
;        3141      SC_DIS = uplit (%asciz'%AUNIT DISABLED BY FIELD SERVICE OR INTERNAL DIAGNOSTICS'),
;        3142      SC_FER =  plit (%asciz'%A"FORCED ERROR" DETECTED WHILE ACCESSING FCT OR RCT'),
;        3143      SC_FE2 = uplit (%asciz'%ASECTOR WRITTEN WITH "FORCED ERROR" MODIFIER'),
;        3144      SC_ISH = uplit (%asciz'%AFCT OR RCT UNREADABLE - INVALID SECTOR HEADER'),
;        3145      SC_IS2 = uplit (%asciz'%AHEADER COMPARE ERROR (VALID HEADER NOT FOUND)'),
;        3146      SC_DST = uplit (%asciz'%AFCT OR RCT UNREADABLE - DATA SYNC TIMEOUT'),
;        3147      SC_DS2 = uplit (%asciz'%ADATA SYNC NOT FOUND (DATA SYNC TIMEOUT)'),
;        3148      SC_ECC = uplit (%asciz'%AFCT OR RCT UNREADABLE - UNCORRECTABLE ECC ERROR'),
;        3149      SC_ECD = uplit (%asciz'%AUNCORRECTABLE ECC ERROR'),
;        3150      SC_RCT = uplit (%asciz'%ARCT CORRUPTED'),
;        3151      SC_FUL = uplit (%asciz'%ANO REPLACEMENT BLOCK AVAILABLE (RCT FULL)'),
;        3152      SC_576 = uplit (%asciz'%ADISK NOT FORMATTED WITH 512 BYTE SECTORS'),
;        3153      SC_FCT = uplit (%asciz'%ADISK NOT FORMATTED OR FCT CORRUPTED'),
;        3154      SC_EC1 = uplit (%asciz'%AONE SYMBOL ECC ERROR'),
;        3155      SC_EC2 = uplit (%asciz'%ATWO SYMBOL ECC ERROR'),
;        3156      SC_EC3 = uplit (%asciz'%ATHREE SYMBOL ECC ERROR'),
;        3157      SC_EC4 = uplit (%asciz'%AFOUR SYMBOL ECC ERROR'),
;        3158      SC_EC5 = uplit (%asciz'%AFIVE SYMBOL ECC ERROR'),
;        3159      SC_EC6 = uplit (%asciz'%ASIX SYMBOL ECC ERROR'),
;        3160      SC_EC7 = uplit (%asciz'%ASEVEN SYMBOL ECC ERROR'),
;        3161      SC_EC8 = uplit (%asciz'%AEIGHT SYMBOL ECC ERROR'),
.        3162      SC_EC9 = uplit (%asciz'%ACORRECTABLE ERROR IN ECC FIELD'),
.        3163      SC_SWP = uplit (%asciz'%AUNIT SOFTWARE WRITE PROTECTED'),
;        3164      SC_HWP = uplit (%asciz'%AUNIT HARDWARE WRITE PROTECTED'),
;        3165      SC_ODA = uplit (%asciz'%AODD TRANSFER ADDRESS'),
;        3166      SC_ODB = uplit (%asciz'%AODD BYTE COUNT'),
;        3167      SC_NXM = uplit (%asciz'%ANON-EXISTENT HOST MEMORY'),
;        3168      SC_PAR = uplit (%asciz'%AHOST MEMORY PARITY ERROR'),
;        3169      SC_CTO = uplit (%asciz'%ACOMMAND TIMOUT OR RETRY LIMIT EXCEEDED'),
;        3170      SC_SDS = uplit (%asciz'%ASERIALIZER/DESERIALIZER OVERRUN OR UNDERRUN'),
;        3171      SC_EDC = uplit (%asciz'%AEDC ERROR'),
;        3172      SC_IDS = uplit (%asciz'%AINCONSISTENT INTERNAL DATA STRUCTURE'),
;        3173      SC_SRT = uplit (%asciz'%ADRIVE COMMAND TIMEOUT (NO RESPONSE OR SEEK INCOMPLETE) ),
;        3174      SC_SRI = uplit (%asciz'%ACONTROLLER DETECTED TRANSMISSION OR PROTOCOL ERROR'),
;        3175      SC_POE = uplit (%asciz'%APOSITION ERROR (MIS-SEEK)'),
;        3176      SC_RDY = uplit (%asciz'%ALOST READ/WRITE READY DURING/BETWEEN TRANSFERS'),
;        3177      SC_CLK = uplit (%asciz'%ADRIVE CLOCK DROPOUT'),
```

```
;        3178      SC_RSP = uplit (%asciz'%ALOST RECEIVER READY BETWEEN SECTORS'),
;        3179      SC_SUR = uplit (%asciz'%ADRIVE DETECTED ERROR'),
;        3180      SC_PSP = uplit (%asciz'%ACONTROLLER DETECTED PULSE OR STATE PARITY ERROR'),
;        3181  !
;        3182  ! MSCP END MESSAGE FLAGS
;        3183  !
;        3184  !F_0 = uplit (%asciz'%N%A) Bad Block Reported'),
;        3185  F_1 = uplit (%asciz'%N%A ) Bad Block Unreported'),
;        3186  F_2 = uplit (%asciz'%N%A ) Error Log Generated'),
;        3187  F_3 = uplit (%asciz'%N%A ) Serious Exception'),
;        3188  ! MSCP Controller Flags
;        3189  F_4 = uplit (%asciz'%N%A) Enable Attention Messages'),
;        3190  F_5 = uplit (%asciz'%N%A) Enable Miscellaneous Error Log Messages'),
;        3191  F_6 = uplit (%asciz'%N%A) Enable Other Hosts Error Log Messages'),
;        3192  F_7 = uplit (%asciz'%N%A) Enable This Hosts Error Log Messages'),
;        3193  F_8 = uplit (%asciz'%N%A) Controller Initiated Bad Block Rplcmnt'),
;        3194  F_9 = uplit (%asciz'%N%A) Shadowing'),
;        3195  F_10 = uplit (%asciz'%N%A) 576 Byte Sectors'),
;        3196  !
;        3197  ! MSCP UNIT FLAGS
;        3198  !
;        3199  F_11 = uplit (%asciz'%N%A) Compare Reads'),
;        3200  F_12 = uplit (%asciz'%N%A) Compare Writes'),
;        3201  F_13 = uplit (%asciz'%N%A) Controller Initiated Bad Block Rplcmnt'),
;        3202  F_14 = uplit (%asciz'%N%A) Inactive Shadow Set Unit'),
;        3203  F_15 = uplit (%asciz'%N%A) Removable Media'),
;        3204  F_16 = uplit (%asciz'%N%A) Suppress Caching (high speed)'),
;        3205  F_17 = uplit (%asciz'%N%A) Suppress Caching (low speed)'),
;        3206  F_18 = uplit (%asciz'%N%A) Write-back (non-voletile)'),
;        3207  F_19 = uplit (%asciz'%N%A) Write Protect (hardware)'),
;        3208  F_20 = uplit (%asciz'%N%A) Write Protect (software or volume)'),
;        3209  F_21 = uplit (%asciz'%N%A) 576 Byte Sectors'),
;        3210  !
;        3211  ! DUP RETURN PACKET MESSAGES
;        3212  ! STATUS CODE
;        3213      EBH_30 = uplit (%asciz'%A - SUCCESS'),
;        3214      EBH_44 = UPLIT (%ASCIZ'%A - INVALID COMMAND(SERVER nonIDLE or no media if EX-LC-PRG cmd)'),
;        3215      EBH_45 = UPLIT (%ASCIZ'%A - NO REGION AVAILABLE'),
;        3216      EBH_46 = UPLIT (%ASCIZ'%A - NO REGION SUITABLE'),
;        3217      EBH_47 = UPLIT (%ASCIZ'%A - PROGRAM NOT KNOWN (NO SUCH PROGRAM ON MEDIA)'),
;        3218      EBH_48 = UPLIT (%ASCIZ'%A - LOAD FAILURE (INPUT ERROR WHILE LOADING PROGRAM)'),
;        3219      EBH_49 = UPLIT (%ASCIZ'%A - STANDALONE (STANDALONE MODIFIER NOT SPECIFIED FOR STAND ALONE PRG.)'),
;        3220
;        3221  !
;        3222  ! DUP GET DUST STATUS FLAGS
;        3223      df_0 = uplit (%asciz'%N%A) One Server at a Time'),
;        3224      df_1 = uplit (%asciz'%N%A) Contains Local Media'),
;        3225      df_2 = uplit (%asciz'%N%A) Execute Local Prg cmd is UNSUPPORTED'),
;        3226      df_3 = uplit (%asciz'%N%A) Currently in Active State'),
;        3227  !
;        3228  ! DUP EXECUTE LOCAL PRG END FLAGS
;        3229      df_4 = uplit (%asciz'%N%A) Standalone Prg'),
;        3230      df_5 = uplit (%asciz'%N%A) Needs overlay'),
;        3231      df_6 = uplit (%asciz'%N%A) Needs Writeable/Readable Overlay'),
;        3232      df_7 = uplit (%asciz'%N%A) Uses Std Dup Dialog; REC/SEND/REC'),
;        3233
```

```
;          3234  ! DUP LOCAL PROGRAM PACKET MESSAGES
;          3235  !
;          3236      T_QUE = uplit (%asciz'%N%A  ** QUESTION'),
;          3237      T_DEF = uplit (%asciz'%N%A  ** DEFAULT QUESTION'),
;          3238      T_INF = uplit (%asciz'%N%A  ** INFORMATION'),
;          3239      T_TER = uplit (%asciz'%N%A  ** TERMINATION'),
;          3240      T_FAT = uplit (%asciz'%N%A  ** FATAL ERROR'),
;          3241      T_SPL = uplit (%asciz'%N%A  ** SPECIAL'),
;          3242  !    E_SUC = UPLIT (%ASCIZ'%N%A - SUCCESS'),
;          3243      E_UNT = UPLIT (%ASCIZ'%N%A - ILLEGAL UNIT NUMBER'),
;          3244      E_BLK = UPLIT (%ASCIZ'%N%A - ILLEGAL PHYSICAL OR RELATIVE BLOCK NUMBER'),
;          3245      E_DEV = UPLIT (%ASCIZ'%N%A - DEVICE ERROR'),
;          3246      E_ZER = UPLIT (%ASCIZ'%N%A   ZERO LENGHT MESSAGE'),
;          3247      M_ASC = UPLIT (%ASCIZ'%N%A -- ASCII INFORMATION '),
;          3248      M_BIN = UPLIT (%ASCIZ'%N%A -- NON-ACSII INFORMATION'),
;          3249      M_TER = UPLIT (%ASCIZ'%N%A -- TERMINATION MESSAGE'),
;          3250      M_COD = UPLIT (%ASCIZ'%N%A -  SUCCESS/FAILURE CODE'),
;          3251      M_DAT = UPLIT (%ASCIZ'%N%A -  SEND BINARY DATA'),
;          3252      M_UR = UPLIT (%ASCIZ'%N%A -- SEND UNIT NUMBER, RELATIVE DBN'),
;          3253      M_URP = UPLIT (%ASCIZ'%N%A -- SEND UNIT NUMBER, RELATIVE DBN, WRITE PATTERN'),
;          3254      M_UP = UPLIT (%ASCIZ'%N%A -- SEND UNIT NUMBER, PHYSICAL BLOCK NUMBER'),
;          3255      M_UL = UPLIT (%ASCIZ'%N%A -- SEND UNIT NUMBER, LOGICAL  BLOCK NUMBER '),
;          3256  !
;          3257  ! CONTROLLER GENERIC ERROR CODES
;          3258  !
;          3259      CNTR_ERR = uplit (
;          3260                  uplit (%asciz'%ACONTROLLER TIMEOUT'),
;          3261                  uplit (%asciz'%AENVELOPE/PACKET READ ERROR (PARITY OR TIMEOUT)'),
;          3262                  uplit (%asciz'%AENVELOPE/PACKET WRITE ERROR (PARITY OR TIMEOUT)'),
;          3263                  uplit (%asciz'%ACONTROLLER ROM AND RAM PARITY ERROR'),
;          3264                  uplit (%asciz'%ACONTROLLER RAM PARITY ERROR'),
;          3265                  uplit (%asciz'%ACONTROLLER ROM PARITY ERROR'),
;          3266                  uplit (%asciz'%ARING READ ERROR (PARITY OR TIMEOUT)'),
;          3267                  uplit (%asciz'%ARING WRITE ERROR (PARITY OR TIMEOUT)'),
;          3268                  uplit (%asciz'INTERRUPT MASTER FAILURE'),
;          3269                  uplit (%asciz'%AHOST ACCESS TIMEOUT (HIGHER LEVEL PROTOCOL DEPENDENT)'),
;          3270                  uplit (%asciz'%ACREDIT LIMIT EXCEEDED'),
;          3271                  uplit (%asciz'%AQ-BUS MASTER ERROR'),
;          3272                  uplit (%asciz'%ACONTROLLER FATAL ERROR'),
;          3273                  uplit (%asciz'%AINSTRUCTION LOOP TIMEOUT'),
;          3274                  uplit (%asciz'%AILLEGAL VIRTUAL CIRCUIT ID'),
;          3275                  uplit (%asciz'%AINTERRUPT VECTOR ILLEGAL'),
;          3276                  uplit (%asciz'%AMAINTENANCE READ/WRITE INVALID REGION IDENTIFIER'),
;          3277                  uplit (%asciz'%AMAINTENANCE WRITE LOAD TO NON-LOADABLE CONTROLLER'),
;          3278                  uplit (%asciz'%ACONTROLLER RAM ERROR (NON-PARITY)'),
;          3279                  uplit (%asciz'%AINIT SEQUENCE ERROR'),
;          3280                  uplit (%asciz'%AHIGHER LEVEL PROTOCOL INCOMPATIBILITY ERROR'),
;          3281                  uplit (%asciz'%APURGE/POLL HARDWARE FAILURE'),
;          3282                  uplit (%asciz'%AMAPPING REGISTER READ FAILURE (PARITY OR TIMEOUT)')) : vector [23],
;          3283  !
;          3284  ! RD/RX CONTROLLER DEPENDENT ERRORS CODES
;          3285  !
;          3286      RDRX_ERR = uplit (
;          3287                  uplit (%asciz'%AT11 CPU FAILURE'),
;          3288                  uplit (%asciz'%ANON-PARITY RAM ERROR'),
;          3289                  uplit (%asciz'%ASTATE MACHINE FAILURE - T11 ADDRESS REGISTER'),
```

```
;        3290                uplit (%asciz'%ASTATE MACHINE FAILURE - Q BUS ADDRESS REGISTER '),
;        3291                uplit (%asciz'%ASTATE MACHINE FAILURE   CRC REGISTER'),
;        3292                uplit (%asciz'%ASTATE MACHINE FAILURE   SERIALIZER/DESERIALIZER REGISTER'),
;        3293                uplit (%asciz'%ASTATE MACHINE FAILURE   WRONG HARDWARE VERSION')) : vector [7],
;        3294    !
;        3295    ! MISCELLANEOUS
;        3296    !
;        3297        EX_WRD = uplit (%asciz'%A %O6 '),
;        3298        EX_OP = uplit (%asciz'%Aoct %O4'),
;        3299        SPACE4    = uplit (%asciz'%S4'),
;        3300        CRLF      = uplit (%asciz'%N'),
;        3301        DASH      = uplit (%asciz'%A - '),
;        3302        ASTERISK = uplit (%asciz'%A* ');
```

```
;          3303  %sbttl  DEFAULT HARDWARE P TABLE
;          3304
;          3305  !+
;          3306  ! THE DEFAULT HARDWARE P TABLE CONTAINS DEFAULT VALUES OF
;          3307  ! THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
;          3308  ! IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P TABLES,
;          3309  ! AND IS USED AS A 'TEMPLATE" FOR BUILDING THE P TABLES.
;          3310  !-
;          3311
;          3312  BGNHW (DFPTBL);
;          3313
;          3314  global
;          3315      HWPT_IP_ADDR : word initial (INIT_IP_ADDR), ! IP ADDRESS
;          3316      HWPT_VECTOR : word initial (INIT_INTR VECT),! VECTOR ADDRESS
;          3317      HWPT_BR_LEVEL : word initial (INIT_BR LEVEL),!BR LEVEL
;          3318      HWPT_DISK : word initial (%o'100034 ),      ! DISK NUMBER, TYPE, PROTECTON BIT
;          3319      HWPT_S_TRK : word initial (0),              ! STARTING TRACK
;          3320      HWPT_E_TRK : word initial (RD MAX LBN);     ! ENDING TRACK
;          3321
;          3322  ENDHW;
```

```
;        3323  %sbttl  SOFTWARE P TABLE
;        3324
;        3325  !+
;        3326  ! THE SOFTWARE TABLE CONTAINS VARIOUS DATA USED BY THE
;        3327  ! PROGRAM AS OPERATIONAL PARAMETERS.  THESE PARAMETERS ARE
;        3328  ! SET UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR
;        3329  ! AT RUN TIME.
;        3330  !
;        3331
;        3332  BGNSW (SFPTBL);
;        3333
;        3334  global
;        3335      SWP_ERROR : word initial (32),              ! HARD ERROR LIMIT FOR DROPPING UNIT
;        3336      SWP_XFER : word initial (20),               ! TRANSFER LIMIT FOR DROPPING UNIT
;        3337      SWP_FLAGS : word initial (%o'202'),         ! FLAGS (SEE DOCUMENTATION)
;        3338      SWP_DPAT : word initial (0),                ! DATA PATTERN NUMBER
;        3339      SWP_UCNT : word initial (MAX_UDP_CNT),      ! USER DATA PATTERN COUNT
;        3340      SWP_RAT : word initial (99),                ! RD51 OPERATION RATIO
;        3341      dupround : word initial (18),               ! NUMBER OF DBN'S WRITTEN AT ONE TIME

;        3342      SWP_UDPAT : vector [MAX_UDP_CNT, word];     ! USER DATA PATTERN
;        3343
;        3344  ENDSW;
```

```
;          3345  %sbttl 'PROTECTION TABLE'
;          3346
;          3347  !+
;          3348  ! THIS TABLE IS USED BY THE RUNTIME SERVICES
;          3349  ! TO PROTECT THE LOAD MEDIA.
;          3350  !
;          3351
;          3352  BGNPROT (0, -1, 6);
;          3353
;          3354  !1ST ARG =      OFFSET INTO P TABLE FOR CSR ADDRESS
;          3355  !2ND ARG =      OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
;          3356  !3RD ARG =      OFFSET INTO P TABLE FOR DRIVE NUMBER
;          3357
;          3358  ENDPROT;
;          3359  end
;          3360
;          3361  eludom


                                                .TITLE  NRQAM1 RD/RX EXERCISER
                                                .IDENT  /V01.0/
                                                .ENABL  AMA

000000                                          .PSECT  $CODE$,  RO
000000    040   103   116           L$NAME::.ASCII  / CN/
000003    122   121   101                   .ASCII  /RQA/
000006    000                               .BYTE   0
000007    000                               .BYTE   0
000010                                  L$REV::
000010      101                               .ASCII  /A/
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$SOFT
000022  023650'                         L$HPTP::.WORD   L$HW
000024  023670'                         L$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$AUTO
000112  023750'         L$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  101 104 103             .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  125  116                .ASCII  /AUN/
002060  111  124  045                .ASCII  /IT%/
002063  104  062  045                .ASCII  /D2%/
002066  101  040  104                .ASCII  /A D/
002071  122  117  120                .ASCII  /ROP/
002074  120  105  104                .ASCII  /PED/
002077  040  055  040                .ASCII  / - /
002102  000  000                     .ASCII  <00><00>
002104  045  101  125     P.ABD:    .ASCII  /%AU/
002107  123  105  122                .ASCII  /SER/
002112  040  103  117                .ASCII  / CO/
002115  115  115  101                .ASCII  /MMA/
002120  116  104  045                .ASCII  /ND%/
002123  116  000  000                .ASCII  /N/<00><00>
002126  045  101  103     P.ABE:    .ASCII  /%AC/
002131  117  116  106                .ASCII  /ONF/
002134  111  107  125                .ASCII  /IGU/
002137  122  101  124                .ASCII  /RAT/
002142  111  117  116                .ASCII  /ION/
002145  040  105  122                .ASCII  / ER/
002150  122  117  122                .ASCII  /ROR/
002153  045  116  000                .ASCII  /%N/<00>
002156  045  101  111     P.ABF:    .ASCII  /%AI/
002161  116  111  124                .ASCII  /NIT/
002164  040  105  122                .ASCII  / ER/
002167  122  117  122                .ASCII  /ROR/
002172  045  116  000                .ASCII  /%N/<00>
002175  000                         .ASCII  <00>
002176  045  10'  124     P.ABG:    .ASCII  /%AT/
002201  122  101  116                .ASCII  /RAN/
002204  123  106  105                .ASCII  /SFE/
002207  122  040  114                .ASCII  /R L/
002212  111  115  111                .ASCII  /IMI/
002215  124  040  122                .ASCII  /T R/
002220  105  101  103                .ASCII  /EAC/
002223  110  105  104                .ASCII  /HED/
002226  045  116  000                .ASCII  /%N/<00>
002231  000                         .ASCII  <00>
002232  045  101  105     P.ABH:    .ASCII  /%AE/
```

```
002235  122  122  117          .ASCII  /RRO/
002240  122  040  114          .ASCII  /R L/
002243  111  115  111          .ASCII  /IMI/
002246  124  040  122          .ASCII  /T R/
002251  105  101  103          .ASCII  /EAC/
002254  110  105  104          .ASCII  /HED/
002257  045  116  000          .ASCII  /%N/<00>
002262  045  101  125  P.ABI:   .ASCII  /%AU/
002265  116  122  105          .ASCII  /NRE/
002270  103  117  126          .ASCII  /COV/
002273  105  122  101          .ASCII  /ERA/
002276  102  114  105          .ASCII  /BLE/
002301  040  104  105          .ASCII  / DE/
002304  126  111  103          .ASCII  /VIC/
002307  105  040  105          .ASCII  /E E/
002312  122  122  117          .ASCII  /RRO/
002315  122  045  116          .ASCII  /R%N/
002320  000  000               .ASCII  <00><00>
002322  045  101  125  P.ABJ:   .ASCII  /%AU/
002325  116  122  105          .ASCII  /NRE/
002330  103  117  126          .ASCII  /COV/
002333  105  122  101          .ASCII  /ERA/
002336  102  114  105          .ASCII  /BLE/
002341  040  103  117          .ASCII  / CO/
002344  116  124  122          .ASCII  /NTR/
002347  117  114  114          .ASCII  /OLL/
002352  105  122  040          .ASCII  /ER /
002355  105  122  122          .ASCII  /ERR/
002360  117  122  045          .ASCII  /OR%/
002363  116  000  000          .ASCII  /N/<00><00>
002366  045  101  106  P.ABK:   .ASCII  /%AF/
002371  101  111  114          .ASCII  /AIL/
002374  105  104  040          .ASCII  /ED /
002377  124  117  040          .ASCII  /TO /
002402  103  117  115          .ASCII  /COM/
002405  105  040  117          .ASCII  /E O/
002410  116  114  111          .ASCII  /NLI/
002413  116  105  045          .ASCII  /NE%/
002416  116  000               .ASCII  /N/<00>
002420  045  101  106  P.ABL:   .ASCII  /%AF/
002423  101  111  114          .ASCII  /AIL/
002426  105  104  040          .ASCII  /ED /
002431  124  117  040          .ASCII  /TO /
002434  101  103  103          .ASCII  /ACC/
002437  105  123  123          .ASCII  /ESS/
002442  040  114  101          .ASCII  / LA/
002445  123  124  040          .ASCII  /ST /
002450  124  122  101          .ASCII  /TRA/
002453  103  113  040          .ASCII  /CK /
002456  104  125  122          .ASCII  /DUR/
002461  111  116  107          .ASCII  /ING/
002464  040  111  116          .ASCII  / IN/
002467  111  124  045          .ASCII  /IT%/
002472  116  000               .ASCII  /N/<00>
002474  045  101  104  P.ABM:   .ASCII  /%AD/
002477  111  123  113          .ASCII  /ISK/
```

```
002502      040      127      122                    .ASCII  / WR/
002505      111      124      105                    .ASCII  /ITE/
002510      040      120      122                    .ASCII  / PR/
002513      117      124      105                    .ASCII  /OTE/
002516      103      124      105                    .ASCII  /CTE/
002521      104      045      116                    .ASCII  /D%N/
002524      000      000                             .ASCII  <00><00>
002526      045      101      103          P.ABN:    .ASCII  /%AC/
002531      117      115      115                    .ASCII  /OMM/
002534      101      116      104                    .ASCII  /AND/
002537      040      124      111                    .ASCII  / TI/
002542      115      105      040                    .ASCII  /ME /
002545      117      125      124                    .ASCII  /OUT/
002550      045      116      000                    .ASCII  /%N/<00>
002553      000                                      .ASCII  <00>
002554      045      101      125          P.ABO:    .ASCII  /%AU/
002557      116      111      124                    .ASCII  /NIT/
002562      040      127      105                    .ASCII  / WE/
002565      116      124      040                    .ASCII  /NT /
002570      124      117      040                    .ASCII  /TO /
002573      101      126      101                    .ASCII  /AVA/
002576      111      114      101                    .ASCII  /ILA/
002601      102      114      105                    .ASCII  /BLE/
002604      040      123      124                    .ASCII  / ST/
002607      101      124      105                    .ASCII  /ATE/
002612      045      116      000                    .ASCII  /%N/<00>
002615      000                                      .ASCII  <00>
002616  002104'                              P.ABC:  .WORD   P.ABD
002620  002126'                                      .WORD   P.ABE
002622  002156'                                      .WORD   P.ABF
002624  002176'                                      .WORD   P.ABG
002626  002232'                                      .WORD   P.ABH
002630  002263'                                      .WORD   P.ABI
002632  002322'                                      .WORD   P.ABJ
002634  002366'                                      .WORD   P.ABK
002636  002420'                                      .WORD   P.ABL
002640  002474'                                      .WORD   P.ABM
002642  002526'                                      .WORD   P.ABN
002644  002554'                                      .WORD   P.ABO
002646      045      116      045          P.ABP:    .ASCII  /%N%/
002651      101      120      117                    .ASCII  /APO/
002654      127      105      122                    .ASCII  /WER/
002657      040      104      105                    .ASCII  / DE/
002662      114      101      131                    .ASCII  /LAY/
002665      040      055      040                    .ASCII  / - /
002670      127      101      111                    .ASCII  /WAI/
002673      124      111      116                    .ASCII  /TIN/
002676      107      000                             .ASCII  /G/<00>
002700      045      116      045          P.ABQ:    .ASCII  /%N%/
002703      101      106      125                    .ASCII  /AFU/
002706      116      103      124                    .ASCII  /NCT/
002711      111      117      116                    .ASCII  /ION/
002714      101      114      040                    .ASCII  /AL /
002717      124      105      123                    .ASCII  /TES/
002722      124      040      123                    .ASCII  /T S/
002725      124      101      122                    .ASCII  /TAR/
```

```
002730  124  105  104                .ASCII  /TED/
002733  000                         .ASCII  <00>
002734  045  116  045       P.ABR:   .ASCII  /%N%/
002737  116  045  101                .ASCII  /N%A/
002742  105  130  105                .ASCII  /EXE/
002745  122  103  111                .ASCII  /RCI/
002750  123  105  122                .ASCII  /SER/
002753  040  123  124                .ASCII  / ST/
002756  101  122  124                .ASCII  /ART/
002761  105  104  045                .ASCII  /ED%/
002764  116  000                    .ASCII  /N/<00>
002766  045  116  045       P.ABS:   .ASCII  /%N%/
002771  116  045  101                .ASCII  /N%A/
002774  125  116  124                .ASCII  /UNT/
002777  040  104  123                .ASCII  / DS/
003002  113  045  123                .ASCII  /K%S/
003005  070  045  101                .ASCII  /8%A/
003010  043  040  117                .ASCII  /# O/
003013  106  040  040                .ASCII  /F  /
003016  040  043  040                .ASCII  / # /
003021  102  131  124                .ASCII  /BYT/
003024  105  123  040                .ASCII  /ES /
003027  040  040  043                .ASCII  /  #/
003032  040  117  106                .ASCII  / OF/
003035  040  040  040                .ASCII  /   /
003040  040  043  040                .ASCII  / # /
003043  102  131  124                .ASCII  /BYT/
003046  105  123  000                .ASCII  /ES/<00>
003051  000                         .ASCII  <00>
003052  045  101  040       P.ABT:   .ASCII  /%A /
003055  040  055  055                .ASCII  / --/
003060  110  101  122                .ASCII  /HAR/
003063  104  040  105                .ASCII  /D E/
003066  122  122  117                .ASCII  /RRO/
003071  122  123  055                .ASCII  /RS-/
003074  055  040  055                .ASCII  /- -/
003077  055  123  117                .ASCII  /-SO/
003102  106  124  040                .ASCII  /FT /
003105  105  122  122                .ASCII  /ERR/
003110  117  122  123                .ASCII  /ORS/
003113  055  055  000                .ASCII  /--/<00>
003116  045  116  045       P.ABU:   .ASCII  /%N%/
003121  101  040  043                .ASCII  /A #/
003124  040  040  040                .ASCII  /   /
003127  043  040  040                .ASCII  /#  /
003132  124  131  120                .ASCII  /TYP/
003135  105  040  040                .ASCII  /E  /
003140  122  105  101                .ASCII  /REA/
003143  104  123  040                .ASCII  /DS /
003146  040  040  040                .ASCII  /   /
003151  040  122  105                .ASCII  / RE/
003154  101  104  040                .ASCII  /AD /
003157  040  040  127                .ASCII  /  W/
003162  122  111  124                .ASCII  /RIT/
003165  105  123  040                .ASCII  /ES /
003170  040  040  127                .ASCII  /  W/
```

```
003112  122  111  124                .ASCII  /RII/
003116  124  105  116                .ASCII  /TEN/
003201  000                         .ASCII  <00>
003202  045  101  040        P.ABV:  .ASCII  /%A /
003205  040  123  105                .ASCII  / SE/
003210  113  040  104                .ASCII  /K D/
003213  101  124  040                .ASCII  /AT /
003216  104  122  126                .ASCII  /DRV/
003221  040  110  123                .ASCII  / HS/
003224  124  040  123                .ASCII  /T S/
003227  105  113  040                .ASCII  /EK /
003232  104  101  124                .ASCII  /DAT/
003235  040  104  122                .ASCII  / DR/
003240  126  040  110                .ASCII  /V H/
003243  123  124  000                .ASCII  /ST/<00>
003246  045  116  045        P.ABW:  .ASCII  /%N%/
003251  101  055  055                .ASCII  /A  /
003254  055  040  055                .ASCII  /  /
003257  055  055  040                .ASCII  /  /
003262  055  055  055                .ASCII  / -/
003265  055  040  040                .ASCII  /  /
003270  055  055  055                .ASCII  /  /
003273  055  055  040                .ASCII  /- /
003276  040  055  055                .ASCII  / -/
003301  055  055  055                .ASCII  /  /
003304  055  055  055                .ASCII  /  /
003307  055  040  055                .ASCII  /  /
003312  055  055  055                .ASCII  /  /
003315  055  055  040                .ASCII  /- /
003320  040  055  055                .ASCII  /  -/
003323  055  055  055                .ASCII  /---/
003326  055  055  055                .ASCII  /---/
003331  055  000  000                .ASCII  /-/<00><00>
003334  045  101  040        P.ABX:  .ASCII  /%A /
003337  055  055  055                .ASCII  / --/
003342  040  055  055                .ASCII  / --/
003345  055  040  055                .ASCII  /- -/
003350  055  055  040                .ASCII  /-- /
003353  055  055  055                .ASCII  / --/
003356  040  055  055                .ASCII  / --/
003361  055  040  055                .ASCII  /  /
003364  055  055  040                .ASCII  / - /
003367  055  055  055                .ASCII  /---/
003372  040  055  055                .ASCII  / --/
003375  055  000  000                .ASCII  /-/<00><00>
003400  045  116  045        P.ABY:  .ASCII  /%N%/
003403  104  062  045                .ASCII  /D2%/
003406  104  064  045                .ASCII  /D4%/
003411  101  040  040                .ASCII  /A  /
003414  122  130  065                .ASCII  /RX5/
003417  060  000  000                .ASCII  /0/<00><00>
003422  045  116  045        P.ABZ:  .ASCII  /%N%/
003425  104  062  045                .ASCII  /D2%/
003430  104  064  045                .ASCII  /D4%/
003433  101  040  040                .ASCII  /A  /
003436  122  104  065                .ASCII  /RD5/
```

```
003441  061  000  000                .ASCII  /1/<00><00>
003444  045  104  064  P.ACA:       .ASCII  /%D4/
003447  045  132  063                .ASCII  /%Z3/
003452  045  104  063                .ASCII  /%D3/
003455  045  101  054                .ASCII  /%A,/
003460  045  132  063                .ASCII  /%Z3/
003463  045  101  054                .ASCII  /%A,/
003466  045  132  063                .ASCII  /%Z3/
003471  000                         .ASCII  <00>
003472  045  104  064  P.ACB:       .ASCII  /%D4/
003475  045  104  064                .ASCII  /%D4/
003500  045  104  064                .ASCII  /%D4/
003503  045  104  064                .ASCII  /%D4/
003506  045  104  064                .ASCII  /%D4/
003511  045  104  064                .ASCII  /%D4/
003514  045  104  064                .ASCII  /%D4/
003517  045  104  064                .ASCII  /%D4/
003522  000  000                     .ASCII  <00><00>
003524  045  116  045  P.ACC:       .ASCII  /%N%/
003527  101  040  056                .ASCII  /A ./
003532  040  040  040                .ASCII  /   /
003535  056  040  040                .ASCII  /.  /
003540  103  116  124                .ASCII  /CNT/
003543  122  040  040                .ASCII  /R  /
003546  040  040  040                .ASCII  /   /
003551  040  056  040                .ASCII  / . /
003554  040  056  056                .ASCII  / ../
003557  056  056  056                .ASCII  /.../
003562  056  056  056                .ASCII  /.../
003565  056  040  040                .ASCII  /.  /
003570  040  040  040                .ASCII  /   /
003573  040  056  040                .ASCII  / . /
003576  040  056  056                .ASCII  / ../
003601  056  056  056                .ASCII  /.../
003604  056  056  056                .ASCII  /.../
003607  056  000  000                .ASCII  /./<00><00>
003612  045  101  040  P.ACD:       .ASCII  /%A /
003615  040  040  056                .ASCII  /  ./
003620  040  040  040                .ASCII  /   /
003623  056  045  104                .ASCII  /.%D/
003626  064  045  101                .ASCII  /4%A/
003631  040  040  040                .ASCII  /   /
003634  056  040  040                .ASCII  /.  /
003637  040  056  040                .ASCII  / . /
003642  040  040  056                .ASCII  /  ./
003645  045  104  064                .ASCII  /%D4/
003650  045  101  040                .ASCII  /%A /
003653  040  040  056                .ASCII  /  ./
003656  000  000                     .ASCII  <00><00>
003660  045  116  045  P.ACE:       .ASCII  /%N%/
003663  116  045  101                .ASCII  /N%A/
003666  125  116  111                .ASCII  /UNI/
003671  124  040  040                .ASCII  /T  /
003674  104  111  123                .ASCII  /DIS/
003677  113  040  040                .ASCII  /K  /
003702  040  040  040                .ASCII  /   /
```

```
003705  040  040  040              .ASCII  /   /
003710  040  040  040              .ASCII  /   /
003713  040  043  040              .ASCII  / 0 /
003716  117  106  040              .ASCII  /OF /
003721  040  040  043              .ASCII  /  0/
003724  040  102  114              .ASCII  / BL/
003727  113  123  040              .ASCII  /KS /
003732  040  040  040              .ASCII  /   /
003735  040  040  040              .ASCII  /   /
003740  043  040  117              .ASCII  /0 O/
003743  106  040  040              .ASCII  /F  /
003746  040  040  043              .ASCII  /  0/
003751  040  102  114              .ASCII  / BL/
003754  113  123  040              .ASCII  /KS /
003757  000                       .ASCII  <00>
003760  045  116  045  P.ACF:      .ASCII  /%N%/
003763  101  040  040              .ASCII  /A  /
003766  043  040  040              .ASCII  /0  /
003771  040  040  040              .ASCII  /   /
003774  043  040  040              .ASCII  /0  /
003777  040  040  040              .ASCII  /   /
004002  124  131  120              .ASCII  /TYP/
004005  105  040  040              .ASCII  /E  /
004010  040  122  105              .ASCII  / RE/
004013  101  104  123              .ASCII  /ADS/
004016  040  040  040              .ASCII  /   /
004021  040  040  122              .ASCII  /  R/
004024  105  101  104              .ASCII  /EAD/
004027  040  040  040              .ASCII  /   /
004032  040  040  040              .ASCII  /   /
004035  127  122  111              .ASCII  /WRI/
004040  124  105  123              .ASCII  /TES/
004043  040  040  127              .ASCII  /  W/
004046  122  111  124              .ASCII  /RIT/
004051  124  105  116              .ASCII  /TEN/
004054  040  000                  .ASCII  / /<00>
004056  045  116  045  P.ACG:      .ASCII  /%N%/
004061  101  055  055              .ASCII  /A--/
004064  055  055  040              .ASCII  /-- /
004067  040  055  055              .ASCII  / --/
004072  055  055  040              .ASCII  /-- /
004075  040  055  055              .ASCII  / --/
004100  055  055  055              .ASCII  /---/
004103  055  055  040              .ASCII  /-- /
004106  040  055  055              .ASCII  / --/
004111  055  055  055              .ASCII  /---/
004114  055  040  040              .ASCII  /-  /
004117  040  055  055              .ASCII  / --/
004122  055  055  055              .ASCII  /---/
004125  055  040  040              .ASCII  /-  /
004130  040  040  040              .ASCII  /   /
004133  055  055  055              .ASCII  /---/
004136  055  055  055              .ASCII  /---/
004141  040  040  040              .ASCII  /   /
004144  055  055  055              .ASCII  /---/
004147  055  055  055              .ASCII  /---/
```

```
004152  040  040  000              .ASCII  /  /<00>
004155  000                       .ASCII  <00>
004156  045  116  045  P.ACH:     .ASCII  /%N%/
004161  123  061  045              .ASCII  /S1%/
004164  104  062  045              .ASCII  /D2%/
004167  123  064  045              .ASCII  /S4%/
004172  104  062  045              .ASCII  /D2%/
004175  101  040  040              .ASCII  /A  /
004200  040  104  102              .ASCII  / DB/
004203  116  122  104              .ASCII  /NRD/
004206  065  061  040              .ASCII  /51 /
004211  040  045  104              .ASCII  / %D/
004214  066  045  123              .ASCII  /6%S/
004217  063  045  104              .ASCII  /3%D/
004222  066  045  123              .ASCII  /6%S/
004225  065  045  104              .ASCII  /5%D/
004230  066  045  123              .ASCII  /6%S/
004233  063  045  104              .ASCII  /3%D/
004236  066  000                  .ASCII  /6/<00>
004240  124  117  117  P.ACI:     .ASCII  /TOO/
004243  040  115  101              .ASCII  / MA/
004246  116  131  040              .ASCII  /NY /
004251  125  116  111              .ASCII  /UNI/
004254  124  123  000              .ASCII  /TS/<00>
004257  000                       .ASCII  <00>
004260  116  117  124  P.ACJ:     .ASCII  /NOT/
004263  040  105  116              .ASCII  / EN/
004266  117  125  107              .ASCII  /OUG/
004271  110  040  106              .ASCII  /H F/
004274  122  105  105              .ASCII  /REE/
004277  040  115  105              .ASCII  / ME/
004302  115  117  122              .ASCII  /MOR/
004305  131  040  106              .ASCII  /Y F/
004310  117  122  040              .ASCII  /OR /
004313  101  114  114              .ASCII  /ALL/
004316  117  103  101              .ASCII  /OCA/
004321  124  111  116              .ASCII  /TIN/
004324  107  040  122              .ASCII  /G R/
004327  105  101  104              .ASCII  /EAD/
004332  057  127  122              .ASCII  <57>/WR/
004335  111  124  105              .ASCII  /ITE/
004340  040  102  125              .ASCII  / BU/
004343  106  106  105              .ASCII  /FFE/
004346  122  123  000              .ASCII  /RS/<00>
004351  000                       .ASCII  <00>
004352  122  105  107  P.ACK:     .ASCII  /REG/
004355  111  123  124              .ASCII  /IST/
004360  105  122  040              .ASCII  /ER /
004363  105  130  111              .ASCII  /EXI/
004366  123  124  105              .ASCII  /STE/
004371  116  103  105              .ASCII  /NCE/
004374  040  124  105              .ASCII  / TE/
004377  123  124  040              .ASCII  /ST /
004402  106  101  111              .ASCII  /FAI/
004405  114  105  104              .ASCII  /LED/
004410  000  000                  .ASCII  <00><00>
```

```
004412  126  105  103  P.ACL:  .ASCII  /VEC/
004415  124  117  122         .ASCII  /TOR/
004420  040  124  105         .ASCII  / TE/
004423  123  124  040         .ASCII  /ST /
004426  106  101  111         .ASCII  /FAI/
004431  114  105  104         .ASCII  /LED/
004434  000  000              .ASCII  <00><00>
004436  102  122  040  P.ACM:  .ASCII  /BR /
004441  114  105  126         .ASCII  /LEV/
004444  105  114  040         .ASCII  /EL /
004447  124  105  123         .ASCII  /TES/
004452  124  040  106         .ASCII  /T F/
004455  101  111  114         .ASCII  /AIL/
004460  105  104  000         .ASCII  /ED/<00>
004463  000                   .ASCII  <00>
004464  111  116  111  P.ACN:  .ASCII  /INI/
004467  124  040  123         .ASCII  /T S/
004472  105  121  125         .ASCII  /EQU/
004475  105  116  103         .ASCII  /ENC/
004500  105  040  106         .ASCII  /E F/
004503  101  111  114         .ASCII  /AIL/
004506  105  104  000         .ASCII  /ED/<00>
004511  000                   .ASCII  <00>
004512  106  101  124  P.ACO:  .ASCII  /FAT/
004515  101  114  040         .ASCII  /AL /
004520  103  117  116         .ASCII  /CON/
004523  124  122  117         .ASCII  /TRO/
004526  114  114  105         .ASCII  /LLE/
004531  122  040  105         .ASCII  /R E/
004534  122  122  117         .ASCII  /RRO/
004537  122  000  000         .ASCII  /R/<00><00>
004542  117  116  114  P.ACP:  .ASCII  /ONL/
004545  111  116  105         .ASCII  /INE/
004550  040  106  101         .ASCII  / FA/
004553  111  114  105         .ASCII  /ILE/
004556  104  000              .ASCII  /D/<00>
004560  127  122  111  P.ACQ:  .ASCII  /WRI/
004563  124  105  055         .ASCII  /TE-/
004566  120  122  117         .ASCII  /PRO/
004571  124  105  103         .ASCII  /TEC/
004574  124  040  103         .ASCII  /T C/
004577  117  116  106         .ASCII  /ONF/
004602  114  111  103         .ASCII  /LIC/
004605  124  000  000         .ASCII  /T/<00><00>
004610  101  103  103  P.ACR:  .ASCII  /ACC/
004613  105  123  123         .ASCII  /ESS/
004616  040  106  101         .ASCII  / FA/
004621  111  114  105         .ASCII  /ILE/
004624  104  000              .ASCII  /D/<00>
004626  106  101  124  P.ACS:  .ASCII  /FAT/
004631  101  114  040         .ASCII  /AL /
004634  111  057  117         .ASCII  /I/<57>/O/
004637  040  105  122         .ASCII  / ER/
004642  122  117  122         .ASCII  /ROR/
004645  000                   .ASCII  <00>
004646  106  101  111  P.ACT:  .ASCII  /FAI/
```

```
004651  114  105  104            .ASCII  /LED/
004654  040  124  117            .ASCII  / TO/
004657  040  123  105            .ASCII  / SE/
004662  116  104  040            .ASCII  /ND /
004665  123  105  124            .ASCII  /SET/
004670  055  103  117            .ASCII  /-CO/
004673  116  124  122            .ASCII  /NTR/
004676  117  114  114            .ASCII  /OLL/
004701  105  122  055            .ASCII  /ER-/
004704  103  110  101            .ASCII  /CHA/
004707  122  101  103            .ASCII  /RAC/
004712  124  105  122            .ASCII  /TER/
004715  111  123  124            .ASCII  /IST/
004720  111  103  123            .ASCII  /ICS/
004723  040  103  117            .ASCII  / CO/
004726  115  115  101            .ASCII  /MMA/
004731  116  104  000            .ASCII  /ND/<00>
004734  123  105  124  P.ACU:    .ASCII  /SET/
004737  055  103  117            .ASCII  /-CO/
004742  116  124  122            .ASCII  /NTR/
004745  117  114  114            .ASCII  /OLL/
004750  105  122  055            .ASCII  /ER-/
004753  103  110  101            .ASCII  /CHA/
004756  122  101  103            .ASCII  /RAC/
004761  124  105  122            .ASCII  /TER/
004764  111  123  124            .ASCII  /IST/
004767  111  103  123            .ASCII  /ICS/
004772  040  122  105            .ASCII  / RE/
004775  123  120  117            .ASCII  /SPO/
005000  116  123  105            .ASCII  /NSE/
005003  040  110  101            .ASCII  / HA/
005006  123  040  102            .ASCII  /S B/
005011  101  104  040            .ASCII  /AD /
005014  105  116  104            .ASCII  /END/
005017  103  117  104            .ASCII  /COD/
005022  105  040  117            .ASCII  /E O/
005025  122  040  106            .ASCII  /R F/
005030  114  101  107            .ASCII  /LAG/
005033  123  040  111            .ASCII  /S I/
005036  116  040  105            .ASCII  /N E/
005041  122  122  117            .ASCII  /RRO/
005044  122  000                  .ASCII  /R/<00>
005046  106  101  111  P.ACV:    .ASCII  /FAI/
005051  114  105  104            .ASCII  /LED/
005054  040  124  117            .ASCII  / TO/
005057  040  123  105            .ASCII  / SE/
005062  116  104  040            .ASCII  /ND /
005065  117  116  055            .ASCII  /ON-/
005070  114  111  116            .ASCII  /LIN/
005073  105  040  103            .ASCII  /E C/
005076  117  115  115            .ASCII  /OMM/
005101  101  116  104            .ASCII  /AND/
005104  000  000                  .ASCII  <00><00>
005106  117  116  055  P.ACW:    .ASCII  /ON-/
005111  114  111  116            .ASCII  /LIN/
005114  105  040  122            .ASCII  /E R/
```

```
005117  105  123  120            .ASCII  /ESP/
005122  117  116  123            .ASCII  /ONS/
005125  105  040  110            .ASCII  /E H/
005130  101  123  040            .ASCII  /AS /
005133  102  101  104            .ASCII  /BAD/
005136  040  105  116            .ASCII  / EN/
005141  104  103  117            .ASCII  /DCO/
005144  104  105  000            .ASCII  /DE/<00>
005147  000                      .ASCII  <00>
005150  111  057  117  P.ACX:     .ASCII  /I/<57>/O/
005153  040  122  105            .ASCII  / RE/
005156  121  125  105            .ASCII  /QUE/
005161  123  124  040            .ASCII  /ST /
005164  106  101  111            .ASCII  /FAI/
005167  114  105  104            .ASCII  /LED/
005172  000  000                 .ASCII  <00><00>
005174  045  101  115  P.ACY:     .ASCII  /%AM/
005177  117  122  105            .ASCII  /ORE/
005202  040  124  110            .ASCII  / TH/
005205  101  116  040            .ASCII  /AN /
005210  045  104  062            .ASCII  /%D2/
005213  045  101  040            .ASCII  /%A /
005216  125  116  111            .ASCII  /UNI/
005221  124  123  040            .ASCII  /TS /
005224  123  120  105            .ASCII  /SPE/
005227  103  111  106            .ASCII  /CIF/
005232  111  105  104            .ASCII  /IED/
005235  000                      .ASCII  <00>
005236  045  101  052  P.ACZ:     .ASCII  /%A*/
005241  040  116  117            .ASCII  / NO/
005244  040  122  105            .ASCII  / RE/
005247  123  120  117            .ASCII  /SPO/
005252  116  123  105            .ASCII  /NSE/
005255  040  101  124            .ASCII  / AT/
005260  040  101  104            .ASCII  / AD/
005263  104  122  105            .ASCII  /DRE/
005266  123  123  040            .ASCII  /SS /
005271  045  117  066            .ASCII  /%O6/
005274  000  000                 .ASCII  <00><00>
005276  045  101  052  P.ADA:     .ASCII  /%A*/
005301  040  111  116            .ASCII  / IN/
005304  103  117  122            .ASCII  /COR/
005307  122  105  103            .ASCII  /REC/
005312  124  040  102            .ASCII  /T B/
005315  122  040  114            .ASCII  /R L/
005320  105  126  105            .ASCII  /EVE/
005323  114  040  106            .ASCII  /L F/
005326  117  122  040            .ASCII  /OR /
005331  104  105  126            .ASCII  /DEV/
005334  111  103  105            .ASCII  /ICE/
005337  040  045  117            .ASCII  / %O/
005342  066  000                 .ASCII  /6/<00>
005344  045  101  052  P.ADB:     .ASCII  /%A*/
005347  040  123  124            .ASCII  / ST/
005352  105  120  040            .ASCII  /EP /
005355  045  104  061            .ASCII  /%D1/
```

```
005360  045  101  040            .ASCII  /%A /
005363  122  105  101            .ASCII  /REA/
005366  104  040  105            .ASCII  /D E/
005371  122  122  117            .ASCII  /RRO/
005374  122  000                 .ASCII  /R/<00>
005376  045  101  052  P.ADC:     .ASCII  /%A*/
005401  040  102  101            .ASCII  / BA/
005404  104  040  123            .ASCII  /D S/
005407  101  040  103            .ASCII  /A C/
005412  117  104  105            .ASCII  /ODE/
005415  040  106  122            .ASCII  / FR/
005420  117  115  040            .ASCII  /OM /
005423  104  105  126            .ASCII  /DEV/
005426  111  103  105            .ASCII  /ICE/
005431  040  045  117            .ASCII  / %O/
005434  066  000                 .ASCII  /6/<00>
005436  045  101  052  P.ADD:     .ASCII  /%A*/
005441  040  104  111            .ASCII  / DI/
005444  123  113  045            .ASCII  /SK%/
005447  104  062  045            .ASCII  /D2%/
005452  101  040  127            .ASCII  /A W/
005455  105  116  124            .ASCII  /ENT/
005460  040  117  106            .ASCII  / OF/
005463  106  114  111            .ASCII  /FLI/
005466  116  105  000            .ASCII  /NE/<00>
005471  000                      .ASCII  <00>
005472  045  101  052  P.ADE:     .ASCII  /%A*/
005475  040  104  105            .ASCII  / DE/
005500  126  111  103            .ASCII  /VIC/
005503  105  040  045            .ASCII  /E %/
005506  117  066  045            .ASCII  /O6%/
005511  101  040  116            .ASCII  /A N/
005514  117  124  040            .ASCII  /OT /
005517  120  122  117            .ASCII  /PRO/
005522  103  105  123            .ASCII  /CES/
005525  123  111  116            .ASCII  /SIN/
005530  107  040  103            .ASCII  /G C/
005533  117  115  115            .ASCII  /OMM/
005536  101  116  104            .ASCII  /AND/
005541  040  120  101            .ASCII  / PA/
005544  103  113  105            .ASCII  /CKE/
005547  124  123  000            .ASCII  /TS/<00>
005552  040  055  040  P.ADF:     .ASCII  / - /
005555  165  156  162            .ASCII  /unr/
005560  145  143  157            .ASCII  /eco/
005563  147  156  151            .ASCII  /gni/
005566  172  145  144            .ASCII  /zed/
005571  040  115  105            .ASCII  / ME/
005574  123  123  101            .ASCII  /SSA/
005577  107  105  040            .ASCII  /GE /
005602  124  131  120            .ASCII  /TYP/
005605  105  000  000            .ASCII  /E/<00><00>
005610  040  055  040  P.ADG:     .ASCII  / - /
005613  165  156  162            .ASCII  /unr/
005616  145  143  157            .ASCII  /eco/
005621  147  156  151            .ASCII  /gni/
```

```
005624  172  145  144              .ASCII  /zed/
005627  040  143  157              .ASCII  / co/
005632  156  156  145              .ASCII  /nne/
005635  143  164  151              .ASCII  /cti/
005640  157  156  040              .ASCII  /on /
005643  151  144  000              .ASCII  /id/<00>
005646  040  055  040  P.ADH:      .ASCII  /  -  /
005651  165  156  162              .ASCII  /unr/
005654  145  143  157              .ASCII  /eco/
005657  147  156  151              .ASCII  /gni/
005662  172  145  144              .ASCII  /zed/
005665  040  122  105              .ASCII  / RE/
005670  124  125  122              .ASCII  /TUR/
005673  116  040  155              .ASCII  /N m/
005676  145  163  163              .ASCII  /ess/
005701  141  147  145              .ASCII  /age/
005704  000  000                   .ASCII  <00><00>
005706  040  055  040  P.ADI:      .ASCII  /  -  /
005711  165  156  162              .ASCII  /unr/
005714  145  143  157              .ASCII  /eco/
005717  147  156  151              .ASCII  /gni/
005722  172  145  144              .ASCII  /zed/
005725  040  122  105              .ASCII  / RE/
005730  124  125  122              .ASCII  /TUR/
005733  116  040  120              .ASCII  /N P/
005736  101  103  113              .ASCII  /ACK/
005741  105  124  000              .ASCII  /ET/<00>
005744  040  055  040  P.ADJ:      .ASCII  /  -  /
005747  165  156  162              .ASCII  /unr/
005752  145  143  157              .ASCII  /eco/
005755  147  156  151              .ASCII  /gni/
005760  172  145  144              .ASCII  /zed/
005763  040  103  122              .ASCII  / CR/
005766  116  000                   .ASCII  /N/<00>
005770  040  055  040  P.ADK:      .ASCII  /  -  /
005773  125  116  122              .ASCII  /UNR/
005776  105  103  117              .ASCII  /ECO/
006001  107  116  111              .ASCII  /GNI/
006004  132  105  104              .ASCII  /ZED/
006007  040  117  120              .ASCII  / OP/
006012  103  117  104              .ASCII  /COD/
006015  105  000  000              .ASCII  /E/<00><00>
006020  040  055  040  P.ADL:      .ASCII  /  -  /
006023  115  123  103              .ASCII  /MSC/
006026  120  040  123              .ASCII  /P S/
006031  124  101  124              .ASCII  /TAT/
006034  125  123  040              .ASCII  /US /
006037  103  117  104              .ASCII  /COD/
006042  105  040  105              .ASCII  /E E/
006045  122  122  117              .ASCII  /RRO/
006050  122  000                   .ASCII  /R/<00>
006052  040  055  040  P.ADM:      .ASCII  /  -  /
006055  104  125  120              .ASCII  /DUP/
006060  040  123  124              .ASCII  / ST/
006063  101  124  125              .ASCII  /ATU/
006066  123  040  103              .ASCII  /S C/
```

```
006071  117  104  105            .ASCII  /ODE/
006074  040  105  122            .ASCII  / ER/
006077  122  117  122            .ASCII  /ROR/
006102  000  000                 .ASCII  <00><00>
006104  040  055  040  P.ADN:    .ASCII  /  -  /
006107  165  156  162            .ASCII  /unr/
006112  145  143  157            .ASCII  /eco/
006115  147  156  151            .ASCII  /gni/
006120  172  145  144            .ASCII  /zed/
006123  040  123  124            .ASCII  / ST/
006126  101  124  125            .ASCII  /ATU/
006131  123  040  103            .ASCII  /S C/
006134  117  104  105            .ASCII  /ODE/
006137  000                      .ASCII  <00>
006140  040  055  040  P.ADO:    .ASCII  /  -  /
006143  114  102  116            .ASCII  /LBN/
006146  040  110  117            .ASCII  / HO/
006151  123  124  040            .ASCII  /ST /
006154  103  117  115            .ASCII  /COM/
006157  120  101  122            .ASCII  /PAR/
006162  105  040  105            .ASCII  /E E/
006165  122  122  117            .ASCII  /RRO/
006170  122  000                 .ASCII  /R/<00>
006172  040  055  040  P.ADP:    .ASCII  /  -  /
006175  104  102  116            .ASCII  /DBN/
006200  040  110  117            .ASCII  / HO/
006203  123  124  040            .ASCII  /ST /
006206  103  117  115            .ASCII  /COM/
006211  120  101  122            .ASCII  /PAR/
006214  105  040  105            .ASCII  /E E/
006217  122  122  117            .ASCII  /RRO/
006222  122  000                 .ASCII  /R/<00>
006224  040  055  040  P.ADQ:    .ASCII  /  -  /
006227  125  116  101            .ASCII  /UNA/
006232  102  114  105            .ASCII  /BLE/
006235  040  124  117            .ASCII  / TO/
006240  040  114  117            .ASCII  / LO/
006243  101  104  040            .ASCII  /AD /
006246  104  125  120            .ASCII  /DUP/
006251  040  115  105            .ASCII  / ME/
006254  104  111  101            .ASCII  /DIA/
006257  040  000  000            .ASCII  / /<00><00>
006262  040  055  040  P.ADR:    .ASCII  /  -  /
006265  105  122  122            .ASCII  /ERR/
006270  117  122  040            .ASCII  /OR /
006273  111  116  040            .ASCII  /IN /
006276  104  125  120            .ASCII  /DUP/
006301  055  120  113            .ASCII  /-PK/
006304  124  040  127            .ASCII  /T W/
006307  110  105  116            .ASCII  /HEN/
006312  040  125  123            .ASCII  / US/
006315  111  116  107            .ASCII  /ING/
006320  040  103  124            .ASCII  / CT/
006323  114  122  040            .ASCII  /LR /
006326  114  103  040            .ASCII  /LC /
006331  120  122  107            .ASCII  /PRG/
```

```
006334  000  000            .ASCII  <00><00>
006336  045  101  111  P.ADT: .ASCII  /%AI/
006341  116  126  101       .ASCII  /NVA/
006344  114  111  104       .ASCII  /LID/
006347  040  103  117       .ASCII  / CO/
006352  115  115  101       .ASCII  /MMA/
006355  116  104  000       .ASCII  /ND/<00>
006360  045  101  103  P.ADU: .ASCII  /%AC/
006363  117  115  115       .ASCII  /OMM/
006366  101  116  104       .ASCII  /AND/
006371  040  101  102       .ASCII  / AB/
006374  117  122  124       .ASCII  /ORT/
006377  105  104  000       .ASCII  /ED/<00>
006402  045  101  125  P.ADV. .ASCII  /%AU/
006405  116  111  124       .ASCII  /NIT/
006410  040  117  106       .ASCII  / OF/
006413  106  114  111       .ASCII  /FLI/
006416  116  105  000       .ASCII  /NE/<00>
006421  000                 .ASCII  <00>
006422  045  101  124  P.ADW: .ASCII  /%AT/
006425  122  101  116       .ASCII  /RAN/
006430  123  111  124       .ASCII  /SIT/
006433  111  117  116       .ASCII  /ION/
006436  040  124  117       .ASCII  / TO/
006441  040  101  126       .ASCII  / AV/
006444  101  111  114       .ASCII  /AIL/
006447  101  102  114       .ASCII  /ABL/
006452  105  040  123       .ASCII  /E S/
006455  124  101  124       .ASCII  /TAT/
006460  105  000            .ASCII  /E/<00>
006462  045  101  115  P.ADX: .ASCII  /%AM/
006465  105  104  111       .ASCII  /EDI/
006470  101  040  106       .ASCII  /A F/
006473  117  122  115       .ASCII  /ORM/
006476  101  124  040       .ASCII  /AT /
006501  105  122  122       .ASCII  /ERR/
006504  117  122  000       .ASCII  /OR/<00>
006507  000                 .ASCII  <00>
006510  045  101  127  P.ADY: .ASCII  /%AW/
006513  122  111  124       .ASCII  /RIT/
006516  105  055  120       .ASCII  /E-P/
006521  122  117  124       .ASCII  /ROT/
006524  105  103  124       .ASCII  /ECT/
006527  105  104  000       .ASCII  /ED/<00>
006532  045  101  104  P.ADZ: .ASCII  /%AD/
006535  105  126  111       .ASCII  /EVI/
006540  103  105  040       .ASCII  /CE /
006543  103  117  115       .ASCII  /COM/
006546  120  101  122       .ASCII  /PAR/
006551  105  040  105       .ASCII  /E E/
006554  122  122  117       .ASCII  /RRO/
006557  122  000  000       .ASCII  /R/<00><00>
006562  045  101  104  P.AEA: .ASCII  /%AD/
006565  101  124  101       .ASCII  /ATA/
006570  040  105  122       .ASCII  / ER/
006573  122  117  122       .ASCII  /ROR/
```

```
006576  000     000                     .ASCII  <00><00>
006600  045     101     110     P.AEB:  .ASCII  /%AH/
006603  117     123     124             .ASCII  /OST/
006606  040     102     125             .ASCII  / BU/
006611  106     106     105             .ASCII  /FFE/
006614  122     040     101             .ASCII  /R A/
006617  103     103     105             .ASCII  /CCE/
006622  123     123     040             .ASCII  /SS /
006625  105     122     122             .ASCII  /ERR/
006630  117     122     000             .ASCII  /OR/<00>
006633  000                             .ASCII  <00>
006634  045     101     103     P.AEC:  .ASCII  /%AC/
006637  117     116     124             .ASCII  /ONT/
006642  122     117     114             .ASCII  /ROL/
006645  114     105     122             .ASCII  /LER/
006650  040     105     122             .ASCII  / ER/
006653  122     117     122             .ASCII  /ROR/
006656  000     000                     .ASCII  <00><00>
006660  045     101     104     P.AED:  .ASCII  /%AD/
006663  122     111     126             .ASCII  /RIV/
006666  105     040     105             .ASCII  /E E/
006671  122     122     117             .ASCII  /RRO/
006674  122     000                     .ASCII  /R/<00>
006676  045     101     115     P.AEE:  .ASCII  /%AM/
006701  105     123     123             .ASCII  /ESS/
006704  101     107     105             .ASCII  /AGE/
006707  040     106     122             .ASCII  / FR/
006712  117     115     040             .ASCII  /OM /
006715  111     116     124             .ASCII  /INT/
006720  105     122     116             .ASCII  /ERN/
006723  101     114     040             .ASCII  /AL /
006726  104     111     101             .ASCII  /DIA/
006731  107     116     117             .ASCII  /GNO/
006734  123     124     111             .ASCII  /STI/
006737  103     123     000             .ASCII  /CS/<00>
006742  045     101     110     P.AEF:  .ASCII  /%AH/
006745  117     123     124             .ASCII  /OST/
006750  040     103     117             .ASCII  / CO/
006753  115     120     101             .ASCII  /MPA/
006756  122     105     040             .ASCII  /RE /
006761  105     122     122             .ASCII  /ERR/
006764  117     122     000             .ASCII  /OR/<00>
006767  000                             .ASCII  <00>
006770  045     101     103     P.AEG:  .ASCII  /%AC/
006773  117     115     115             .ASCII  /OMM/
006776  101     116     104             .ASCII  /AND/
007001  040     124     111             .ASCII  / TI/
007004  115     105     117             .ASCII  /MEO/
007007  125     124     000             .ASCII  /UT/<00>
007012  006336'                 P.ADS:  .WORD   P.ADT
007014  006360'                         .WORD   P.ADU
007016  006402'                         .WORD   P.ADV
007020  006422'                         .WORD   P.ADW
007022  006462'                         .WORD   P.ADX
007024  006510'                         .WORD   P.ADY
007026  006532                          .WORD   P.ADZ
```

```
007030  006562                                  .WORD   P.AEA
007032  006600                                  .WORD   P.AEB
007034  006634                                  .WORD   P.AEC
007036  006660                                  .WORD   P.AED
007040  006676                                  .WORD   P.AEE
007042  006742                                  .WORD   P.AEF
007044  006770                                  .WORD   P.AEG
007046     045     116     045          P.AEH:  .ASCII  /-N-/
007051     101     105     122                  .ASCII  /AER/
007054     122     117     122                  .ASCII  /ROR/
007057     040     114     117                  .ASCII  / LO/
007062     107     040     115                  .ASCII  /G M/
007065     105     123     123                  .ASCII  /ESS/
007070     101     107     105                  .ASCII  /AGE/
007073     040     122     105                  .ASCII  / RE/
007076     103     105     111                  .ASCII  /CEI/
007101     126     105     104                  .ASCII  /VED/
007104     072     045     116                  .ASCII  /:-N/
007107     000                                  .ASCII  <00>
007110     045     101     052          P.AEJ:  .ASCII  /-A*/
007113     040     103     117                  .ASCII  / CO/
007116     116     124     122                  .ASCII  /NTR/
007121     117     114     114                  .ASCII  /OLL/
007124     105     122     040                  .ASCII  /ER /
007127     105     122     122                  .ASCII  /ERR/
007132     117     122     045                  .ASCII  /OR-/
007135     116     000     000                  .ASCII  /N/<00><00>
007140     045     101     052          P.AEK:  .ASCII  /-A*/
007143     040     110     117                  .ASCII  / HO/
007146     123     124     040                  .ASCII  /ST /
007151     115     105     115                  .ASCII  /MEM/
007154     117     122     131                  .ASCII  /ORY/
007157     040     101     103                  .ASCII  / AC/
007162     103     105     123                  .ASCII  /CES/
007165     123     040     105                  .ASCII  /S E/
007170     122     122     117                  .ASCII  /RRO/
007173     122     045     116                  .ASCII  /R-N/
007176     000     000                          .ASCII  <00><00>
007200     045     101     052          P.AEL:  .ASCII  /-A*/
007203     040     104     111                  .ASCII  / D*/
007206     123     113     045                  .ASCII  /SK-/
007211     104     062     045                  .ASCII  /D2-/
007214     101     040     055                  .ASCII  /A -/
007217     040     104     111                  .ASCII  / DI/
007222     123     113     040                  .ASCII  /SK /
007225     124     122     101                  .ASCII  /TRA/
007230     116     123     106                  .ASCII  /NSF/
007233     105     122     040                  .ASCII  /ER /
007236     105     122     122                  .ASCII  /ERR/
007241     117     122     045                  .ASCII  /OR-/
007244     116     000                          .ASCII  /N/<00>
007246     045     101     052          P.AEM:  .ASCII  /-A*/
007251     040     104     111                  .ASCII  / DI/
007254     123     113     045                  .ASCII  /SK-/
007257     104     062     045                  .ASCII  /D2-/
007262     101     040     055                  .ASCII  /A  /
```

```
007269      040     042     123             .ASCII  / "S/
007274      124     101     116             .ASCII  /TAN/
007277      104     101     122             .ASCII  /DAR/
007302      104     040     104             .ASCII  /D D/
007305      111     123     113             .ASCII  /ISK/
007310      040     111     116             .ASCII  / IN/
007313      124     105     122             .ASCII  /TER/
007316      103     117     116             .ASCII  /CON/
007321      116     105     103             .ASCII  /NEC/
007324      124     042     040             .ASCII  /T" /
007327      105     122     122             .ASCII  /ERR/
007332      117     122     045             .ASCII  /OR%/
007335      116     000     000             .ASCII  /N/<00><00>
007340      045     101     052     P.AEN:  .ASCII  /%A*/
007343      040     104     111             .ASCII  / DI/
007346      123     113     045             .ASCII  /SK%/
007351      104     062     045             .ASCII  /D2%/
007354      101     040     055             .ASCII  /A  /
007357      040     042     123             .ASCII  / "S/
007362      115     101     114             .ASCII  /MAL/
007365      114     040     104             .ASCII  /L D/
007370      111     123     113             .ASCII  /ISK/
007373      042     040     105             .ASCII  /" E/
007376      122     122     117             .ASCII  /RRO/
007403      122     045     116             .ASCII  /R%N/
007406      000     000                     .ASCII  <00><00>
007410  007110'                     P.AEI:  .WORD   P.AEJ
007412  007140                              .WORD   P.AEK
007414  007200                              .WORD   P.AEL
007416  007246                              .WORD   P.AEM
007420  007354'                             .WORD   P.AEN
007422      045     116     045     P.AEO:  .ASCII  /%N%/
007425      101     111     057             .ASCII  /AI/<57>
007432      117     040     102             .ASCII  /O B/
007435      125     106     106             .ASCII  /UFF/
007440      105     122     040             .ASCII  /ER /
007443      101     104     104             .ASCII  /ADD/
007446      122     105     123             .ASCII  /RES/
007451      123     040     106             .ASCII  /S F/
007454      117     122     040             .ASCII  /OR /
007457      122     105     101             .ASCII  /REA/
007462      104     040     050             .ASCII  /D (/
007465      063     062     040             .ASCII  /32 /
007470      102     111     124             .ASCII  /BIT/
007473      123     051     072             .ASCII  /S):/
007476      040     045     117             .ASCII  / %O/
007501      066     045     101             .ASCII  /6%A/
007504      040     045     117             .ASCII  / %O/
007507      066     045     116             .ASCII  /6%N/
007512      000     000                     .ASCII  <00><00>
007514      045     116     045     P.AEP:  .ASCII  /%N%/
007517      101     111     057             .ASCII  /AI/<57>
007522      117     040     102             .ASCII  /O B/
007525      125     106     106             .ASCII  /UFF/
007530      105     122     040             .ASCII  /ER /
007533      101     104     104             .ASCII  /ADD/
```

```
007524  122  105  123                .ASCII  /RES/
007531  123  040  106                .ASCII  /S F/
007534  117  122  040                .ASCII  /OR /
007537  127  122  111                .ASCII  /WRI/
007542  124  105  040                .ASCII  /TE /
007545  050  063  062                .ASCII  /(32/
007550  040  102  111                .ASCII  / BI/
007553  124  123  051                .ASCII  /TS)/
007556  072  040  045                .ASCII  /: %/
007561  117  066  045                .ASCII  /06%/
007564  101  040  045                .ASCII  /A %/
007567  117  066  000                .ASCII  /06/<00>
007572  045  116  045  P.AEQ:        .ASCII  /%N%/
007575  101  114  102                .ASCII  /ALB/
007600  116  072  040                .ASCII  /N: /
007603  050  122  105                .ASCII  /(RE/
007606  101  104  051                .ASCII  /AD)/
007611  040  045  104                .ASCII  / %D/
007614  065  045  101                .ASCII  /5%A/
007617  056  040  050                .ASCII  /. (/
007622  117  103  124                .ASCII  /OCT/
007625  040  045  117                .ASCII  / %O/
007630  066  045  101                .ASCII  /6%A/
007633  051  000  000                .ASCII  /)/<00><00>
007636  045  116  045  P.AER:        .ASCII  /%N%/
007641  101  114  102                .ASCII  /ALB/
007644  116  072  040                .ASCII  /N: /
007647  050  127  122                .ASCII  /(WR/
007652  111  124  105                .ASCII  /ITE/
007655  051  040  045                .ASCII  /) %/
007660  104  065  045                .ASCII  /D5%/
007663  101  056  040                .ASCII  /A. /
007666  050  117  103                .ASCII  /(OC/
007671  124  040  045                .ASCII  /T %/
007674  117  066  045                .ASCII  /06%/
007677  101  051  000                .ASCII  /A)/<00>
007702  045  116  045  P.AES:        .ASCII  /%N%/
007705  101  122  105                .ASCII  /ARE/
007710  120  114  101                .ASCII  /PLA/
007713  103  105  115                .ASCII  /CEM/
007716  105  116  124                .ASCII  /ENT/
007721  040  102  114                .ASCII  / BL/
007724  117  103  113                .ASCII  /OCK/
007727  040  116  117                .ASCII  / NO/
007732  056  040  045                .ASCII  /. %/
007735  104  065  045                .ASCII  /D5%/
007740  101  056  040                .ASCII  /A. /
007743  050  117  103                .ASCII  /(OC/
007746  124  040  045                .ASCII  /T %/
007751  117  066  045                .ASCII  /06%/
007754  101  051  000                .ASCII  /A)/<00>
007757  000                         .ASCII  <00>
007760  045  116  045  P.AET:        .ASCII  /%N%/
007763  101  102  131                .ASCII  /ABY/
007766  124  105  040                .ASCII  /TE /
007771  103  117  125                .ASCII  /COU/
```

```
007774  116  124  040                .ASCII  /NT /
007777  111  116  040                .ASCII  /IN /
010002  122  105  101                .ASCII  /REA/
010005  104  040  103                .ASCII  /D C/
010010  117  115  115                .ASCII  /OMM/
010013  101  116  104                .ASCII  /AND/
010016  072  040  045                .ASCII  /: %/
010021  104  065  045                .ASCII  /D5%/
010024  101  056  000                .ASCII  /A./<00>
010027  000                          .ASCII  <00>
010030  045  116  045  P.AEU:        .ASCII  /%N%/
010033  101  102  131                .ASCII  /ABY/
010036  124  105  040                .ASCII  /TE /
010041  103  117  125                .ASCII  /COU/
010044  116  124  040                .ASCII  /NT /
010047  111  116  040                .ASCII  /IN /
010052  127  122  111                .ASCII  /WRI/
010055  124  105  040                .ASCII  /TE /
010060  103  117  115                .ASCII  /COM/
010063  115  101  116                .ASCII  /MAN/
010066  104  072  040                .ASCII  /D: /
010071  045  104  065                .ASCII  /%D5/
010074  045  101  056                .ASCII  /%A./
010077  000                          .ASCII  <00>
010100  045  116  045  P.AEV:        .ASCII  /%N%/
010103  101  040  052                .ASCII  /A *
010106  040  104  111                .ASCII  / DI/
010111  123  113  040                .ASCII  /SK /
010114  072  040  045                .ASCII  /: %/
010117  104  062  000                .ASCII  /D2/<00>
010122  045  116  045  P.AEW:        .ASCII  /%N%/
010125  101  123  101                .ASCII  /ASA/
010130  072  040  045                .ASCII  /: %/
010133  117  066  000                .ASCII  /06/<00>
010136  045  116  045  P.AEX:        .ASCII  /%N%/
010141  101  123  124                .ASCII  /AST/
010144  101  124  125                .ASCII  /ATU/
010147  123  040  103                .ASCII  /S C/
010152  117  104  105                .ASCII  /ODE/
010155  072  040  000                .ASCII  /: /<00>
010160  045  116  045  P.AEY:        .ASCII  /%N%/
010163  101  123  124                .ASCII  /AST/
010166  101  124  125                .ASCII  /ATU/
010171  123  040  123                .ASCII  /S S/
010174  125  102  055                .ASCII  /UB-/
010177  103  117  104                .ASCII  /COD/
010202  105  072  040                .ASCII  /E: /
010205  000                          .ASCII  <00>
010206  045  116  045  P.AEZ:        .ASCII  /%N%/
010211  101  103  117                .ASCII  /ACO/
010214  115  115  101                .ASCII  /MMA/
010217  116  104  072                .ASCII  /ND:/
010222  040  000                     .ASCII  / /<00>
010224  045  101  055  P.AFA:        .ASCII  /%A-/
010227  104  125  120                .ASCII  /DUP/
010232  055  000                     .ASCII  /-/<00>
```

```
010234  045  101  055        P.AFB:  .ASCII  /%A /
010237  115  123  103                .ASCII  /MSC/
010242  120  055  000                .ASCII  /P /<00>
010245  000                        .ASCII  <00>
010246  045  101  055        P.AFC:  .ASCII  /%A-/
010251  103  117  115                .ASCII  /COM/
010254  120  101  122                .ASCII  /PAR/
010257  105  000  000                .ASCII  /E/<00><00>
010262  045  116  045        P.AFD:  .ASCII  /%N%/
010265  101  102  101                .ASCII  /ABA/
010270  104  040  102                .ASCII  /D B/
010273  114  117  103                .ASCII  /LOC/
010276  113  040  122                .ASCII  /K R/
010301  105  120  117                .ASCII  /EPO/
010304  122  124  105                .ASCII  /RTE/
010307  104  072  040                .ASCII  /D: /
010312  045  117  065                .ASCII  /%05/
010315  045  101  056                .ASCII  /%A./
010320  000  000                    .ASCII  <00><00>
010322  045  116  045        P.AFE.  .ASCII  /%N%/
010325  101  114  102                .ASCII  /ALB/
010330  116  072  040                .ASCII  /N: /
010333  045  104  065                .ASCII  /%D5/
010336  045  101  056                .ASCII  /%A./
010341  040  050  117                .ASCII  / (O/
010344  103  124  040                .ASCII  /CT /
010347  045  117  066                .ASCII  /%O6/
010352  045  101  051                .ASCII  /%A)/
010355  000                        .ASCII  <00>
010356  045  116  045        P.AFF:  .ASCII  /%N%/
010361  101  104  102                .ASCII  /ADB/
010364  116  072  040                .ASCII  /N: /
010367  045  104  065                .ASCII  /%D5/
010372  045  101  056                .ASCII  /%A./
010375  040  050  117                .ASCII  / (O/
010400  103  124  040                .ASCII  /CT /
010403  045  117  066                .ASCII  /%O6/
010406  045  101  051                .ASCII  /%A)/
010411  000                        .ASCII  <00>
010412  045  116  045        P.AFG:  .ASCII  /%N%/
010415  101  102  131                .ASCII  /ABY/
010420  124  105  040                .ASCII  /TE /
010423  103  117  125                .ASCII  /COU/
010426  116  124  040                .ASCII  /NT /
010431  111  116  040                .ASCII  /IN /
010434  103  117  115                .ASCII  /COM/
010437  115  101  116                .ASCII  /MAN/
010442  104  072  040                .ASCII  /D: /
010445  045  104  065                .ASCII  /%D5/
010450  000  000                    .ASCII  <00><00>
010452  045  116  045        P.AFH:  .ASCII  /%N%/
010455  101  101  103                .ASCII  /AAC/
010460  124  125  101                .ASCII  /TUA/
010463  114  040  043                .ASCII  /L #/
010466  040  117  106                .ASCII  / OF/
010471  040  102  131                .ASCII  / BY/
```

```
010474  124     105     123             .ASCII  /TES/
010477  040     124     122             .ASCII  / TR/
010502  101     116     123             .ASCII  /ANS/
010505  106     105     122             .ASCII  /FER/
010510  122     105     104             .ASCII  /RED/
010513  072     040     045             .ASCII  /: %/
010516  104     065     000             .ASCII  /D5/<00>
010521  000                             .ASCII  <00>
010522  045     116     045     P.AFI:  .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     050             .ASCII  /S (/
010552  063     062     040             .ASCII  /32 /
010555  102     111     124             .ASCII  /BIT/
010560  123     051     072             .ASCII  /S):/
010563  040     045     117             .ASCII  / %O/
010566  066     045     101             .ASCII  /6%A/
010571  040     045     117             .ASCII  / %O/
010574  066     000                     .ASCII  /6/<00>
010576  045     116     045     P.AFJ:  .ASCII  /%N%/
010601  101     103     117             .ASCII  /ACO/
010604  116     124     105             .ASCII  /NTE/
010607  116     124     123             .ASCII  /NTS/
010612  040     117     106             .ASCII  / OF/
010615  040     122     105             .ASCII  / RE/
010620  124     125     122             .ASCII  /TUR/
010623  116     040     120             .ASCII  /N P/
010626  101     103     113             .ASCII  /ACK/
010631  105     124     072             .ASCII  /ET:/
010634  045     116     000             .ASCII  /%N/<00>
010637  000                             .ASCII  <00>
010640  045     116     045     P.AFK:  .ASCII  /%N%/
010643  101     115     105             .ASCII  /AME/
010646  123     123     101             .ASCII  /SSA/
010651  107     105     040             .ASCII  /GE /
010654  124     131     120             .ASCII  /TYP/
010657  105     072     040             .ASCII  /E: /
010662  000     000                     .ASCII  <00><00>
010664  045     116     045     P.AFL:  .ASCII  /%N%/
010667  101     115     105             .ASCII  /AME/
010672  123     123     101             .ASCII  /SSA/
010675  107     105     040             .ASCII  /GE /
010700  116     125     115             .ASCII  /NUM/
010703  102     105     122             .ASCII  /BER/
010706  123     072     040             .ASCII  /S: /
010711  000                             .ASCII  <00>
010712  045     116     045     P.AFM:  .ASCII  /%N%/
010715  101     115     105             .ASCII  /AME/
010720  123     123     101             .ASCII  /SSA/
010723  107     105     040             .ASCII  /GE /
010726  105     122     122             .ASCII  /ERR/
010731  117     122     040             .ASCII  /OR /
```

```
010734  103  117  104            .ASCII  /COD/
010737  105  123  072            .ASCII  /ES:/
010742  040  000                 .ASCII  / /<00>
010744  045  116  045  P.AFN:    .ASCII  /%N%/
010747  101  102  131            .ASCII  /ABY/
010752  124  105  040            .ASCII  /TE /
010755  116  125  115            .ASCII  /NUM/
010760  102  105  122            .ASCII  /BER/
010763  072  040  045            .ASCII  /: %/
010766  104  063  000            .ASCII  /D3/<00>
010771  000                      .ASCII  <00>
010772  045  116  045  P.AFO:    .ASCII  /%N%/
010775  101  122  101            .ASCII  /ARA/
011000  116  104  117            .ASCII  /NDO/
011003  115  040  127            .ASCII  /M W/
011006  122  111  124            .ASCII  /RIT/
011011  124  105  116            .ASCII  /TEN/
011014  040  127  117            .ASCII  / WO/
011017  122  104  040            .ASCII  /RD /
011022  072  045  102            .ASCII  /:%B/
011025  061  066  000            .ASCII  /16/<00>
011030  045  116  045  P.AFP:    .ASCII  /%N%/
011033  101  122  101            .ASCII  /ARA/
011036  116  104  117            .ASCII  /NDO/
011041  115  040  122            .ASCII  /M R/
011044  105  101  104            .ASCII  /EAD/
011047  040  127  117            .ASCII  / WO/
011052  122  104  040            .ASCII  /RD /
011055  142  151  156            .ASCII  /bin/
011060  072  045  102            .ASCII  /:%B/
011063  061  066  045            .ASCII  /16%/
011066  101  040  157            .ASCII  /A o/
011071  143  164  072            .ASCII  /ct:/
011074  045  117  066            .ASCII  /%O6/
011077  000                      .ASCII  <00>
011100  045  116  045  P.AFQ:    .ASCII  /%N%/
011103  101  103  122            .ASCII  /ACR/
011106  116  040  072            .ASCII  /N :/
011111  040  045  117            .ASCII  / %O/
011114  066  000                 .ASCII  /6/<00>
011116  045  101  040  P.AFR:    .ASCII  /%A /
011121  055  040  125            .ASCII  /- U/
011124  116  113  116            .ASCII  /NKN/
011127  117  127  116            .ASCII  /OWN/
011132  040  072  040            .ASCII  / : /
011135  045  104  062            .ASCII  /%D2/
011140  000  000                 .ASCII  <00><00>
011142  045  101  040  P.AFS:    .ASCII  /%A /
011145  055  040  125            .ASCII  /- U/
011150  116  113  116            .ASCII  /NKN/
011153  117  127  116            .ASCII  /OWN/
011156  040  103  117            .ASCII  / CO/
011161  116  116  105            .ASCII  /NNE/
011164  103  124  111            .ASCII  /CTI/
011167  117  116  040            .ASCII  /ON /
011172  111  104  072            .ASCII  /ID:/
```

```
011175  040  045  104                .ASCII  / %D/
011200  063  045  101                .ASCII  /3%A/
011203  040  055  000                .ASCII  /  /<00>
011206  045  116  045  P.AFT:        .ASCII  /%N%/
011211  101  103  117                .ASCII  /ACO/
011214  116  124  122                .ASCII  /NTR/
011217  117  114  114                .ASCII  /OLL/
011222  105  122  040                .ASCII  /ER /
011225  106  114  101                .ASCII  /FLA/
011230  107  123  072                .ASCII  /GS:/
011233  000                        .ASCII  <00>
011234  045  116  045  P.AFU:        .ASCII  /%N%/
011237  101  125  116                .ASCII  /AUN/
011242  111  124  040                .ASCII  /IT /
011245  106  114  101                .ASCII  /FLA/
011250  107  123  072                .ASCII  /GS:/
011253  000                        .ASCII  <00>
011254  045  116  045  P.AFV:        .ASCII  /%N%/
011257  101  105  116                .ASCII  /AEN/
011262  104  040  115                .ASCII  /D M/
011265  105  123  123                .ASCII  /ESS/
011270  101  107  105                .ASCII  /AGE/
011273  040  106  114                .ASCII  / FL/
011276  101  107  123                .ASCII  /AGS/
011301  072  000  000                .ASCII  /:/<00><00>
011304  045  116  045  P.AFW:        .ASCII  /%N%/
011307  101  040  104                .ASCII  /A D/
011312  122  111  126                .ASCII  /RIV/
011315  105  122  040                .ASCII  /ER /
011320  103  117  116                .ASCII  /CON/
011323  124  122  117                .ASCII  /TRO/
011326  114  114  105                .ASCII  /LLE/
011331  122  040  124                .ASCII  /R T/
011334  101  102  114                .ASCII  /ABL/
011337  105  040  075                .ASCII  /E =/
011342  040  101  104                .ASCII  / AD/
011345  104  122  072                .ASCII  /DR:/
011350  040  045  104                .ASCII  / %D/
011353  066  045  116                .ASCII  /6%N/
011356  000  000                    .ASCII  <00><00>
011360  045  116  045  P.AFX:        .ASCII  /%N%/
011363  101  040  103                .ASCII  /A C/
011366  115  104  040                .ASCII  /MD /
011371  111  116  124                .ASCII  /INT/
011374  054  040  122                .ASCII  /, R/
011377  123  120  040                .ASCII  /SP /
011402  111  116  124                .ASCII  /INT/
011405  054  040  103                .ASCII  /, C/
011410  117  115  115                .ASCII  /OMM/
011413  101  116  104                .ASCII  /AND/
011416  040  122  111                .ASCII  / RI/
011421  116  107  040                .ASCII  /NG /
011424  075  040  101                .ASCII  /= A/
011427  104  104  122                .ASCII  /DDR/
011432  072  040  045                .ASCII  /: %/
011435  104  066  045                .ASCII  /D6%/
```

```
011440  116  000                    .ASCII  /N/<00>
011442  045  116  045       P.AFY:  .ASCII  /%N%/
011445  101  040  101               .ASCII  /A A/
011450  114  114  040               .ASCII  /LL /
011453  120  101  103               .ASCII  /PAC/
011456  113  105  124               .ASCII  /KET/
011461  123  040  111               .ASCII  /S I/
011464  116  040  115               .ASCII  /N M/
011467  105  123  123               .ASCII  /ESS/
011472  101  107  105               .ASCII  /AGE/
011475  040  101  122               .ASCII  / AR/
011500  105  101  000               .ASCII  /EA/<00>
011503  000                        .ASCII  <00>
011504  045  116  045       P.AFZ:  .ASCII  /%N%/
011507  101  040  101               .ASCII  /A A/
011512  114  114  040               .ASCII  /LL /
011515  122  105  124               .ASCII  /RET/
011520  125  122  116               .ASCII  /URN/
011523  040  120  101               .ASCII  / PA/
011526  103  113  105               .ASCII  /CKE/
011531  124  123  040               .ASCII  /TS /
011534  111  116  040               .ASCII  /IN /
011537  101  122  105               .ASCII  /ARE/
011542  101  000                    .ASCII  /A/<00>
011544  045  116  045       P.AGA:  .ASCII  /%N%/
011547  101  040  101               .ASCII  /A A/
011552  104  104  122               .ASCII  /DDR/
011555  072  040  045               .ASCII  /: %/
011560  104  066  045               .ASCII  /D6%/
011563  101  040  040               .ASCII  /A  /
011566  040  040  040               .ASCII  /   /
011571  120  101  103               .ASCII  /PAC/
011574  113  105  124               .ASCII  /KET/
011577  040  075  040               .ASCII  / = /
011602  045  116  000               .ASCII  /%N/<00>
011605  000                        .ASCII  <00>
011606  045  116  045       P.AGB:  .ASCII  /%N%/
011611  101  040  101               .ASCII  /A A/
011614  104  104  122               .ASCII  /DDR/
011617  040  117  106               .ASCII  / OF/
011622  040  122  105               .ASCII  / RE/
011625  123  120  117               .ASCII  /SPO/
011630  116  123  105               .ASCII  /NSE/
011633  040  122  111               .ASCII  / RI/
011636  116  107  040               .ASCII  /NG /
011641  124  117  040               .ASCII  /TO /
011644  102  105  040               .ASCII  /BE /
011647  120  117  114               .ASCII  /POL/
011652  114  105  104               .ASCII  /LED/
011655  040  045  104               .ASCII  / %D/
011660  066  000                    .ASCII  /6/<00>
011662  045  116  045       P.AGC:  .ASCII  /%N%/
011665  101  040  101               .ASCII  /A A/
011670  104  104  122               .ASCII  /DDR/
011673  040  117  106               .ASCII  / OF/
011676  040  115  105               .ASCII  / ME/
```

```
011701  123  123  101          .ASCII  /SSA/
011704  107  105  040          .ASCII  /GE /
011707  120  101  103          .ASCII  /PAC/
011712  113  105  124          .ASCII  /KET/
011715  040  122  105          .ASCII  / RE/
011720  123  120  117          .ASCII  /SPO/
011723  116  123  105          .ASCII  /NSE/
011726  040  122  111          .ASCII  / RI/
011731  116  107  040          .ASCII  /NG /
011734  123  114  117          .ASCII  /SLO/
011737  124  040  120          .ASCII  /T P/
011742  117  111  116          .ASCII  /OIN/
011745  124  123  040          .ASCII  /TS /
011750  124  117  040          .ASCII  /TO /
011753  045  104  066          .ASCII  /%D6/
011756  000  000               .ASCII  <00><00>
011760  045  116  045  P.AGD:   .ASCII  /%N%/
011763  101  040  101          .ASCII  /A A/
011766  104  104  122          .ASCII  /DDR/
011771  040  117  106          .ASCII  / OF/
011774  040  115  105          .ASCII  / ME/
011777  123  123  101          .ASCII  /SSA/
012002  107  105  040          .ASCII  /GE /
012005  120  101  103          .ASCII  /PAC/
012010  113  105  124          .ASCII  /KET/
012013  040  103  117          .ASCII  / CO/
012016  115  115  101          .ASCII  /MMA/
012021  116  104  040          .ASCII  /ND /
012024  122  111  116          .ASCII  /RIN/
012027  107  040  123          .ASCII  /G S/
012032  114  117  124          .ASCII  /LOT/
012035  040  120  117          .ASCII  / PO/
012040  111  116  124          .ASCII  /INT/
012043  123  040  124          .ASCII  /S T/
012046  117  040  045          .ASCII  /O %/
012051  104  066  000          .ASCII  /D6/<00>
012054  045  116  045  P.AGE:   .ASCII  /%N%/
012057  101  104  125          .ASCII  /ADU/
012062  120  114  111          .ASCII  /PLI/
012065  103  101  124          .ASCII  /CAT/
012070  105  040  125          .ASCII  /E U/
012073  116  111  124          .ASCII  /NIT/
012076  072  045  104          .ASCII  /:%D/
012101  062  045  101          .ASCII  /2%A/
012104  040  101  124          .ASCII  / AT/
012107  040  111  120          .ASCII  / IP/
012112  072  040  045          .ASCII  /: %/
012115  117  066  000          .ASCII  /06/<00>
012120  045  116  045  P.AGF:   .ASCII  /%N%/
012123  101  115  117          .ASCII  /AMO/
012126  122  105  040          .ASCII  /RE /
012131  124  110  101          .ASCII  /THA/
012134  116  040  045          .ASCII  /N %/
012137  104  061  045          .ASCII  /D1%/
012142  101  040  104          .ASCII  /A D/
012145  111  106  106          .ASCII  /IFF/
```

```
012150  105  122  105            .ASCII  /ERE/
012153  116  124  040            .ASCII  /NT /
012156  111  120  040            .ASCII  /IP /
012161  101  104  104            .ASCII  /ADD/
012164  122  105  123            .ASCII  /RES/
012167  123  105  123            .ASCII  /SES/
012172  000  000                 .ASCII  <00><00>
012174  045  101  055  P.AGG:     .ASCII  /%A /
012177  040  123  105            .ASCII  / SE/
012202  121  125  105            .ASCII  /QUE/
012205  116  124  111            .ASCII  /NTI/
012210  101  114  000            .ASCII  /AL/<00>
012213  000                      .ASCII  <00>
012214  045  101  055  P.AGH:     .ASCII  /%A-/
012217  040  103  122            .ASCII  / CR/
012222  105  104  111            .ASCII  /EDI/
012225  124  040  116            .ASCII  /T N/
012230  117  124  111            .ASCII  /OTI/
012233  106  111  103            .ASCII  /FIC/
012236  101  124  111            .ASCII  /ATI/
012241  117  116  000            .ASCII  /ON/<00>
012244  045  101  055  P.AGI:     .ASCII  /%A-/
012247  040  115  101            .ASCII  / MA/
012252  111  116  124            .ASCII  /INT/
012255  105  116  101            .ASCII  /ENA/
012260  116  103  105            .ASCII  /NCE/
012263  000                      .ASCII  <00>
012264  045  101  055  P.AGJ:     .ASCII  /%A-/
012267  040  104  101            .ASCII  / DA/
012272  124  101  107            .ASCII  /TAG/
012275  122  101  115            .ASCII  /RAM/
012300  000  000                 .ASCII  <00><00>
012302  045  101  122  P.AGK:     .ASCII  /%AR/
012305  105  101  104            .ASCII  /EAD/
012310  000  000                 .ASCII  <00><00>
012312  045  101  127  P.AGL:     .ASCII  /%AW/
012315  122  111  124            .ASCII  /RIT/
012320  105  000                 .ASCII  /E/<00>
012322  045  101  101  P.AGM:     .ASCII  /%AA/
012325  103  103  105            .ASCII  /CCE/
012330  123  123  000            .ASCII  /SS/<00>
012333  000                      .ASCII  <00>
012334  045  101  117  P.AGN:     .ASCII  /%AO/
012337  116  040  114            .ASCII  /N L/
012342  111  116  105            .ASCII  /INE/
012345  000                      .ASCII  <00>
012346  045  101  123  P.AGO:     .ASCII  /%AS/
012351  105  124  040            .ASCII  /ET /
012354  103  117  116            .ASCII  /CON/
012357  124  122  117            .ASCII  /TRO/
012362  114  040  103            .ASCII  /L C/
012365  110  101  122            .ASCII  /HAR/
012370  056  000                 .ASCII  /./<00>
012372  045  101  107  P.AGP:     .ASCII  /%AG/
012375  105  124  040            .ASCII  /ET /
012400  104  125  123            .ASCII  /DUS/
```

```
012403  124  040  123            .ASCII  /T S/
012406  124  101  124            .ASCII  /TAT/
012411  125  123  000            .ASCII  /US/<00>
012414  045  101  105  P.AGQ:    .ASCII  /%AE/
012417  130  105  103            .ASCII  /XEC/
012422  125  124  105            .ASCII  /UTE/
012425  040  123  125            .ASCII  / SU/
012430  120  120  114            .ASCII  /PPL/
012433  111  105  104            .ASCII  /IED/
012436  040  120  122            .ASCII  / PR/
012441  107  000  000            .ASCII  /G/<00><00>
012444  045  101  105  P.AGR:    .ASCII  /%AE/
012447  130  105  103            .ASCII  /XEC/
012452  125  124  105            .ASCII  /UTE/
012455  040  114  117            .ASCII  / LO/
012460  103  101  114            .ASCII  /CAL/
012463  040  120  122            .ASCII  / PR/
012466  107  000                 .ASCII  /G/<00>
012470  045  101  123  P.AGS:    .ASCII  /%AS/
012473  105  116  104            .ASCII  /END/
012476  040  104  101            .ASCII  / DA/
012501  124  101  000            .ASCII  /TA/<00>
012504  045  101  122  P.AGT:    .ASCII  /%AR/
012507  105  103  105            .ASCII  /ECE/
012512  111  126  105            .ASCII  /IVE/
012515  040  104  101            .ASCII  / DA/
012520  124  101  000            .ASCII  /TA/<00>
012523  000                      .ASCII  <00>
012524  045  101  101  P.AGU:    .ASCII  /%AA/
012527  102  117  122            .ASCII  /BOR/
012532  124  000                 .ASCII  /T/<00>
012534  045  101  123  P.AGV:    .ASCII  /%AS/
012537  120  111  116            .ASCII  /PIN/
012542  055  104  117            .ASCII  /-DO/
012545  127  116  040            .ASCII  /WN /
012550  111  107  116            .ASCII  /IGN/
012553  117  122  105            .ASCII  /ORE/
012556  104  000                 .ASCII  /D/<00>
012560  045  101  123  P.AGW:    .ASCII  /%AS/
012563  124  111  114            .ASCII  /TIL/
012566  114  040  103            .ASCII  /L C/
012571  117  116  116            .ASCII  /ONN/
012574  105  103  124            .ASCII  /ECT/
012577  105  104  000            .ASCII  /ED/<00>
012602  045  101  104  P.AGX:    .ASCII  /%AD/
012605  125  120  114            .ASCII  /UPL/
012610  111  103  101            .ASCII  /ICA/
012613  124  105  040            .ASCII  /TE /
012616  125  116  111             ASCII  /UNI/
012621  124  040  116            .ASCII  /T N/
012624  125  115  102            .ASCII  /UMB/
012627  105  122  000            .ASCII  /ER/<00>
012632  045  101  101  P.AGY:    .ASCII  /%AA/
012635  114  122  105            .ASCII  /LRE/
012640  101  104  131            .ASCII  /ADY/
012643  040  117  116            .ASCII  / ON/
```

```
012646  114  111  116                .ASCII  /LIN/
012651  105  000  000                .ASCII  /E/<00><00>
012654  045  101  123  P.AGZ:        .ASCII  /%AS/
012657  124  111  114                .ASCII  /TIL/
012662  114  040  117                .ASCII  /L O/
012665  116  114  111                .ASCII  /NLI/
012670  116  105  000                .ASCII  /NE/<00>
012673  000                        .ASCII  <00>
012674  045  101  125  P.AHA:        .ASCII  /%AU/
012677  116  111  124                .ASCII  /NIT/
012702  040  125  116                .ASCII  / UN/
012705  113  116  117                .ASCII  /KNO/
012710  127  116  040                .ASCII  /WN /
012713  117  122  040                .ASCII  /OR /
012716  117  116  114                .ASCII  /ONL/
012721  111  116  105                .ASCII  /INE/
012724  040  124  117                .ASCII  / TO/
012727  040  101  116                .ASCII  / AN/
012732  117  124  110                .ASCII  /OTH/
012735  105  122  040                .ASCII  /ER /
012740  103  117  116                .ASCII  /CON/
012743  124  122  117                .ASCII  /TRO/
012746  114  114  105                .ASCII  /LLE/
012751  122  000  000                .ASCII  /R/<00><00>
012754  045  101  116  P.AHB:        .ASCII  /%AN/
012757  117  040  126                .ASCII  /O V/
012762  117  114  125                .ASCII  /OLU/
012765  115  105  040                .ASCII  /ME /
012770  115  117  125                .ASCII  /MOU/
012773  116  124  105                .ASCII  /NTE/
012776  104  040  117                .ASCII  /D O/
013001  122  040  104                .ASCII  /R D/
013004  122  111  126                .ASCII  /RIV/
013007  105  040  104                .ASCII  /E D/
013012  111  123  101                .ASCII  /ISA/
013015  102  114  105                .ASCII  /BLE/
013020  104  040  102                .ASCII  /D B/
013023  131  040  123                .ASCII  /Y S/
013026  127  111  124                .ASCII  /WIT/
013031  103  110  000                .ASCII  /CH/<00>
013034  045  101  125  P.AHC:        .ASCII  /%AU/
013037  116  111  124                .ASCII  /NIT/
013042  040  111  116                .ASCII  / IN/
013045  117  120  105                .ASCII  /OPE/
013050  122  101  124                .ASCII  /RAT/
013053  111  126  105                .ASCII  /IVE/
013056  040  050  122                .ASCII  / (R/
013061  104  065  061                .ASCII  /D51/
013064  040  127  122                .ASCII  / WR/
013067  111  124  105                .ASCII  /ITE/
013072  040  106  101                .ASCII  / FA/
013075  125  114  124                .ASCII  /ULT/
013100  000  000                    .ASCII  <00><00>
013102  045  101  125  P.AHD:        .ASCII  /%AU/
013105  116  111  124                .ASCII  /NIT/
013110  040  104  111                .ASCII  / DI/
```

```
013115  123  101  102                .ASCII  /SAB/
013116  114  105  104                .ASCII  /LED/
013121  040  102  131                .ASCII  / BY/
013124  040  106  111                .ASCII  / FI/
013127  105  114  104                .ASCII  /ELD/
013132  040  123  105                .ASCII  / SE/
013135  122  126  111                .ASCII  /RVI/
013140  103  105  040                .ASCII  /CE /
013143  117  122  040                .ASCII  /OR /
013146  111  116  124                .ASCII  /INT/
013151  105  122  116                .ASCII  /ERN/
013154  101  114  040                .ASCII  /AL /
013157  104  111  101                .ASCII  /DIA/
013162  107  116  117                .ASCII  /GNO/
013165  123  124  111                .ASCII  /STI/
013170  103  123  000                .ASCII  /CS/<00>
013173  000                        .ASCII  <00>
013174  045  101  042     P.AME:     .ASCII  /%A"/
013177  106  117  122                .ASCII  /FOR/
013202  103  105  104                .ASCII  /CED/
013205  040  105  122                .ASCII  / ER/
013210  122  117  122                .ASCII  /ROR/
013213  042  040  104                .ASCII  /" D/
013216  105  124  105                .ASCII  /ETE/
013221  103  124  105                .ASCII  /CTE/
013224  104  040  127                .ASCII  /D W/
013227  110  111  114                .ASCII  /HIL/
013232  105  040  101                .ASCII  /E A/
013235  103  103  105                .ASCII  /CCE/
013240  123  123  111                .ASCII  /SSI/
013243  116  107  040                .ASCII  /NG /
013246  106  103  124                .ASCII  /FCT/
013251  040  117  122                .ASCII  / OR/
013254  040  122  103                .ASCII  / RC/
013257  124  000  000                .ASCII  /T/<00><00>
013262  045  101  123     P.AMF:     .ASCII  /%AS/
013265  105  103  124                .ASCII  /ECT/
013270  117  122  040                .ASCII  /OR /
013273  127  122  111                .ASCII  /WRI/
013276  124  124  105                .ASCII  /TTE/
013301  116  040  127                .ASCII  /N W/
013304  111  124  110                .ASCII  /ITH/
013307  040  042  106                .ASCII  / "F/
013312  117  122  103                .ASCII  /ORC/
013315  105  104  040                .ASCII  /ED /
013320  105  122  122                .ASCII  /ERR/
013323  117  122  042                .ASCII  /OR'/
013326  040  115  117                .ASCII  / MO/
013331  104  111  106                .ASCII  /DIF/
013334  111  105  122                .ASCII  /IER/
013337  000                        .ASCII  <00>
013340  045  101  106     P.AMG:     .ASCII  /%AF/
013343  103  124  040                .ASCII  /CT /
013346  117  122  040                .ASCII  /OR /
013351  122  103  124                .ASCII  /RCT/
013354  040  125  116                .ASCII  / UN/
```

```
013535  122  105  101                .ASCII  /REA/
013542  104  101  102                .ASCII  /DAB/
013545  114  105  040                .ASCII  /LE /
013550  055  040  111                .ASCII  / I/
013553  116  126  101                .ASCII  /NVA/
013556  114  111  104                .ASCII  /LID/
013601  040  123  105                .ASCII  / SE/
013604  103  124  117                .ASCII  /CTO/
013607  122  040  110                .ASCII  /R H/
013612  105  101  104                .ASCII  /EAD/
013615  105  122  000                .ASCII  /ER/<00>
013620  045  101  110      P.AHH:    .ASCII  /%AH/
013623  105  101  104                .ASCII  /EAD/
013626  105  122  040                .ASCII  /ER /
013631  103  117  115                .ASCII  /COM/
013634  120  101  122                .ASCII  /PAR/
013637  105  040  105                .ASCII  /E E/
013642  122  122  117                .ASCII  /RRO/
013645  122  040  050                .ASCII  /R (/
013650  126  101  114                .ASCII  /VAL/
013653  111  104  040                .ASCII  /ID /
013656  110  105  101                .ASCII  /HEA/
013661  104  105  122                .ASCII  /DER/
013664  040  116  117                .ASCII  / NO/
013667  124  040  106                .ASCII  /T F/
013672  117  125  116                .ASCII  /OUN/
013675  104  051  000                .ASCII  /D)/<00>
013500  045  101  106      P.AHI:    .ASCII  /%AF/
013503  103  124  040                .ASCII  /CT /
013506  117  122  040                .ASCII  /OR /
013511  122  103  124                .ASCII  /RCT/
013514  040  125  116                .ASCII  / UN/
013517  122  105  101                .ASCII  /REA/
013522  104  101  102                .ASCII  /DAB/
013525  114  105  040                .ASCII  /LE /
013530  055  040  104                .ASCII  /- D/
013533  101  124  101                .ASCII  /ATA/
013536  040  123  131                .ASCII  / SY/
013541  116  103  040                .ASCII  /NC /
013544  124  111  115                .ASCII  /TIM/
013547  105  117  125                .ASCII  /EOU/
013552  124  000                     .ASCII  /T/<00>
013554  045  101  104      P.AHJ:    .ASCII  /%AD/
013557  101  124  101                .ASCII  /ATA/
013562  040  123  131                .ASCII  / SY/
013565  116  103  040                .ASCII  /NC /
013570  116  117  124                .ASCII  /NOT/
013573  040  106  117                .ASCII  / FO/
013576  125  116  104                .ASCII  /UND/
013601  040  050  104                .ASCII  / (D/
013604  101  124  101                .ASCII  /ATA/
013607  040  123  131                .ASCII  / SY/
013612  116  103  040                .ASCII  /NC /
013615  124  111  115                .ASCII  /TIM/
013620  105  117  125                .ASCII  /EOU/
013623  124  051  000                .ASCII  /T)/<00>
```

```
013624  045  101  106      P.AHK:  .ASCII  /%AF/
013631  103  124  040              .ASCII  /CT /
013634  117  122  040              .ASCII  /OR /
013637  122  103  124              .ASCII  /RCT/
013642  040  125  116              .ASCII  / UN/
013645  122  105  101              .ASCII  /REA/
013650  104  101  102              .ASCII  /DAB/
013653  114  105  040              .ASCII  /LE /
013656  055  040  125              .ASCII  / U/
013661  116  103  117              .ASCII  /NCO/
013664  122  122  105              .ASCII  /RRE/
013667  103  124  101              .ASCII  /CTA/
013672  102  114  105              .ASCII  /BLE/
013675  040  105  103              .ASCII  / EC/
013700  103  040  105              .ASCII  /C E/
013703  122  122  117              .ASCII  /RRO/
013706  122  000                   .ASCII  /R/<00>
013710  045  101  125      P.AHL:  .ASCII  /%AU/
013713  116  103  117              .ASCII  /NCO/
013716  122  122  105              .ASCII  /RRE/
013721  103  124  101              .ASCII  /CTA/
013724  102  114  105              .ASCII  /BLE/
013727  040  105  103              .ASCII  / EC/
013732  103  040  105              .ASCII  /C E/
013735  122  122  117              .ASCII  /RRO/
013740  122  000                   .ASCII  /R/<00>
013742  045  101  122      P.AHM:  .ASCII  /%AR/
013745  103  124  040              .ASCII  /CT /
013750  103  117  122              .ASCII  /COR/
013753  122  125  120              .ASCII  /RUP/
013756  124  105  104              .ASCII  /TED/
013761  000                       .ASCII  <00>
013762  045  101  116      P.AHN:  .ASCII  /%AN/
013765  117  040  122              .ASCII  /O R/
013770  105  120  114              .ASCII  /EPL/
013773  101  103  105              .ASCII  /ACE/
013776  115  105  116              .ASCII  /MEN/
014001  124  040  102              .ASCII  /T B/
014004  114  117  103              .ASCII  /LOC/
014007  113  040  101              .ASCII  /K A/
014012  126  101  111              .ASCII  /VAI/
014015  114  101  102              .ASCII  /LAB/
014020  114  105  040              .ASCII  /LE /
014023  050  122  103              .ASCII  /(RC/
014026  124  040  106              .ASCII  /T F/
014031  125  114  114              .ASCII  /ULL/
014034  051  000                   .ASCII  /)/<00>
014036  045  101  104      P.AHO:  .ASCII  /%AD/
014041  111  123  113              .ASCII  /ISK/
014044  040  116  117              .ASCII  / NO/
014047  124  040  106              .ASCII  /T F/
014052  117  122  115              .ASCII  /ORM/
014055  101  124  124              .ASCII  /ATT/
014060  105  104  040              .ASCII  /ED /
014063  127  111  124              .ASCII  /WIT/
014066  110  040  065              .ASCII  /H 5/
```

```
014071  061  062  040                .ASCII  /12 /
014074  102  131  124                .ASCII  /BYT/
014077  105  040  123                .ASCII  /E S/
014102  105  103  124                .ASCII  /ECT/
014105  117  122  123                .ASCII  /ORS/
014110  000  000                     .ASCII  <00><00>
014112  045  101  104    P.AHP:      .ASCII  /%AD/
014115  111  123  113                .ASCII  /ISK/
014120  040  116  117                .ASCII  / NO/
014123  124  040  106                .ASCII  /T F/
014126  117  122  115                .ASCII  /ORM/
014131  101  124  124                .ASCII  /ATT/
014134  105  104  040                .ASCII  /ED /
014137  117  122  040                .ASCII  /OR /
014142  106  103  124                .ASCII  /FCT/
014145  040  103  117                .ASCII  / CO/
014150  122  122  125                .ASCII  /RRU/
014153  120  124  105                .ASCII  /PTE/
014156  104  000                     .ASCII  /D/<00>
014160  045  101  117    P.AHQ:      .ASCII  /%AO/
014163  116  105  040                .ASCII  /NE /
014166  123  131  115                .ASCII  /SYM/
014171  102  117  114                .ASCII  /BOL/
014174  040  105  103                .ASCII  / EC/
014177  103  040  105                .ASCII  /C E/
014202  122  122  117                .ASCII  /RRO/
014205  122  000  000                .ASCII  /R/<00><00>
014210  045  101  124    P.AHR:      .ASCII  /%AT/
014213  127  117  040                .ASCII  /WO /
014216  123  131  115                .ASCII  /SYM/
014221  102  117  114                .ASCII  /BOL/
014224  040  105  103                .ASCII  / EC/
014227  103  040  105                .ASCII  /C E/
014232  122  122  117                .ASCII  /RRO/
014235  122  000  000                .ASCII  /R/<00><00>
014240  045  101  124    P.AHS:      .ASCII  /%AT/
014243  110  122  105                .ASCII  /HRE/
014246  105  040  123                .ASCII  /E S/
014251  131  115  102                .ASCII  /YMB/
014254  117  114  040                .ASCII  /OL /
014257  105  103  103                .ASCII  /ECC/
014262  040  105  122                .ASCII  / ER/
014265  122  117  122                .ASCII  /ROR/
014270  000  000                     .ASCII  <00><00>
014272  045  101  106    P.AHT:      .ASCII  /%AF/
014275  117  125  122                .ASCII  /OUR/
014300  040  123  131                .ASCII  / SY/
014303  115  102  117                .ASCII  /MBO/
014306  114  040  105                .ASCII  /L E/
014311  103  103  040                .ASCII  /CC /
014314  105  122  122                .ASCII  /ERR/
014317  117  122  000                .ASCII  /OR/<00>
014322  045  101  106    P.AHU:      .ASCII  /%AF/
014325  111  126  105                .ASCII  /IVE/
014330  040  123  131                .ASCII  / SY/
014333  115  102  117                .ASCII  /MBO/
```

```
014336  114  040  105            .ASCII  /L E/
014341  103  103  040            .ASCII  /CC /
014344  105  122  122            .ASCII  /ERR/
014347  117  122  000            .ASCII  /OR/<00>
014352  045  101  123  P.AHV:     .ASCII  /%AS/
014355  111  130  040            .ASCII  /IX /
014360  123  131  115            .ASCII  /SYM/
014363  102  117  114            .ASCII  /BOL/
014366  040  105  103            .ASCII  / EC/
014371  103  040  105            .ASCII  /C E/
014374  122  122  117            .ASCII  /RRO/
014377  122  000  000            .ASCII  /R/<00><00>
014402  045  101  123  P.AHW:     .ASCII  /%AS/
014405  105  126  105            .ASCII  /EVE/
014410  116  040  123            .ASCII  /N S/
014413  131  115  102            .ASCII  /YMB/
014416  117  114  040            .ASCII  /OL /
014421  105  103  103            .ASCII  /ECC/
014424  040  105  122            .ASCII  / ER/
014427  122  117  122            .ASCII  /ROR/
014432  000  000                 .ASCII  <00><00>
014434  045  101  105  P.AHX:     .ASCII  /%AE/
014437  111  107  110            .ASCII  /IGH/
014442  124  040  123            .ASCII  /T S/
014445  131  115  102            .ASCII  /YMB/
014450  117  114  040            .ASCII  /OL /
014453  105  103  103            .ASCII  /ECC/
014456  040  105  122            .ASCII  / ER/
014461  122  117  122            .ASCII  /ROR/
014464  000  000                 .ASCII  <00><00>
014466  045  101  103  P.AHY:     .ASCII  /%AC/
014471  117  122  122            .ASCII  /ORR/
014474  105  103  124            .ASCII  /ECT/
014477  101  102  114            .ASCII  /ABL/
014502  105  040  105            .ASCII  /E E/
014505  122  122  117            .ASCII  /RRO/
014510  122  040  111            .ASCII  /R I/
014513  116  040  105            .ASCII  /N E/
014516  103  103  040            .ASCII  /CC /
014521  106  111  105            .ASCII  /FIE/
014524  114  104  000            .ASCII  /LD/<00>
014527  000                      .ASCII  <00>
014530  045  101  125  P.AHZ:     .ASCII  /%AU/
014533  116  111  124            .ASCII  /NIT/
014536  040  123  117            .ASCII  / SO/
014541  106  124  127            .ASCII  /FTW/
014544  101  122  105            .ASCII  /ARE/
014547  040  127  122            .ASCII  / WR/
014552  111  124  105            .ASCII  /ITE/
014555  040  120  122            .ASCII  / PR/
014560  117  124  105            .ASCII  /OTE/
014563  103  124  105            .ASCII  /CTE/
014566  104  000                 .ASCII  /D/<00>
014570  045  101  125  P.AIA:     .ASCII  /%AU/
014573  116  111  124            .ASCII  /NIT/
014576  040  110  101            .ASCII  / HA/
```

```
014601  122     104     127             .ASCII  /RDW/
014604  101     122     105             .ASCII  /ARE/
014607  040     127     122             .ASCII  / WR/
014612  111     124     105             .ASCII  /ITE/
014615  040     120     122             .ASCII  / PR/
014620  117     124     105             .ASCII  /OTE/
014623  103     124     105             .ASCII  /CTE/
014626  104     000                     .ASCII  /D/<00>
014630  045     101     117     P.AIB:  .ASCII  /%AO/
014633  104     104     040             .ASCII  /DD /
014636  124     122     101             .ASCII  /TRA/
014641  116     123     106             .ASCII  /NSF/
014644  105     122     040             .ASCII  /ER /
014647  101     104     104             .ASCII  /ADD/
014652  122     105     123             .ASCII  /RES/
014655  123     000     000             .ASCII  /S/<00><00>
014660  045     101     117     P.AIC:  .ASCII  /%AO/
014663  104     104     040             .ASCII  /DD /
014666  102     131     124             .ASCII  /BYT/
014671  105     040     103             .ASCII  /E C/
014674  117     125     116             .ASCII  /OUN/
014677  124     000     000             .ASCII  /T/<00><00>
014702  045     101     116     P.AID:  .ASCII  /%AN/
014705  117     116     055             .ASCII  /ON-/
014710  105     130     111             .ASCII  /EXI/
014713  123     124     105             .ASCII  /STE/
014716  116     124     040             .ASCII  /NT /
014721  110     117     123             .ASCII  /HOS/
014724  124     040     115             .ASCII  /T M/
014727  105     115     117             .ASCII  /EMO/
014732  122     131     000             .ASCII  /RY/<00>
014735  000                             .ASCII  <00>
014736  045     101     110     P.AIE:  .ASCII  /%AH/
014741  117     123     124             .ASCII  /OST/
014744  040     115     105             .ASCII  / ME/
014747  115     117     122             .ASCII  /MOR/
014752  131     040     120             .ASCII  /Y P/
014755  101     122     111             .ASCII  /ARI/
014760  124     131     040             .ASCII  /TY /
014763  105     122     122             .ASCII  /ERR/
014766  117     122     000             .ASCII  /OR/<00>
014771  000                             .ASCII  <00>
014772  045     101     103     P.AIF:  .ASCII  /%AC/
014775  117     115     115             .ASCII  /OMM/
015000  101     116     104             .ASCII  /AND/
015003  040     124     111             .ASCII  / TI/
015006  115     117     125             .ASCII  /MOU/
015011  124     040     117             .ASCII  /T O/
015014  122     040     122             .ASCII  /R R/
015017  105     124     122             .ASCII  /ETR/
015022  131     040     114             .ASCII  /Y L/
015025  111     115     111             .ASCII  /IMI/
015030  124     040     105             .ASCII  /T E/
015033  130     103     105             .ASCII  /XCE/
015036  105     104     105             .ASCII  /EDE/
015041  104     000     000             .ASCII  /D/<00><00>
```

```
015044  045  101  123        P.AIG:  .ASCII  /%AS/
015047  105  122  111                .ASCII  /ERI/
015052  101  114  111                .ASCII  /ALI/
015055  132  105  122                .ASCII  /ZER/
015060  057  104  105                .ASCII  <57>/DE/
015063  123  105  122                .ASCII  /SER/
015066  111  101  114                .ASCII  /IAL/
015071  111  132  105                .ASCII  /IZE/
015074  122  040  117                .ASCII  /R O/
015077  126  105  122                .ASCII  /VER/
015102  122  125  116                .ASCII  /RUN/
015105  040  117  122                .ASCII  / OR/
015110  040  125  116                .ASCII  / UN/
015113  104  105  122                .ASCII  /DER/
015116  122  125  116                .ASCII  /RUN/
015121  000                        .ASCII  <00>
015122  045  101  105        P.AIH:  .ASCII  /%AE/
015125  104  103  040                .ASCII  /DC /
015130  105  122  122                .ASCII  /ERR/
015133  117  122  000                .ASCII  /OR/<00>
015136  045  101  111        P.AII:  .ASCII  /%AI/
015141  116  103  117                .ASCII  /NCO/
015144  116  123  111                .ASCII  /NSI/
015147  123  124  105                .ASCII  /STE/
015152  116  124  040                .ASCII  /NT /
015155  111  116  124                .ASCII  /INT/
015160  105  122  116                .ASCII  /ERN/
015163  101  114  040                .ASCII  /AL /
015166  104  101  124                .ASCII  /DAT/
015171  101  040  123                .ASCII  /A S/
015174  124  122  125                .ASCII  /TRU/
015177  103  124  125                .ASCII  /CTU/
015202  122  105  000                .ASCII  /RE/<00>
015205  000                        .ASCII  <00>
015206  045  101  104        P.AIJ:  .ASCII  /%AD/
015211  122  111  126                .ASCII  /RIV/
015214  105  040  103                .ASCII  /E C/
015217  117  115  115                .ASCII  /OMM/
015222  101  116  104                .ASCII  /AND/
015225  040  124  111                .ASCII  / TI/
015230  115  105  117                .ASCII  /MEO/
015233  125  124  040                .ASCII  /UT /
015236  050  116  117                .ASCII  /(NO/
015241  040  122  105                .ASCII  / RE/
015244  123  120  117                .ASCII  /SPO/
015247  116  123  105                .ASCII  /NSE/
015252  040  117  122                .ASCII  / OR/
015255  040  123  105                .ASCII  / SE/
015260  105  113  040                .ASCII  /EK /
015263  111  116  103                .ASCII  /INC/
015266  117  115  120                .ASCII  /OMP/
015271  114  105  124                .ASCII  /LET/
015274  105  051  000                .ASCII  /E)/<00>
015277  000                        .ASCII  <00>
015300  045  101  103        P.AIK:  .ASCII  /%AC/
015303  117  116  124                .ASCII  /ONT/
```

```
015304  122  117  114            .ASCII  /ROL/
015311  114  105  122            .ASCII  /LER/
015314  040  104  105            .ASCII  / DE/
015317  124  105  103            .ASCII  /TEC/
015322  124  105  104            .ASCII  /TED/
015325  040  124  122            .ASCII  / TR/
015330  101  116  123            .ASCII  /ANS/
015333  115  111  123            .ASCII  /MIS/
015336  123  111  117            .ASCII  /SIO/
015341  116  040  117            .ASCII  /N O/
015344  122  040  120            .ASCII  /R P/
015347  122  117  124            .ASCII  /ROT/
015352  117  103  117            .ASCII  /OCO/
015355  114  040  105            .ASCII  /L E/
015360  122  122  117            .ASCII  /RRO/
015363  122  000  000            .ASCII  /R/<00><00>
015366  045  101  120  P.AIL:    .ASCII  /%AP/
015371  117  123  111            .ASCII  /OSI/
015374  124  111  117            .ASCII  /TIO/
015377  116  040  105            .ASCII  /N E/
015402  122  122  117            .ASCII  /RRO/
015405  122  040  050            .ASCII  /R (/
015410  115  111  123            .ASCII  /MIS/
015413  055  123  105            .ASCII  /-SE/
015416  105  113  051            .ASCII  /EK)/
015421  000                      .ASCII  <00>
015422  045  101  114  P.AIM:    .ASCII  /%AL/
015425  117  123  124            .ASCII  /OST/
015430  040  122  105            .ASCII  / RE/
015433  101  104  057            .ASCII  /AD/<57>
015436  127  122  111            .ASCII  /WRI/
015441  124  105  040            .ASCII  /TE /
015444  122  105  101            .ASCII  /REA/
015447  104  131  040            .ASCII  /DY /
015452  104  125  122            .ASCII  /DUR/
015455  111  116  107            .ASCII  /ING/
015460  057  102  105            .ASCII  <57>/BE/
015463  124  127  105            .ASCII  /TWE/
015466  105  116  040            .ASCII  /EN /
015471  124  122  101            .ASCII  /TRA/
015474  116  123  106            .ASCII  /NSF/
015477  105  122  123            .ASCII  /ERS/
015502  000  000                  .ASCII  <00><00>
015504  045  101  104  P.AIN:    .ASCII  /%AD/
015507  122  111  126            .ASCII  /RIV/
015512  105  040  103            .ASCII  /E C/
015515  114  117  103            .ASCII  /LOC/
015520  113  040  104            .ASCII  /K D/
015523  122  117  120            .ASCII  /ROP/
015526  117  125  124            .ASCII  /OUT/
015531  000                      .ASCII  <00>
015532  045  101  114  P.AIO:    .ASCII  /%AL/
015535  117  123  124            .ASCII  /OST/
015540  040  122  105            .ASCII  / RE/
015543  103  105  111            .ASCII  /CEI/
015546  126  105  122            .ASCII  /VER/
```

```
015551  040  122  105            .ASCII  / RE/
015554  101  104  131            .ASCII  /ADY/
015557  040  102  105            .ASCII  / BE/
015562  124  127  105            .ASCII  /TWE/
015565  105  116  040            .ASCII  /EN /
015570  123  105  103            .ASCII  /SEC/
015573  124  117  122            .ASCII  /TOR/
015576  123  000                 .ASCII  /S/<00>
015600  045  101  104  P.AIP:    .ASCII  /%AD/
015603  122  111  126            .ASCII  /RIV/
015606  105  040  104            .ASCII  /E D/
015611  105  124  105            .ASCII  /ETE/
015614  103  124  105            .ASCII  /CTE/
015617  104  040  105            .ASCII  /D E/
015622  122  122  117            .ASCII  /RRO/
015625  122  000  000            .ASCII  /R/<00><00>
015630  045  101  103  P.AIQ:    .ASCII  /%AC/
015633  117  116  124            .ASCII  /ONT/
015636  122  117  114            .ASCII  /ROL/
015641  114  105  122            .ASCII  /LER/
015644  040  104  105            .ASCII  / DE/
015647  124  105  103            .ASCII  /TEC/
015652  124  105  104            .ASCII  /TED/
015655  040  120  125            .ASCII  / PU/
015660  114  123  105            .ASCII  /LSE/
015663  040  117  122            .ASCII  / OR/
015666  040  123  124            .ASCII  / ST/
015671  101  124  105            .ASCII  /ATE/
015674  040  120  101            .ASCII  / PA/
015677  122  111  124            .ASCII  /RIT/
015702  131  040  105            .ASCII  /Y E/
015705  122  122  117            .ASCII  /RRO/
015710  122  000                 .ASCII  /R/<00>
015712  045  116  045  P.AIR:    .ASCII  /%N%/
015715  101  040  174            .ASCII  /A /<174>
015720  040  102  141            .ASCII  / Ba/
015723  144  040  102            .ASCII  /d B/
015726  154  157  143            .ASCII  /loc/
015731  153  040  125            .ASCII  /k U/
015734  156  162  145            .ASCII  /nre/
015737  160  157  162            .ASCII  /por/
015742  164  145  144            .ASCII  /ted/
015745  000                      .ASCII  <00>
015746  045  116  045  P.AIS:    .ASCII  /%N%/
015751  101  040  174            .ASCII  /A /<174>
015754  040  105  162            .ASCII  / Er/
015757  162  157  162            .ASCII  /ror/
015762  040  114  157            .ASCII  / Lo/
015765  147  040  107            .ASCII  /g G/
015770  145  156  145            .ASCII  /ene/
015773  162  141  164            .ASCII  /rat/
015776  145  144  000            .ASCII  /ed/<00>
016001  000                      .ASCII  <00>
016002  045  116  045  P.AIT:    .ASCII  /%N%/
016005  101  040  174            .ASCII  /A /<174>
016010  040  123  145            .ASCII  / Se/
```

```
016013  162  151  157                .ASCII  /rio/
016016  165  163  040                .ASCII  /us /
016021  105  170  143                .ASCII  /Exc/
016024  145  160  164                .ASCII  /ept/
016027  151  157  156                .ASCII  /ion/
016032  000  000                     .ASCII  <00><00>
016034  045  116  045       P.AIU:   .ASCII  /%N%/
016037  101  174  040                .ASCII  /A/<174>/ /
016042  105  156  141                .ASCII  /Ena/
016045  142  154  145                .ASCII  /ble/
016050  040  101  164                .ASCII  / At/
016053  164  145  156                .ASCII  /ten/
016056  164  151  157                .ASCII  /tio/
016061  156  040  115                .ASCII  /n M/
016064  145  163  163                .ASCII  /ess/
016067  141  147  145                .ASCII  /age/
016072  163  000                     .ASCII  /s/<00>
016074  045  116  045       P.AIV:   .ASCII  /%N%/
016077  101  174  040                .ASCII  /A/<174>/ /
016102  105  156  141                .ASCII  /Ena/
016105  142  154  145                .ASCII  /ble/
016110  040  115  151                .ASCII  / Mi/
016113  163  143  145                .ASCII  /sce/
016116  154  154  141                .ASCII  /lla/
016121  156  145  157                .ASCII  /neo/
016124  165  163  040                .ASCII  /us /
016127  105  162  162                .ASCII  /Err/
016132  157  162  040                .ASCII  /or /
016135  114  157  147                .ASCII  /Log/
016140  040  115  145                .ASCII  / Me/
016143  163  163  141                .ASCII  /ssa/
016146  147  145  163                .ASCII  /ges/
016151  000                         .ASCII  <00>
016152  045  116  045       P.AIW:   .ASCII  /%N%/
016155  101  174  040                .ASCII  /A/<174>/ /
016160  105  156  141                .ASCII  /Ena/
016163  142  154  145                .ASCII  /ble/
016166  040  117  164                .ASCII  / Ot/
016171  150  145  162                .ASCII  /her/
016174  040  110  157                .ASCII  / Ho/
016177  163  164  163                .ASCII  /sts/
016202  040  105  162                .ASCII  / Er/
016205  162  157  162                .ASCII  /ror/
016210  040  114  157                .ASCII  / Lo/
016213  147  040  115                .ASCII  /g M/
016216  145  163  163                .ASCII  /ess/
016221  141  147  145                .ASCII  /age/
016224  163  000                     .ASCII  /s/<00>
016226  045  116  045       P.AIX:   .ASCII  /%N%/
016231  101  174  040                .ASCII  /A/<174>/ /
016234  105  156  141                .ASCII  /Ena/
016237  142  154  145                .ASCII  /ble/
016242  040  124  150                .ASCII  / Th/
016245  151  163  040                .ASCII  /is /
016250  110  157  163                .ASCII  /Hos/
016253  164  163  040                .ASCII  /ts /
```

```
016256  105  162  162          .ASCII  /Err/
016261  157  162  040          .ASCII  /or /
016264  114  157  147          .ASCII  /Log/
016267  040  115  145          .ASCII  / Me/
016272  163  163  141          .ASCII  /ssa/
016275  147  145  163          .ASCII  /ges/
016300  000  000               .ASCII  <00><00>
016302  045  116  045  P.AIY:   .ASCII  /%N%/
016305  101  174  040          .ASCII  /A/<174>/ /
016310  103  157  156          .ASCII  /Con/
016313  164  162  157          .ASCII  /tro/
016316  154  154  145          .ASCII  /lle/
016321  162  040  111          .ASCII  /r I/
016324  156  151  164          .ASCII  /nit/
016327  151  141  164          .ASCII  /iat/
016332  145  144  040          .ASCII  /ed /
016335  102  141  144          .ASCII  /Bad/
016340  040  102  154          .ASCII  / Bl/
016343  157  143  153          .ASCII  /ock/
016346  040  122  160          .ASCII  / Rp/
016351  154  143  155          .ASCII  /lcm/
016354  156  164  000          .ASCII  /nt/<00>
016357  000                    .ASCII  <00>
016360  045  116  045  P.AIZ:   .ASCII  /%N%/
016363  101  174  040          .ASCII  /A/<174>/ /
016366  123  150  141          .ASCII  /Sha/
016371  144  157  167          .ASCII  /dow/
016374  151  156  147          .ASCII  /ing/
016377  000                    .ASCII  <00>
016400  045  116  045  P.AJA:   .ASCII  /%N%/
016403  101  174  040          .ASCII  /A/<174>/ /
016406  065  067  066          .ASCII  /576/
016411  040  102  171          .ASCII  / By/
016414  164  145  040          .ASCII  /te /
016417  123  145  143          .ASCII  /Sec/
016422  164  157  162          .ASCII  /tor/
016425  163  000  000          .ASCII  /s/<00><00>
016430  045  116  045  P.AJB:   .ASCII  /%N%/
016433  101  174  040          .ASCII  /A/<174>/ /
016436  103  157  155          .ASCII  /Com/
016441  160  141  162          .ASCII  /par/
016444  145  040  122          .ASCII  /e R/
016447  145  141  144          .ASCII  /ead/
016452  163  000               .ASCII  /s/<00>
016454  045  116  045  P.AJC:   .ASCII  /%N%/
016457  101  174  040          .ASCII  /A/<174>/ /
016462  103  157  155          .ASCII  /Com/
016465  160  141  162          .ASCII  /par/
016470  145  040  127          .ASCII  /e W/
016473  162  151  164          .ASCII  /rit/
016476  145  163  000          .ASCII  /es/<00>
016501  000                    .ASCII  <00>
016502  045  116  045  P.AJD:   .ASCII  /%N%/
016505  101  174  040          .ASCII  /A/<174>/ /
016510  103  157  156          .ASCII  /Con/
016513  164  162  157          .ASCII  /tro/
```

```
016516  154  154  145              .ASCII  /lle/
016521  162  040  111              .ASCII  /r I/
016524  156  151  164              .ASCII  /nit/
016527  151  141  164              .ASCII  /iat/
016532  145  144  040              .ASCII  /ed /
016535  102  141  144              .ASCII  /Bad/
016540  040  102  154              .ASCII  / Bl/
016543  157  143  153              .ASCII  /ock/
016546  040  122  160              .ASCII  / Rp/
016551  154  143  155              .ASCII  /lcm/
016554  156  164  000              .ASCII  /nt/<00>
016557  000                       .ASCII  <00>
016560  045  116  045  P.AJE:     .ASCII  /%N%/
016563  101  174  040              .ASCII  /A/<174>/ /
016566  111  156  141              .ASCII  /Ina/
016571  143  164  151              .ASCII  /cti/
016574  166  145  040              .ASCII  /ve /
016577  123  150  141              .ASCII  /Sha/
016602  144  157  167              .ASCII  /dow/
016605  040  123  145              .ASCII  / Se/
016610  164  040  125              .ASCII  /t U/
016613  156  151  164              .ASCII  /nit/
016616  000  000                   .ASCII  <00><00>
016620  045  116  045  P.AJF:     .ASCII  /%N%/
016623  101  174  040              .ASCII  /A/<174>/ /
016626  122  145  155              .ASCII  /Rem/
016631  157  166  141              .ASCII  /ova/
016634  142  154  145              .ASCII  /ble/
016637  040  115  145              .ASCII  / Me/
016642  144  151  141              .ASCII  /dia/
016645  000                       .ASCII  <00>
016646  045  116  045  P.AJG:     .ASCII  /%N%/
016651  101  174  040              .ASCII  /A/<174>/ /
016654  123  165  160              .ASCII  /Sup/
016657  160  162  145              .ASCII  /pre/
016662  163  163  040              .ASCII  /ss /
016665  103  141  143              .ASCII  /Cac/
016670  150  151  156              .ASCII  /hin/
016673  147  040  050              .ASCII  /g (/
016676  150  151  147              .ASCII  /hig/
016701  150  040  163              .ASCII  /h s/
016704  160  145  145              .ASCII  /pee/
016707  144  051  000              .ASCII  /d)/<00>
016712  045  116  045  P.AJH:     .ASCII  /%N%/
016715  101  174  040              .ASCII  /A/<174>/ /
016720  123  165  160              .ASCII  /Sup/
016723  160  162  145              .ASCII  /pre/
016726  163  163  040              .ASCII  /ss /
016731  103  141  143              .ASCII  /Cac/
016734  150  151  156              .ASCII  /hin/
016737  147  040  050              .ASCII  /g (/
016742  154  157  167              .ASCII  /low/
016745  040  163  160              .ASCII  / sp/
016750  145  145  144              .ASCII  /eed/
016753  051  000  000              .ASCII  /)/<00><00>
016756  045  116  045  P.AJI:     .ASCII  /%N%/
```

```
016761  101  174  040            .ASCII  /A/<174>/ /
016764  127  162  151            .ASCII  /Wri/
016767  164  145  055            .ASCII  /te-/
016772  142  141  143            .ASCII  /bac/
016775  153  040  050            .ASCII  /k (/
017000  156  157  156            .ASCII  /non/
017003  055  166  157            .ASCII  /-vo/
017006  154  141  164            .ASCII  /lat/
017011  151  154  145            .ASCII  /ile/
017014  051  000                 .ASCII  /)/<00>
017016  045  116  045  P.AJJ:     .ASCII  /%N%/
017021  101  174  040            .ASCII  /A/<174>/ /
017024  127  162  151            .ASCII  /Wri/
017027  164  145  040            .ASCII  /te /
017032  120  162  157            .ASCII  /Pro/
017035  164  145  143            .ASCII  /tec/
017040  164  040  050            .ASCII  /t (/
017043  150  141  162            .ASCII  /har/
017046  144  167  141            .ASCII  /dwa/
017051  162  145  051            .ASCII  /re)/
017054  000  000                 .ASCII  <00><00>
017056  045  116  045  P.AJK:     .ASCII  /%N%/
017061  101  174  040            .ASCII  /A/<174>/ /
017064  127  162  151            .ASCII  /Wri/
017067  164  145  040            .ASCII  /te /
017072  120  162  157            .ASCII  /Pro/
017075  164  145  143            .ASCII  /tec/
017100  164  040  050            .ASCII  /t (/
017103  163  157  146            .ASCII  /sof/
017106  164  167  141            .ASCII  /twa/
017111  162  145  040            .ASCII  /re /
017114  157  162  040            .ASCII  /or /
017117  166  157  154            .ASCII  /vol/
017122  165  155  145            .ASCII  /ume/
017125  051  000  000            .ASCII  /)/<00><00>
017130  045  116  045  P.AJL:     .ASCII  /%N%/
017133  101  174  040            .ASCII  /A/<174>/ /
017136  065  067  066            .ASCII  /576/
017141  040  102  171            .ASCII  / By/
017144  164  145  040            .ASCII  /te /
017147  123  145  143            .ASCII  /Sec/
017152  164  157  162            .ASCII  /tor/
017155  163  000  000            .ASCII  /s/<00><00>
017160  045  101  040  P.AJM:     .ASCII  /%A /
017163  055  040  123            .ASCII  /- S/
017166  125  103  103            .ASCII  /UCC/
017171  105  123  123            .ASCII  /ESS/
017174  000  000                 .ASCII  <00><00>
017176  045  101  040  P.AJN:     .ASCII  /%A /
017201  055  040  111            .ASCII  /- I/
017204  116  126  101            .ASCII  /NVA/
017207  114  111  104            .ASCII  /LID/
017212  040  103  117            .ASCII  / CO/
017215  115  115  101            .ASCII  /MMA/
017220  116  104  050            .ASCII  /ND(/
017223  123  105  122            .ASCII  /SER/
```

```
017226  126  105  122                .ASCII  /VER/
017231  040  156  157                .ASCII  / no/
017234  156  111  104                .ASCII  /nID/
017237  114  105  040                .ASCII  /LE /
017242  157  162  040                .ASCII  /or /
017245  156  157  040                .ASCII  /no /
017250  155  145  144                .ASCII  /med/
017253  151  141  040                .ASCII  /ia /
017256  151  146  040                .ASCII  /if /
017261  105  130  055                .ASCII  /EX /
017264  114  103  055                .ASCII  /LC /
017267  120  122  107                .ASCII  /PRG/
017272  040  143  155                .ASCII  / cm/
017275  144  051  000                .ASCII  /d)/<00>
017300  045  101  040  P.AJO:        .ASCII  /%A /
017303  055  040  116                .ASCII  /- N/
017306  117  040  122                .ASCII  /O R/
017311  105  107  111                .ASCII  /EGI/
017314  117  116  040                .ASCII  /ON /
017317  101  126  101                .ASCII  /AVA/
017322  111  114  101                .ASCII  /ILA/
017325  102  114  105                .ASCII  /BLE/
017330  000  000                     .ASCII  <00><00>
017332  045  101  040  P.AJP:        .ASCII  /%A /
017335  055  040  116                .ASCII  /- N/
017340  117  040  122                .ASCII  /O R/
017343  105  107  111                .ASCII  /EGI/
017346  117  116  040                .ASCII  /ON /
017351  123  125  111                .ASCII  /SUI/
017354  124  101  102                .ASCII  /TAB/
017357  114  105  000                .ASCII  /LE/<00>
017362  045  101  040  P.AJQ:        .ASCII  /%A /
017365  055  040  120                .ASCII  /- P/
017370  122  117  107                .ASCII  /ROG/
017373  122  101  115                .ASCII  /RAM/
017376  040  116  117                .ASCII  / NO/
017401  124  040  113                .ASCII  /T K/
017404  116  117  127                .ASCII  /NOW/
017407  116  040  050                .ASCII  /N (/
017412  116  117  040                .ASCII  /NO /
017415  123  125  103                .ASCII  /SUC/
017420  110  040  120                .ASCII  /H P/
017423  122  117  107                .ASCII  /ROG/
017426  122  101  115                .ASCII  /RAM/
017431  040  117  116                .ASCII  / ON/
017434  040  115  105                .ASCII  / ME/
017437  104  111  101                .ASCII  /DIA/
017442  051  000                     .ASCII  /)/<00>
017444  045  101  040  P.AJR:        .ASCII  /%A /
017447  055  040  114                .ASCII  /- L/
017452  117  101  104                .ASCII  /OAD/
017455  040  106  101                .ASCII  / FA/
017460  111  114  125                .ASCII  /ILU/
017463  122  105  040                .ASCII  /RE /
017466  050  111  116                .ASCII  /(IN/
017471  120  125  124                .ASCII  /PUT/
```

```
017414  040  105  122          .ASCII  / ER/
017417  122  117  122          .ASCII  /ROR/
017502  040  127  110          .ASCII  / WH/
017505  111  114  105          .ASCII  /ILE/
017510  040  114  117          .ASCII  / LO/
017513  101  104  111          .ASCII  /ADI/
017516  116  107  040          .ASCII  /NG /
017521  120  122  117          .ASCII  /PRO/
017524  107  122  101          .ASCII  /GRA/
017527  115  051  000          .ASCII  /M)/<00>
017532  045  101  040  P.AJS:   .ASCII  /%A /
017535  055  040  123          .ASCII  /- S/
017540  124  101  116          .ASCII  /TAN/
017543  104  101  114          .ASCII  /DAL/
017546  117  116  105          .ASCII  /ONE/
017551  040  050  123          .ASCII  / (S/
017554  124  101  116          .ASCII  /TAN/
017557  104  101  114          .ASCII  /DAL/
017562  117  116  105          .ASCII  /ONE/
017565  040  115  117          .ASCII  / MO/
017570  104  111  106          .ASCII  /DIF/
017573  111  105  122          .ASCII  /IER/
017576  040  116  117          .ASCII  / NO/
017601  124  040  123          .ASCII  /T S/
017604  120  105  103          .ASCII  /PEC/
017607  111  106  111          .ASCII  /IFI/
017612  105  104  040          .ASCII  /ED /
017615  106  117  122          .ASCII  /FOR/
017620  040  123  124          .ASCII  / ST/
017623  101  116  104          .ASCII  /AND/
017626  040  101  114          .ASCII  / AL/
017631  117  116  105          .ASCII  /ONE/
017634  040  120  122          .ASCII  / PR/
017637  107  056  051          .ASCII  /G.)/
017642  000  000               .ASCII  <00><00>
017644  045  116  045  P.AJT:   .ASCII  /%N%/
017647  101  174  040          .ASCII  /A/<174>/ /
017652  117  156  145          .ASCII  /One/
017655  040  123  145          .ASCII  / Se/
017660  162  166  145          .ASCII  /rve/
017663  162  040  141          .ASCII  /r a/
017666  164  040  141          .ASCII  /t a/
017671  040  124  151          .ASCII  / Ti/
017674  155  145  000          .ASCII  /me/<00>
017677  000                    .ASCII  <00>
017700  045  116  045  P.AJU:   .ASCII  /%N%/
017703  101  174  040          .ASCII  /A/<174>/ /
017706  103  157  156          .ASCII  /Con/
017711  164  141  151          .ASCII  /tai/
017714  156  163  040          .ASCII  /ns /
017717  114  157  143          .ASCII  /Loc/
017722  141  154  040          .ASCII  /al /
017725  115  145  144          .ASCII  /Med/
017730  151  141  000          .ASCII  /ia/<00>
017733  000                    .ASCII  <00>
017734  045  116  045  P.AJV:   .ASCII  /%N%/
```

```
017737  101     174     040             .ASCII  /A/<174>/ /
017742  105     170     145             .ASCII  /Exe/
017745  143     165     164             .ASCII  /cut/
017750  145     040     114             .ASCII  /e L/
017753  157     143     141             .ASCII  /oca/
017756  154     040     120             .ASCII  /l P/
017761  162     147     040             .ASCII  /rg /
017764  143     155     144             .ASCII  /cmd/
017767  040     151     163             .ASCII  / is/
017772  040     125     116             .ASCII  / UN/
017775  123     125     120             .ASCII  /SUP/
020000  120     117     122             .ASCII  /POR/
020003  124     105     104             .ASCII  /TED/
020006  000     000                     .ASCII  <00><00>
020010  045     116     045     P.AJW:  .ASCII  /%N%/
020013  101     174     040             .ASCII  /A/<174>/ /
020016  103     165     162             .ASCII  /Cur/
020021  162     145     156             .ASCII  /ren/
020024  164     154     171             .ASCII  /tly/
020027  040     151     156             .ASCII  / in/
020032  040     101     143             .ASCII  / Ac/
020035  164     151     166             .ASCII  /tiv/
020040  145     040     123             .ASCII  /e S/
020043  164     141     164             .ASCII  /tat/
020046  145     000                     .ASCII  /e/<00>
020050  045     116     045     P.AJX:  .ASCII  /%N%/
020053  101     174     040             .ASCII  /A/<174>/ /
020056  123     164     141             .ASCII  /Sta/
020061  156     144     141             .ASCII  /nda/
020064  154     157     156             .ASCII  /lon/
020067  145     040     120             .ASCII  /e P/
020072  162     147     000             .ASCII  /rg/<00>
020075  000                             .ASCII  <00>
020076  045     116     045     P.AJY:  .ASCII  /%N%/
020101  101     174     040             .ASCII  /A/<174>/ /
020104  116     145     145             .ASCII  /Nee/
020107  144     163     040             .ASCII  /ds /
020112  157     166     145             .ASCII  /ove/
020115  162     154     141             .ASCII  /rla/
020120  171     000                     .ASCII  /y/<00>
020122  045     116     045     P.AJZ:  .ASCII  /%N%/
020125  101     174     040             .ASCII  /A/<174>/ /
020130  116     145     145             .ASCII  /Nee/
020133  144     163     040             .ASCII  /ds /
020136  127     162     151             .ASCII  /Wri/
020141  164     145     141             .ASCII  /tea/
020144  142     154     145             .ASCII  /ble/
020147  057     122     145             .ASCII  <57>/Re/
020152  141     144     141             .ASCII  /ada/
020155  142     154     145             .ASCII  /ble/
020160  040     117     166             .ASCII  / Ov/
020163  145     162     154             .ASCII  /erl/
020166  141     171     000             .ASCII  /ay/<00>
020171  000                             .ASCII  <00>
020172  045     116     045     P.AKA:  .ASCII  /%N%/
020175  101     174     040             .ASCII  /A/<174>/ /
```

```
020200  125  163  145            .ASCII  /Use/
020203  163  040  123            .ASCII  /s S/
020206  164  144  040            .ASCII  /td /
020211  104  165  160            .ASCII  /Dup/
020214  040  104  151            .ASCII  / Di/
020217  141  154  157            .ASCII  /alo/
020222  147  073  040            .ASCII  /g; /
020225  122  105  103            .ASCII  /REC/
020230  057  123  105            .ASCII  <57>/SE/
020233  116  104  057            .ASCII  /ND/<57>
020236  122  105  103            .ASCII  /REC/
020241  000                      .ASCII  <00>
020242  045  116  045  P.AKB:    .ASCII  /%N%/
020245  101  011  052            .ASCII  /A/<11>/*/
020250  052  040  121            .ASCII  /* Q/
020253  125  105  123            .ASCII  /UES/
020256  124  111  117            .ASCII  /TIO/
020261  116  000  000            .ASCII  /N/<00><00>
020264  045  116  045  P.AKC:    .ASCII  /%N%/
020267  101  011  052            .ASCII  /A/<11>/*/
020272  052  040  104            .ASCII  /* D/
020275  105  106  101            .ASCII  /EFA/
020300  125  114  124            .ASCII  /ULT/
020303  040  121  125            .ASCII  / QU/
020306  105  123  124            .ASCII  /EST/
020311  111  117  116            .ASCII  /ION/
020314  000  000                 .ASCII  <00><00>
020316  045  116  045  P.AKD:    .ASCII  /%N%/
020321  101  011  052            .ASCII  /A/<11>/*/
020324  052  040  111            .ASCII  /* I/
020327  116  106  117            .ASCII  /NFO/
020332  122  115  101            .ASCII  /RMA/
020335  124  111  117            .ASCII  /TIO/
020340  116  000                 .ASCII  /N/<00>
020342  045  116  045  P.AKE:    .ASCII  /%N%/
020345  101  011  052            .ASCII  /A/<11>/*/
020350  052  040  124            .ASCII  /* T/
020353  105  122  115            .ASCII  /ERM/
020356  111  116  101            .ASCII  /INA/
020361  124  111  117            .ASCII  /TIO/
020364  116  000                 .ASCII  /N/<00>
020366  045  116  045  P.AKF:    .ASCII  /%N%/
020371  101  011  052            .ASCII  /A/<11>/*/
020374  052  040  106            .ASCII  /* F/
020377  101  124  101            .ASCII  /ATA/
020402  114  040  105            .ASCII  /L E/
020405  122  122  117            .ASCII  /RRO/
020410  122  000                 .ASCII  /R/<00>
020412  045  116  045  P.AKG:    .ASCII  /%N%/
020415  101  011  052            .ASCII  /A/<11>/*/
020420  052  040  123            .ASCII  /* S/
020423  120  105  103            .ASCII  /PEC/
020426  111  101  114            .ASCII  /IAL/
020431  000                      .ASCII  <00>
020432  045  116  045  P.AKH:    .ASCII  /%N%/
020435  101  040  055            .ASCII  /A  /
```

```
020440  040  111  114            .ASCII  / IL/
020443  114  105  107            .ASCII  /LEG/
020446  101  114  040            .ASCII  /AL /
020451  125  116  111            .ASCII  /UNI/
020454  124  040  116            .ASCII  /T N/
020457  125  115  102            .ASCII  /UMB/
020462  105  122  000            .ASCII  /ER/<00>
020465  000                      .ASCII  <00>
020466  045  116  045  P.AKI:    .ASCII  /%N%/
020471  101  040  055            .ASCII  /A -/
020474  040  111  114            .ASCII  / IL/
020477  114  105  107            .ASCII  /LEG/
020502  101  114  040            .ASCII  /AL /
020505  120  110  131            .ASCII  /PHY/
020510  123  111  103            .ASCII  /SIC/
020513  101  114  040            .ASCII  /AL /
020516  117  122  040            .ASCII  /OR /
020521  122  105  114            .ASCII  /REL/
020524  101  124  111            .ASCII  /ATI/
020527  126  105  040            .ASCII  /VE /
020532  102  114  117            .ASCII  /BLO/
020535  103  113  040            .ASCII  /CK /
020540  116  125  115            .ASCII  /NUM/
020543  102  105  122            .ASCII  /BER/
020546  000  000                 .ASCII  <00><00>
020550  045  116  045  P.AKJ:    .ASCII  /%N%/
020553  101  040  055            .ASCII  /A -/
020556  040  104  105            .ASCII  / DE/
020561  126  111  103            .ASCII  /VIC/
020564  105  040  105            .ASCII  /E E/
020567  122  122  117            .ASCII  /RRO/
020572  122  000                 .ASCII  /R/<00>
020574  045  116  045  P.AKK:    .ASCII  /%N%/
020577  101  040  055            .ASCII  /A -/
020602  040  132  105            .ASCII  / ZE/
020605  122  117  040            .ASCII  /RO /
020610  114  105  116            .ASCII  /LEN/
020613  107  110  124            .ASCII  /GHT/
020616  040  115  105            .ASCII  / ME/
020621  123  123  101            .ASCII  /SSA/
020624  107  105  000            .ASCII  /GE/<00>
020627  000                      .ASCII  <00>
020630  045  116  045  P.AKL:    .ASCII  /%N%/
020633  101  040  055            .ASCII  /A -/
020636  055  040  101            .ASCII  /- A/
020641  123  103  111            .ASCII  /SCI/
020644  111  040  111            .ASCII  /I I/
020647  116  106  117            .ASCII  /NFO/
020652  122  115  101            .ASCII  /RMA/
020655  124  111  117            .ASCII  /TIO/
020660  116  040  000            .ASCII  /N /<00>
020663  000                      .ASCII  <00>
020664  045  116  045  P.AKM:    .ASCII  /%N%/
020667  101  040  055            .ASCII  /A -/
020672  055  040  116            .ASCII  /- N/
020675  117  116  055            .ASCII  /ON-/
```

```
020700  101  103  123            .ASCII  /ACS/
020703  111  111  040            .ASCII  /II /
020706  111  116  106            .ASCII  /INF/
020711  117  122  115            .ASCII  /ORM/
020714  101  124  111            .ASCII  /ATI/
020717  117  116  000            .ASCII  /ON/<00>
020722  045  116  045  P.AKN:    .ASCII  /%N%/
020725  101  040  055            .ASCII  /A -/
020730  055  040  124            .ASCII  /- T/
020733  105  122  115            .ASCII  /ERM/
020736  111  116  101            .ASCII  /INA/
020741  124  111  117            .ASCII  /TIO/
020744  116  040  115            .ASCII  /N M/
020747  105  123  123            .ASCII  /ESS/
020752  101  107  105            .ASCII  /AGE/
020755  000                      .ASCII  <00>
020756  045  116  045  P.AKO:    .ASCII  /%N%/
020761  101  040  055            .ASCII  /A -/
020764  055  040  123            .ASCII  /- S/
020767  125  103  103            .ASCII  /UCC/
020772  105  123  123            .ASCII  /ESS/
020775  057  106  101            .ASCII  <57>/FA/
021000  111  114  125            .ASCII  /ILU/
021003  122  105  040            .ASCII  /RE /
021006  103  117  104            .ASCII  /COD/
021011  105  000  000            .ASCII  /E/<00><00>
021014  045  116  045  P.AKP:    .ASCII  /%N%/
021017  101  040  055            .ASCII  /A -/
021022  055  040  123            .ASCII  /- S/
021025  105  116  104            .ASCII  /END/
021030  040  102  111            .ASCII  / BI/
021033  116  101  122            .ASCII  /NAR/
021036  131  040  104            .ASCII  /Y D/
021041  101  124  101            .ASCII  /ATA/
021044  000  000                 .ASCII  <00><00>
021046  045  116  045  P.AKQ:    .ASCII  /%N%/
021051  101  040  055            .ASCII  /A -/
021054  055  040  123            .ASCII  /- S/
021057  105  116  104            .ASCII  /END/
021062  040  125  116            .ASCII  / UN/
021065  111  124  040            .ASCII  /IT /
021070  116  125  115            .ASCII  /NUM/
021073  102  105  122            .ASCII  /BER/
021076  054  040  122            .ASCII  /, R/
021101  105  114  101            .ASCII  /ELA/
021104  124  111  126            .ASCII  /TIV/
021107  105  040  104            .ASCII  /E D/
021112  102  116  000            .ASCII  /BN/<00>
021115  000                      .ASCII  <00>
021116  045  116  045  P.AKR:    .ASCII  /%N%/
021121  101  040  055            .ASCII  /A -/
021124  055  040  123            .ASCII  /- S/
021127  105  116  104            .ASCII  /END/
021132  040  125  116            .ASCII  / UN/
021135  111  124  040            .ASCII  /IT /
021140  116  125  115            .ASCII  /NUM/
```

```
021143  102  105  122            .ASCII  /BER/
021146  054  040  122            .ASCII  /, R/
021151  105  114  101            .ASCII  /ELA/
021154  124  111  126            .ASCII  /TIV/
021157  105  040  104            .ASCII  /E D/
021162  102  116  054            .ASCII  /BN,/
021165  040  127  122            .ASCII  / WR/
021170  111  124  105            .ASCII  /ITE/
021173  040  120  101            .ASCII  / PA/
021176  124  124  105            .ASCII  /TTE/
021201  122  116  000            .ASCII  /RN/<00>
021204  045  116  045  P.AKS:     .ASCII  /%N%/
021207  101  040  055            .ASCII  /A -/
021212  055  040  123            .ASCII  /- S/
021215  105  116  104            .ASCII  /END/
021220  040  125  116            .ASCII  / UN/
021223  111  124  040            .ASCII  /IT /
021226  116  125  115            .ASCII  /NUM/
021231  102  105  122            .ASCII  /BER/
021234  054  040  120            .ASCII  /, P/
021237  110  131  123            .ASCII  /HYS/
021242  111  103  101            .ASCII  /ICA/
021245  114  040  102            .ASCII  /L B/
021250  114  117  103            .ASCII  /LOC/
021253  113  040  116            .ASCII  /K N/
021256  125  115  102            .ASCII  /UMB/
021261  105  122  000            .ASCII  /ER/<00>
021264  045  116  045  P.AKT:     .ASCII  /%N%/
021267  101  040  055            .ASCII  /A -/
021272  055  040  123            .ASCII  /- S/
021275  105  116  104            .ASCII  /END/
021300  040  125  116            .ASCII  / UN/
021303  111  124  040            .ASCII  /IT /
021306  116  125  115            .ASCII  /NUM/
021311  102  105  122            .ASCII  /BER/
021314  054  040  114            .ASCII  /, L/
021317  117  107  111            .ASCII  /OGI/
021322  103  101  114            .ASCII  /CAL/
021325  040  040  102            .ASCII  /  B/
021330  114  117  103            .ASCII  /LOC/
021333  113  040  116            .ASCII  /K N/
021336  125  115  102            .ASCII  /UMB/
021341  105  122  040            .ASCII  /ER /
021344  000  000                 .ASCII  <00><00>
021346  045  101  103  P.AKV:     .ASCII  /%AC/
021351  117  116  124            .ASCII  /ONT/
021354  122  117  114            .ASCII  /ROL/
021357  114  105  122            .ASCII  /LER/
021362  040  124  111            .ASCII  / TI/
021365  115  105  117            .ASCII  /MEO/
021370  125  124  000            .ASCII  /UT/<00>
021373  000                      .ASCII  <00>
021374  045  101  105  P.AKW:     .ASCII  /%AE/
021377  116  126  105            .ASCII  /NVE/
021402  114  117  120            .ASCII  /LOP/
021405  105  057  120            .ASCII  /E/<57>/P/
```

```
021410  101  103  113            .ASCII  /ACK/
021413  105  124  040            .ASCII  /ET /
021416  122  105  101            .ASCII  /REA/
021421  104  040  105            .ASCII  /D E/
021424  122  122  117            .ASCII  /RRO/
021427  122  040  050            .ASCII  /R (/
021432  120  101  122            .ASCII  /PAR/
021435  111  124  131            .ASCII  /ITY/
021440  040  117  122            .ASCII  / OR/
021443  040  124  111            .ASCII  / TI/
021446  115  105  117            .ASCII  /MEO/
021451  125  124  051            .ASCII  /UT)/
021454  000  000                 .ASCII  <00><00>
021456  045  101  105  P.AKX:    .ASCII  /%AE/
021461  116  126  105            .ASCII  /NVE/
021464  114  117  120            .ASCII  /LOP/
021467  105  057  120            .ASCII  /E/<57>/P/
021472  101  103  113            .ASCII  /ACK/
021475  105  124  040            .ASCII  /ET /
021500  127  122  111            .ASCII  /WRI/
021503  124  105  040            .ASCII  /TE /
021506  105  122  122            .ASCII  /ERR/
021511  117  122  040            .ASCII  /OR /
021514  050  120  101            .ASCII  /(PA/
021517  122  111  124            .ASCII  /RIT/
021522  131  040  117            .ASCII  /Y O/
021525  122  040  124            .ASCII  /R T/
021530  111  115  105            .ASCII  /IME/
021533  117  125  124            .ASCII  /OUT/
021536  051  000                 .ASCII  /)/<00>
021540  045  101  103  P.AKY:    .ASCII  /%AC/
021543  117  116  124            .ASCII  /ONT/
021546  122  117  114            .ASCII  /ROL/
021551  114  105  122            .ASCII  /LER/
021554  040  122  117            .ASCII  / RO/
021557  115  040  101            .ASCII  /M A/
021562  116  104  040            .ASCII  /ND /
021565  122  101  115            .ASCII  /RAM/
021570  040  120  101            .ASCII  / PA/
021573  122  111  124            .ASCII  /RIT/
021576  131  040  105            .ASCII  /Y E/
021601  122  122  117            .ASCII  /RRO/
021604  122  000                 .ASCII  /R/<00>
021606  045  101  103  P.AKZ:    .ASCII  /%AC/
021611  117  116  124            .ASCII  /ONT/
021614  122  117  114            .ASCII  /ROL/
021617  114  105  122            .ASCII  /LER/
021622  040  122  101            .ASCII  / RA/
021625  115  040  120            .ASCII  /M P/
021630  101  122  111            .ASCII  /ARI/
021633  124  131  040            .ASCII  /TY /
021636  105  122  122            .ASCII  /ERR/
021641  117  122  000            .ASCII  /OR/<00>
021644  045  101  103  P.ALA:    .ASCII  /%AC/
021647  117  116  124            .ASCII  /ONT/
021652  122  117  114            .ASCII  /ROL/
```

```
021655  114  105  122            .ASCII  /LER/
021660  040  122  117            .ASCII  / RO/
021663  115  040  120            .ASCII  /M P/
021666  101  122  111            .ASCII  /ARI/
021671  124  131  040            .ASCII  /TY /
021674  105  122  122            .ASCII  /ERR/
021677  117  122  000            .ASCII  /OR/<00>
021702  045  101  122  P.ALB:    .ASCII  /%AR/
021705  111  116  107            .ASCII  /ING/
021710  040  122  105            .ASCII  / RE/
021713  101  104  040            .ASCII  /AD /
021716  105  122  122            .ASCII  /ERR/
021721  117  122  040            .ASCII  /OR /
021724  050  120  101            .ASCII  /(PA/
021727  122  111  124            .ASCII  /RIT/
021732  131  040  117            .ASCII  /Y O/
021735  122  040  124            .ASCII  /R T/
021740  111  115  105            .ASCII  /IME/
021743  117  125  124            .ASCII  /OUT/
021746  051  000                 .ASCII  /)/<00>
021750  045  101  122  P.ALC:    .ASCII  /%AR/
021753  111  116  107            .ASCII  /ING/
021756  040  127  122            .ASCII  / WR/
021761  111  124  105            .ASCII  /ITE/
021764  040  105  122            .ASCII  / ER/
021767  122  117  122            .ASCII  /ROR/
021772  040  050  120            .ASCII  / (P/
021775  101  122  111            .ASCII  /ARI/
022000  124  131  040            .ASCII  /TY /
022003  117  122  040            .ASCII  /OR /
022006  124  111  115            .ASCII  /TIM/
022011  105  117  125            .ASCII  /EOU/
022014  124  051  000            .ASCII  /T)/<00>
022017  000                      .ASCII  <00>
022020  111  116  124  P.ALD:    .ASCII  /INT/
022023  105  122  122            .ASCII  /ERR/
022026  125  120  124            .ASCII  /UPT/
022031  040  115  101            .ASCII  / MA/
022034  123  124  105            .ASCII  /STE/
022037  122  040  106            .ASCII  /R F/
022042  101  111  114            .ASCII  /AIL/
022045  125  122  105            .ASCII  /URE/
022050  000  000                 .ASCII  <00><00>
022052  045  101  110  P.ALE:    .ASCII  /%AH/
022055  117  123  124            .ASCII  /OST/
022060  040  101  103            .ASCII  / AC/
022063  103  105  123            .ASCII  /CES/
022066  123  040  124            .ASCII  /S T/
022071  111  115  105            .ASCII  /IME/
022074  117  125  124            .ASCII  /OUT/
022077  040  050  110            .ASCII  / (H/
022102  111  107  110            .ASCII  /IGH/
022105  105  122  040            .ASCII  /ER /
022110  114  105  126            .ASCII  /LEV/
022113  105  114  040            .ASCII  /EL /
022116  120  122  117            .ASCII  /PRO/
```

```
022121  124  117  103            .ASCII  /TOC/
022124  117  114  040            .ASCII  /OL /
022127  104  105  120            .ASCII  /DEP/
022132  105  116  104            .ASCII  /END/
022135  105  116  124            .ASCII  /ENT/
022140  051  000                 .ASCII  /)/<00>
022142  045  101  103  P.ALF:    .ASCII  /%AC/
022145  122  105  104            .ASCII  /RED/
022150  111  124  040            .ASCII  /IT /
022153  114  111  115            .ASCII  /LIM/
022156  111  124  040            .ASCII  /IT /
022161  105  130  103            .ASCII  /EXC/
022164  105  105  104            .ASCII  /EED/
022167  105  104  000            .ASCII  /ED/<00>
022172  045  101  121  P.ALG:    .ASCII  /%AQ/
022175  055  102  125            .ASCII  /-BU/
022200  123  040  115            .ASCII  /S M/
022203  101  123  124            .ASCII  /AST/
022206  105  122  040            .ASCII  /ER /
022211  105  122  122            .ASCII  /ERR/
022214  117  122  000            .ASCII  /OR/<00>
022217  000                      .ASCII  <00>
022220  045  101  103  P.ALH:    .ASCII  /%AC/
022223  117  116  124            .ASCII  /ONT/
022226  122  117  114            .ASCII  /ROL/
022231  114  105  122            .ASCII  /LER/
022234  040  106  101            .ASCII  / FA/
022237  124  101  114            .ASCII  /TAL/
022242  040  105  122            .ASCII  / ER/
022245  122  117  122            .ASCII  /ROR/
022250  000  000                 .ASCII  <00><00>
022252  045  101  111  P.ALI:    .ASCII  /%AI/
022255  116  123  124            .ASCII  /NST/
022260  122  125  103            .ASCII  /RUC/
022263  124  111  117            .ASCII  /TIO/
022266  116  040  114            .ASCII  /N L/
022271  117  117  120            .ASCII  /OOP/
022274  040  124  111            .ASCII  / TI/
022277  115  105  117            .ASCII  /MEO/
022302  125  124  000            .ASCII  /UT/<00>
022305  000                      .ASCII  <00>
022306  045  101  111  P.ALJ:    .ASCII  /%AI/
022311  114  114  105            .ASCII  /LLE/
022314  107  101  114            .ASCII  /GAL/
022317  040  126  111            .ASCII  / VI/
022322  122  124  125            .ASCII  /RTU/
022325  101  114  040            .ASCII  /AL /
022330  103  111  122            .ASCII  /CIR/
022333  103  125  111            .ASCII  /CUI/
022336  124  040  111            .ASCII  /T I/
022341  104  000  000            .ASCII  /D/<00><00>
022344  045  101  111  P.ALK:    .ASCII  /%AI/
022347  116  124  105            .ASCII  /NTE/
022352  122  122  125            .ASCII  /RRU/
022355  120  124  040            .ASCII  /PT /
022360  126  105  103            .ASCII  /VEC/
```

```
022363  124  117  122                .ASCII  /TOR/
022366  040  111  114                .ASCII  / IL/
022371  114  105  107                .ASCII  /LEG/
022374  101  114  000                .ASCII  /AL/<00>
022377  000                         .ASCII  <00>
022400  045  101  115        P.ALL:  .ASCII  /%AM/
022403  101  111  116                .ASCII  /AIN/
022406  124  105  116                .ASCII  /TEN/
022411  101  116  103                .ASCII  /ANC/
022414  105  040  122                .ASCII  /E R/
022417  105  101  104                .ASCII  /EAD/
022422  057  127  122                .ASCII  <57>/WR/
022425  111  124  105                .ASCII  /ITE/
022430  040  111  116                .ASCII  / IN/
022433  126  101  114                .ASCII  /VAL/
022436  111  104  040                .ASCII  /ID /
022441  122  105  107                .ASCII  /REG/
022444  111  117  116                .ASCII  /ION/
022447  040  111  104                .ASCII  / ID/
022452  105  116  124                .ASCII  /ENT/
022455  111  106  111                .ASCII  /IFI/
022460  105  122  000                .ASCII  /ER/<00>
022463  000                         .ASCII  <00>
022464  045  101  115        P.ALM:  .ASCII  /%AM/
022467  101  111  116                .ASCII  /AIN/
022472  124  105  116                .ASCII  /TEN/
022475  101  116  103                .ASCII  /ANC/
022500  105  040  127                .ASCII  /E W/
022503  122  111  124                .ASCII  /RIT/
022506  105  040  114                .ASCII  /E L/
022511  117  101  104                .ASCII  /OAD/
022514  040  124  117                .ASCII  / TO/
022517  040  116  117                .ASCII  / NO/
022522  116  055  114                .ASCII  /N-L/
022525  117  101  104                .ASCII  /OAD/
022530  101  102  114                .ASCII  /ABL/
022533  105  040  103                .ASCII  /E C/
022536  117  116  124                .ASCII  /ONT/
022541  122  117  114                .ASCII  /ROL/
022544  114  105  122                .ASCII  /LER/
022547  000                         .ASCII  <00>
022550  045  101  103        P.ALN:  .ASCII  /%AC/
022553  117  116  124                .ASCII  /ONT/
022556  122  117  114                .ASCII  /ROL/
022561  114  105  122                .ASCII  /LER/
022564  040  122  101                .ASCII  / RA/
022567  115  040  105                .ASCII  /M E/
022572  122  122  117                .ASCII  /RRO/
022575  122  040  050                .ASCII  /R (/
022600  116  117  116                .ASCII  /NON/
022603  055  120  101                .ASCII  /-PA/
022606  122  111  124                .ASCII  /RIT/
022611  131  051  000                .ASCII  /Y)/<00>
022614  045  101  111        P.ALO:  .ASCII  /%AI/
022617  116  111  124                .ASCII  /NIT/
022622  040  123  105                .ASCII  / SE/
```

```
022625  121     125     105             .ASCII  /QUE/
022630  116     103     105             .ASCII  /NCE/
022633  040     105     122             .ASCII  / ER/
022636  122     117     122             .ASCII  /ROR/
022641  000                             .ASCII  <00>
022642  045     101     110     P.ALP:  .ASCII  /%AH/
022645  111     107     110             .ASCII  /IGH/
022650  105     122     040             .ASCII  /ER /
022653  114     105     126             .ASCII  /LEV/
022656  105     114     040             .ASCII  /EL /
022661  120     122     117             .ASCII  /PRO/
022664  124     117     103             .ASCII  /TOC/
022667  117     114     040             .ASCII  /OL /
022672  111     116     103             .ASCII  /INC/
022675  117     115     120             .ASCII  /OMP/
022700  101     124     111             .ASCII  /ATI/
022703  102     111     114             .ASCII  /BIL/
022706  111     124     131             .ASCII  /ITY/
022711  040     105     122             .ASCII  / ER/
022714  122     117     122             .ASCII  /ROR/
022717  000                             .ASCII  <00>
022720  045     101     120     P.ALQ:  .ASCII  /%AP/
022723  125     122     107             .ASCII  /URG/
022726  105     057     120             .ASCII  /E/<57>/P/
022731  117     114     114             .ASCII  /OLL/
022734  040     110     101             .ASCII  / HA/
022737  122     104     127             .ASCII  /RDW/
022742  101     122     105             .ASCII  /ARE/
022745  040     106     101             .ASCII  / FA/
022750  111     114     125             .ASCII  /ILU/
022753  122     105     000             .ASCII  /RE/<00>
022756  045     101     115     P.ALR:  .ASCII  /%AM/
022761  101     120     120             .ASCII  /APP/
022764  111     116     107             .ASCII  /ING/
022767  040     122     105             .ASCII  / RE/
022772  107     111     123             .ASCII  /GIS/
022775  124     105     122             .ASCII  /TER/
023000  040     122     105             .ASCII  / RE/
023003  101     104     040             .ASCII  /AD /
023006  106     101     111             .ASCII  /FAI/
023011  114     125     122             .ASCII  /LUR/
023014  105     040     050             .ASCII  /E (/
023017  120     101     122             .ASCII  /PAR/
023022  111     124     131             .ASCII  /ITY/
023025  040     117     122             .ASCII  / OR/
023030  040     124     111             .ASCII  / TI/
023033  115     105     117             .ASCII  /MEO/
023036  125     124     051             .ASCII  /UT)/
023041  000                             .ASCII  <00>
023042  021346'                 P.AKU:  .WORD   P.AKV
023044  021374'                         .WORD   P.AKW
023046  021456'                         .WORD   P.AKX
023050  021540'                         .WORD   P.AKY
023052  021606'                         .WORD   P.AKZ
023054  021644'                         .WORD   P.ALA
023056  021702'                         .WORD   P.ALB
```

```
023060  021750                          .WORD   P.ALC
023062  022020                          .WORD   P.ALD
023064  022052'                         .WORD   P.ALE
023066  022142'                         .WORD   P.ALF
023070  022172                          .WORD   P.ALG
023072  022220'                         .WORD   P.ALH
023074  022252'                         .WORD   P.ALI
023076  022306'                         .WORD   P.ALJ
023100  022344'                         .WORD   P.ALK
023102  022400                          .WORD   P.ALL
023104  022464'                         .WORD   P.ALM
023106  022550'                         .WORD   P.ALN
023110  022614'                         .WORD   P.ALO
023112  022642'                         .WORD   P.ALP
023114  022720'                         .WORD   P.ALQ
023116  022756'                         .WORD   P.ALR
023120     045     101     124  P.ALT:  .ASCII  /#AT/
023123     061     061     040          .ASCII  /11 /
023126     103     120     125          .ASCII  /CPU/
023131     040     106     101          .ASCII  / FA/
023134     111     114     125          .ASCII  /ILU/
023137     122     105     000          .ASCII  /RE/<00>
023142     045     101     116  P.ALU:  .ASCII  /#AN/
023145     117     116     055          .ASCII  /ON-/
023150     120     101     122          .ASCII  /PAR/
023153     111     124     131          .ASCII  /ITY/
023156     040     122     101          .ASCII  / RA/
023161     115     040     105          .ASCII  /M E/
023164     122     122     117          .ASCII  /RRO/
023167     122     000     000          .ASCII  /R/<00><00>
023172     045     101     123  P.ALV:  .ASCII  /#AS/
023175     124     101     124          .ASCII  /TAT/
023200     105     040     115          .ASCII  /E M/
023203     101     103     110          .ASCII  /ACH/
023206     111     116     105          .ASCII  /INE/
023211     040     106     101          .ASCII  / FA/
023214     111     114     125          .ASCII  /ILU/
023217     122     105     040          .ASCII  /RE /
023222     055     040     124          .ASCII  /- T/
023225     061     061     040          .ASCII  /11 /
023230     101     104     104          .ASCII  /ADD/
023233     122     105     123          .ASCII  /RES/
023236     123     040     122          .ASCII  /S R/
023241     105     107     111          .ASCII  /EGI/
023244     123     124     105          .ASCII  /STE/
023247     122     000     000          .ASCII  /R/<00><00>
023252     045     101     123  P.ALW:  .ASCII  /#AS/
023255     124     101     124          .ASCII  /TAT/
023260     105     040     115          .ASCII  /E M/
023263     101     103     110          .ASCII  /ACH/
023266     111     116     105          .ASCII  /INE/
023271     040     106     101          .ASCII  / FA/
023274     111     114     125          .ASCII  /ILU/
023277     122     105     040          .ASCII  /RE /
023302     055     040     121          .ASCII  /- Q/
023305     055     102     125          .ASCII  /-BU/
```

```
023310  123  040  101            .ASCII  /S A/
023313  104  104  122            .ASCII  /DDR/
023316  105  123  123            .ASCII  /ESS/
023321  040  122  105            .ASCII  / RE/
023324  107  111  123            .ASCII  /GIS/
023327  124  105  122            .ASCII  /TER/
023332  000  000                 .ASCII  <00><00>
023334  045  101  123  P.ALX:    .ASCII  /%AS/
023337  124  101  124            .ASCII  /TAT/
023342  105  040  115            .ASCII  /E M/
023345  101  103  110            .ASCII  /ACH/
023350  111  116  105            .ASCII  /INE/
023353  040  106  101            .ASCII  / FA/
023356  111  114  125            .ASCII  /ILU/
023361  122  105  040            .ASCII  /RE /
023364  055  040  103            .ASCII  /- C/
023367  122  103  040            .ASCII  /RC /
023372  122  105  107            .ASCII  /REG/
023375  111  123  124            .ASCII  /IST/
023400  105  122  000            .ASCII  /ER/<00>
023403  000                      .ASCII  <00>
023404  045  101  123  P.ALY:    .ASCII  /%AS/
023407  124  101  124            .ASCII  /TAT/
023412  105  040  115            .ASCII  /E M/
023415  101  103  110            .ASCII  /ACH/
023420  111  116  105            .ASCII  /INE/
023423  040  106  101            .ASCII  / FA/
023426  111  114  125            .ASCII  /ILU/
023431  122  105  040            .ASCII  /RE /
023434  055  040  123            .ASCII  /- S/
023437  105  122  111            .ASCII  /ERI/
023442  101  114  111            .ASCII  /ALI/
023445  132  105  122            .ASCII  /ZER/
023450  057  104  105            .ASCII  <57>/DE/
023453  123  105  122            .ASCII  /SER/
023456  111  101  114            .ASCII  /IAL/
023461  111  132  105            .ASCII  /IZE/
023464  122  040  122            .ASCII  /R R/
023467  105  107  111            .ASCII  /EGI/
023472  123  124  105            .ASCII  /STE/
023475  122  000  000            .ASCII  /R/<00><00>
023500  045  101  123  P.ALZ:    .ASCII  /%AS/
023503  124  101  124            .ASCII  /TAT/
023506  105  040  115            .ASCII  /E M/
023511  101  103  110            .ASCII  /ACH/
023514  111  116  105            .ASCII  /INE/
023517  040  106  101            .ASCII  / FA/
023522  111  114  125            .ASCII  /ILU/
023525  122  105  040            .ASCII  /RE /
023530  055  040  127            .ASCII  /- W/
023533  122  117  116            .ASCII  /RON/
023536  107  040  110            .ASCII  /G H/
023541  101  122  104            .ASCII  /ARD/
023544  127  101  122            .ASCII  /WAR/
023547  105  040  126            .ASCII  /E V/
023552  105  122  123            .ASCII  /ERS/
```

```
023555    111     117     116                    .ASCII  /ION/
023560    000     000                            .ASCII  <00><00>
023562  023120                          P.ALS:    .WORD   P.ALT
023564  023142                                    .WORD   P.ALU
023566  023172                                    .WORD   P.ALV
023570  023252                                    .WORD   P.ALW
023572  023334                                    .WORD   P.ALX
023574  023404                                    .WORD   P.ALY
023576  023500                                    .WORD   P.ALZ
023600    045     101     040           P.AMA:    .ASCII  /%A /
023603    045     117     066                    .ASCII  /%06/
023606    000     000                            .ASCII  <00><00>
023610    045     101     157           P.AMB:    .ASCII  /%Ao/
023613    143     164     040                    .ASCII  /ct /
023616    045     117     064                    .ASCII  /%04/
023621    000                                    .ASCII  <0>
023622    045     125     064           P.AMC:    .ASCII  /%S4/
023625    000                                    .ASCII  <00>
023626    045     116     000           P.AMD:    .ASCII  /%N/<00>
023631    000                                    .ASCII  <00>
023632    045     101     040           P.AME:    .ASCII  /%A /
023635    055     040     000                    .ASCII  /- /<00>
023640    045     101     052           P.AMF:    .ASCII  /%A*/
023643    040     000     000                    .ASCII  / /<00><00>
023646  000000C                         L$HWLEN::
                                                  .WORD   <<L$NDHW-L$HWLEN>/2>
023650  176150                          HWPT.IP.ADDR::
                                                  .WORD   1630
023652  000154                          HWPT.VECTOR::
                                                  .WORD   154
023654  000004                          HWPT.BR.LEVEL::
                                                  .WORD   4
023656  100034                          HWPT.DISK::
                                                  .WORD   77744
023660  000000                          HWPT.S.TRK::
                                                  .WORD   0
023662  052137                          HWPT.E.TRK::
                                                  .WORD   52137
023664                                  L$NDHW::  .BLKW   1
023666  000000C                         L$SWLEN::
                                                  .WORD   <<L$NDSW-L$SWLEN>/2>
023670  000040                          SWP.ERROR::
                                                  .WORD   40
023672  000024                          SWP.XFER::
                                                  .WORD   24
023674  000202                          SWP.FLAGS::
                                                  .WORD   202
023676  000000                          SWP.DPAT::
                                                  .WORD   0
023700  000020                          SWP.UCNT::
                                                  .WORD   20
023702  000143                          SWP.RAT::
                                                  .WORD   143
023704  000022                          DUPROUND::
                                                  .WORD   22
023706                                  SWP.UDPAT::
```

```
023746                                  .BLKW   20
023750                          L$NDSW::.BLKW   1
023750  000000                  L$PROT::.WORD   0
023752  177777                          .WORD   -1
023754  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::
                                        .BLKW   1
005232                          CREDIT.BAL::
                                        .BLKW   1
005234                          NEXT.PKT.USE::
                                        .BLKB   1

                                .GLOBL  L$SOFT, T$PTMV, 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$ERRTBL--      ERRTYP
023670'         L$SW--          L$SWLEN*2
023650'         L$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'         DU.MSG--        P.ABB
002616'         DU.RSN--        P.ABC
002646          MSG.01--        P.ABP
002700'         MSG.02--        P.ABQ
002734'         MSG.03--        P.ABR
002766'         RPT1--          P.ABS
003052'         RPT2--          P.ABT
003116'         RPT3--          P.ABU
003202'         RPT4--          P.ABV
003246'         RPT5--          P.ABW
003334'         RPT6--          P.ABX
003400'         RPT7--          P.ABY
003422'         RPT8--          P.ABZ
003444'         RPT9--          P.ACA
003472'         RPT10--         P.ACB
003524'         RPT11--         P.ACC
003612'         RPT12--         P.ACD
003660'         RPT13--         P.ACE
003760'         RPT14--         P.ACF
004056'         RPT15--         P.ACG
004156'         RPT16--         P.ACH
004240'         EGS.01--        P.ACI
004260'         EGS.02--        P.ACJ
004352'         EGD.10--        P.ACK
004412'         EGD.11--        P.ACL
004436'         EGD.12--        P.ACM
004464'         EGD.13--        P.ACN
004512'         EGD.14--        P.ACO
004542'         EGD.15--        P.ACP
004560'         EGD.16--        P.ACQ
004610'         EGD.17--        P.ACR
004626'         EGD.18--        P.ACS
004646'         EGD.20--        P.ACT
004734'         EGD.21--        P.ACU
005046'         EGD.22--        P.ACV
005106'         EGD.23--        P.ACW
005150'         EGH.30--        P.ACX
005174'         EBS.01--        P.ACY
005236'         EBD.10--        P.ACZ
005276'         EBD.12--        P.ADA
005344'         EBD.13--        P.ADB
005376'         EBD.14--        P.ADC
005436'         EBD.18--        P.ADD
005472'         EBD.19--        P.ADE
```

```
005552'                 EH.0--          P.ADF
005610'                 EH.1--          P.ADG
005646'                 EH.2--          P.ADH
005706'                 EH.3--          P.ADI
005744'                 EH.4--          P.ADJ
005770'                 EH.5--          P.ADK
006020'                 EH.6--          P.ADL
006052'                 EH.7--          P.ADM
006104'                 EH.8--          P.ADN
006140                  EH.9--          P.ADO
006172'                 EH.10--         P.ADP
006224                  EH.12--         P.ADQ
006262'                 EH.13--         P.ADR
007012'                 ERR.COD--       P.ADS
007046'                 ELG.OO--        P.AEH
007402'                 ELG.FMT--       P.AEI
007414'                 EX.BDR--        P.AEO
007504'                 EX.BDW--        P.AEP
007572'                 EX.LBR--        P.AEQ
007636'                 EX.LBW--        P.AER
007702'                 EX.RBN--        P.AES
007760'                 EX.CBR--        P.AET
010030'                 EX.CBW--        P.AEU
010100'                 XX13--          P.AEV
010122'                 XX14--          P.AEW
010136'                 XX15--          P.AEX
010160                  XX16--          P.AEY
010206'                 XX17--          P.AEZ
010224'                 XX18--          P.AFA
010234'                 XX19--          P.AFB
010246'                 XX20--          P.AFC
010262'                 XX21--          P.AFD
010322'                 XX22--          P.AFE
010356'                 XX23--          P.AFF
010412'                 XX24--          P.AFG
010452'                 XX25--          P.AFH
010522'                 XX26--          P.AFI
010576'                 XX27--          P.AFJ
010640'                 XX29--          P.AFK
010664'                 XX30--          P.AFL
010712'                 XX31--          P.AFM
010744'                 XX32--          P.AFN
010772'                 XX33--          P.AFO
011030'                 XX34--          P.AFP
011100'                 XX35--          P.AFQ
011116'                 XX37--          P.AFR
011142'                 XX38--          P.AFS
011206'                 XX39--          P.AFT
011234'                 XX40--          P.AFU
011254'                 XX41--          P.AFV
011304'                 EB.DCT--        P.AFW
011360'                 EB.COMM--       P.AFX
011442'                 EB.PKT--        P.AFY
011504'                 EB.RAL--        P.AFZ
011544'                 EB.ADDR--       P.AGA
011606'                 EBNEX1--        P.AGB
```

```
011662          EB.NEX2--       P.AGC
011760          EBNEX3--        P.AGD
012054          CER.01--        P.AGE
012120'         CER.02--        P.AGF
012174          EX.SEQ--        P.AGG
012214'         EX.CRD--        P.AGH
012244'         EX.MTN--        P.AGI
012264'         EX.DGM--        P.AGJ
012302'         EX.RD--         P.AGK
012312'         EX.WRT--        P.AGL
012322'         EX.ACC--        P.AGM
012334'         EX.ONL--        P.AGN
012346'         EX.SCC--        P.AGO
012372'         EX.GDS--        P.AGP
012414'         EX.ESP--        P.AGQ
012444'         EX.ELP--        P.AGR
012470'         EX.SDD--        P.AGS
012504'         EX.RCD--        P.AGT
012524'         EX.ABP--        P.AGU
012534'         SC.SDI--        P.AGV
012560'         SC.CON--        P.AGW
012602'         SC.DUP--        P.AGX
012632'         SC.ONL--        P.AGY
012654'         SC.SON--        P.AGZ
012674'         SC.UNK--        P.AHA
012754'         SC.VOL--        P.AHB
013034'         SC.IOP--        P.AHC
013102'         SC.DIS--        P.AHD
013174'         SC.FER--        P.AHE
013262'         SC.FE2--        P.AHF
013340'         SC.ISH--        P.AHG
013420'         SC.IS2--        P.AHH
013500'         SC.DST--        P.AHI
013554'         SC.DS2--        P.AHJ
013626'         SC.ECC--        P.AHK
013710'         SC.ECD--        P.AHL
013742'         SC.RCT--        P.AHM
013762'         SC.FUL--        P.AHN
014036'         SC.576--        P.AHO
014112'         SC.FCT--        P.AHP
014160'         SC.EC1--        P.AHQ
014210'         SC.EC2--        P.AHR
014240'         SC.EC3--        P.AHS
014272'         SC.EC4--        P.AHT
014322'         SC.EC5--        P.AHU
014352'         SC.EC6--        P.AHV
014402'         SC.EC7--        P.AHW
014434'         SC.EC8--        P.AHX
014466'         SC.EC9--        P.AHY
014530'         SC.SWP--        P.AHZ
014570'         SC.HWP--        P.AIA
014630'         SC.ODA--        P.AIB
014660'         SC.ODB--        P.AIC
014702'         SC.NXM--        P.AID
014736'         SC.PAR--        P.AIE
014772'         SC.CTO--        P.AIF
```

```
015044'                 SC.SDS**        P.AIG
015122                  SC.EDC**        P.AIH
015136                  SC.IDS**        P.AII
015206'                 SC.SRT**        P.AIJ
015300'                 SC.SRI**        P.AIK
015366'                 SC.POE**        P.AIL
015422'                 SC.RDY**        P.AIM
015504'                 SC.CLK**        P.AIN
015532'                 SC.RSP**        P.AIO
015600'                 SC.SUR**        P.AIP
015630'                 SC.PSP**        P.AIQ
015712'                 F.1**           P.AIR
015746'                 F.2**           P.AIS
016002'                 F.3**           P.AIT
016034'                 F.4**           P.AIU
016074'                 F.5**           P.AIV
016152'                 F.6**           P.AIW
016226'                 F.7**           P.AIX
016302'                 F.8**           P.AIY
016360'                 F.9**           P.AIZ
016400'                 F.10**          P.AJA
016430'                 F.11**          P.AJB
016454'                 F.12**          P.AJC
016502'                 F.13**          P.AJD
016560'                 F.14**          P.AJE
016620'                 F.15**          P.AJF
016646'                 F.16**          P.AJG
016712'                 F.17**          P.AJH
016756'                 F.18**          P.AJI
017016'                 F.19**          P.AJJ
017056'                 F.20**          P.AJK
017130'                 F.21**          P.AJL
017160'                 EBH.30**        P.AJM
017176'                 EBH.44**        P.AJN
017300'                 EBH.45**        P.AJO
017332'                 EBH.46**        P.AJP
017362'                 EBH.47**        P.AJQ
017444'                 EBH.48**        P.AJR
017532'                 EBH.49**        P.AJS
017644'                 DF.0**          P.AJT
017700'                 DF.1**          P.AJU
017734'                 DF.2**          P.AJV
020010'                 DF.3**          P.AJW
020050'                 DF.4**          P.AJX
020076'                 DF.5**          P.AJY
020122'                 DF.6**          P.AJZ
020172'                 DF.7**          P.AKA
020242'                 T.QUE**         P.AKB
020264'                 T.DEF**         P.AKC
020316'                 T.INF**         P.AKD
020342'                 T.TER**         P.AKE
020366'                 T.FAT**         P.AKF
020412'                 T.SPL**         P.AKG
020432                  E.UNT**         P.AKH
020466'                 E.BLK**         P.AKI
020550'                 E.DEV**         P.AKJ
```

```
        020574                  E.ZER--             P.AKK
        020630                  M.ASC--             P.AKL
        020664'                 M.BIN--             P.AKM
        020722'                 M.TER--             P.AKN
        020756'                 M.COD--             P.AKO
        021014'                 M.DAT--             P.AKP
        021046'                 M.UR--              P.AKQ
        021116'                 M.URP--             P.AKR
        021204'                 M.UP--              P.AKS
        021264'                 M.UL--              P.AKT
        023042'                 CNTR.ERR--          P.AKU
        023562'                 RDRX.ERR--          P.ALS
        023600'                 EX.WRD--            P.AMA
        023610                  EX.OP--             P.AMB
        023622'                 SPACE4--            P.AMC
        023626'                 CRLF--              P.AMD
        023632'                 DASH--              P.AME
        023640'                 ASTERISK--          P.AMF
        023650'                 DFPTBL--            L$HWLEN*2
        023670'                 SFPTBL--            L$SWLEN*2

                                        .EVEN
;                               PSECT SUMMARY
;
;       Psect Name              Words     Attributes
;        $CODE$                  5111       RO  ,  I  ,  LCL,  REL,  CON
;        $FFF$                   1358       RO  ,  I  ,  LCL,  REL,  CON

;                                  LIBRARY STATISTICS
;
;                                          --------  Symbols  --------     Blocks
;       File                               Total     Loaded   Percent       Read
;
;  SPIDER$USERS:[DOUCETTE.FALCON]CNRQAA.L16;7
;                                           457        227        49         54


;                                       COMMAND QUALIFIERS
;       BLISS /PDP11 CNRQAA.BL1/LIST=CNRQAA.LI1/OBJECT=CNRQAA.OB1/SOURCE=PAGE:56
```

```
module NRQAM2 (

%title 'RD/RX EXERCISER
                 dent =  V01.2 ,
                addressing mode (absolute),
                environment (noe's)
                ) =

begin

%sbttl  DECLARATIONS'

library 'CNRQAA.L16';                        ! RDRX EXERCISER GLOBAL LIBRARY

require 'BLSMAC.REQ';                        ! DIAGNOSTIC SUPERVISOR LIBRARY

forward routine
    NEX_TRAP : L$ISR novalue,
    EMS_01 : novalue,
    EMS_DUP : novalue,
    EMS_BLK : novalue,
    EMSCMD : NOVALUE,
    SET_CPAR : novalue,
    SET_UPAR : novalue,
    EMS_DBN : novalue;
external
    CST : blockvector [MAX_CTLR, CST_LEN, word] field (CST_FIELDS),
                        ! RUN-TIME CONTROLLER STATUS TABLES
    CST_ADDR : ref block [CST_LEN, word] field (CST_FIELDS),
                        ! CONTROLLER STATUS TABLE ADDRESS OF "CURRENT" CONTROLLER
    DCT : blockvector [MAX_CTLR, DCT_LEN, word] field (DCT_FIELDS),
                        ! DRIVER CONTROLLER TABLES
    DCT_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS),
                        ! ADDRESS OF "CURRENT" DRIVER CONTROLLER TABLE
    RDRX_ADDR : ref rdrx field (RC_REG),
                        ! DEVICE ADDRESS OF "CURRENT" CONTROLLER
    IRDRX_ADDR : ref rdrx field (RC_REG),
                        ! DEVICE ADDRESS OF INTERRUPTING CONTROLLER
    DUPPKT : BLOCK [257, WORD] field (DP_FIELDS),
                ! BUFFER CONTAINING DUP INFORMATION FROM RECEIVE AND SEND COMMANDS
    TALLY : vector [MAX_UNITS * TALLY_LEN, word] field (T_FIELDS),
                        ! STATISTICS TABLES
    T_ADDR : ref block [TALLY_LEN, word] field (T_FIELDS),
                        ! ADDRESS OF STATISTICS TABLE (TALLY) FOR CURRENT UNIT
    C_ERR_TBL : blockvector [MAX_CTLR, C_ERR_LEN, word] field (C_ERR_FIELDS),
                        ! STATISTICS TABLE FOR CONTROLLER ERRORS
    MSCP_PKT : blockvector [PKT_CNT, PKT_LEN, word] field (PKT_FIELDS),
                        ! MSCP PACKET POOL
    IPKT_ADDR : ref block [PKT_LEN, word] field (PKT_FIELDS),
                        ! ADDRESS OF AN MSCP PACKET (INTERRUPT PROCESSING)
    PKT_USE : vector [PKT_CNT, byte, signed],
                        ! MSCP PACKET POOL ALLOCATION TABLE
    RETPKT : blockvector [RP_CNT, RP_LEN, word] field (RP_FIELDS),
                        ! RETURN PACKET POOL
    RP_USE : vector [RP_CNT, byte, signed],
                        ! RETURN PACKET POOL ALLOCATION TABLE
```

```
;        4906      RP_INDX : word,      ! CURRENT RETURN PACKET INDEX
;        4907      RP_ADDR : ref block [RP_LEN, word] field (RP_FIELDS),
;        4908                          ! CURRENT RETURN PACKET ADDRESS
;        4909      ELOG_PKT : blockvector [EP_CNT, EP_LEN, word] field (EP_FIELDS),
;        4910                          ! ERROR LOG PACKET SAVE AREA
;        4911      BUFF_ADDR : vector [MAX_BUF_CNT],                  ! TABLE OF I/O BUFFER DESCRIPTORS
;        4912      BUFF_OWN : vector [MAX_BUF_CNT, byte, signed],     ! I/O BUFFER OWNERSHIP (CONTROLLER NUMBER)
;        4913      IODQ : vector [IODQ_LEN, byte],
;        4914                                        ! I/O DONE QUEUE - CIRCULAR QUEUE OF RETPKT INDECES
;        4915      IODQ_IN : word,                   ! I/O DONE QUEUE IN POINTER
;        4916      IODQ_OUT : word,                  ! I/O DONE QUEUE OUT POINTER
;        4917      ENTRY_REASON : byte,              ! CURRENT OPERATOR COMMAND
;        4918      EOP_FLAG : byte,                  ! END-OF-PASS FLAG
;        4919      DUP_FLAGS : WORD,                 ! DUP FLAGS
;        4920      CCTLR : word,                     ! NUMBER OF "CURRENT" CONTROLLER
;        4921      CDISK : word,                     ! CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;        4922      CUOFF : word,                     ! CURRENT UNIT CST OFFSET
;        4923      CTLR_CNT : word,                  ! TOTAL NUMBER OF CONFIGURED CONTROLLERS
;        4924      DUR : vector [MAX_UNITS, byte],   ! DROP UNIT REASON
;        4925      QIO : vector [MAX_CTLR, byte],    ! NUMBER OF OUTSTANDING QIOS PER CONTROLLER
;        4926      FREE_MEM_ADDR,                    ! START OF FREE MEMORY
;        4927      BYTS_PER_QIO : word,              ! SIZE (BYTES) OF AN I/O BUFFER
;        4928      ST_CODE : word,                   ! CURRENT STATUS CODE
;        4929      SB_CODE : word,                   ! CURRENT SUB-CODE
;        4930      STEP : word,                      ! CURRENT STEP IN HARD_INIT
;        4931      OF_RC : signed word,              ! OFFSET (0 OR 2) TO READ IP OR SA
;        4932      SA_REG : word,                    ! STORAGE FOR SA REGISTER READS AND WRITES
;        4933      CMD_TIME : word,                  ! COMMAND TIMEOUT VALUE (IN SECONDS)
;        4934      NEX : word,                       ! NON-EXISTENT MEMORY TRAP INDICATOR
;        4935      CRN_LOW : word,                         ! COMMAND REF NUMBER OF LAST COMMAND SENT
;        4936      CRN_HIGH : word,                        ! COMMAND REF NUMBER (HI ORDER)
;        4937      P_INDEX : signed word,          ! CURRENT message PACKET INDEX
;        4938      S_DUPPKT : WORD,                ! DBN BYTE COUNTER
;        4939      S_PATTERN : WORD,               ! THE PATTERN WRITTEN TO DBN'S
;        4940      CREDIT_BAL : word,                      ! CREDIT BALANCE
;        4941      NEXT_PKT_USE : byte,                    ! POINTER TO NEXT ENTRY IN PKT USE TABLE
;        4942  !     DBM5,                                 !JSD REV A - REMOVED
;        4943  !     DBM107,                               !JSD REV A - REMOVED
;        4944      DU_MSG,
;        4945      DU_RSN : vector [12],
;        4946      ERR_COD : vector [14],
;        4947      ELG_FMT : vector [5],
;        4948  !
;        4949      HWQ1,
;        4950      HWQ2,
;        4951      HWQ3,
;        4952      HWQ4,
;        4953      HWQ5,
;        4954      HWQ6,
;        4955      HWQ7,
;        4956      HWQ8,
;        4957      HWQ9,
;        4958      HWQ10,
;        4959      HWQ11,
;        4960      SWQ1,
;        4961      SWQ2,
```

```
;        4962        SWQ4.
;        4963        SWQ7.
.        4964        SWQ9.
;        4965        SWQ10.
;        4966        SWQ11.
.        4967        SWQ12.
;        4968        SWQ13.
;        4969        SWQ14.
;        4970        SWQ15.
;        4971        SWQ17.
;        4972        SWQ19.
;        4973        SWQ22.
;        4974
;        4975        SWM1.
;        4976        NULL.
;        4977        MSG_01.
;        4978        MSG_02.
;        4979        MSG_03.
;        4980        RPT1.
;        4981        RPT2.
;        4982        RPT3.
;        4983        RPT4.
;        4984        RPT5.
;        4985        RPT6.
;        4986        RPT7.
;        4987        RPT8.
;        4988        RPT9.
;        4989        RPT10.
;        4990        RPT11.
.        4991        RPT12.
;        4992        RPT13.
.        4993        RPT14.
;        4994        RPT15.
;        4995        RPT16.
;        4996        EGS_01.
.        4997        EGS_02.
.        4998        EGD_10.
.        4999        EGD_11.
.        5000        EGD_12.
.        5001        EGD_13.
.        5002        FGD_14.
.        5003        EGD_15.
         5004        EGD_16.
.        5005        EGD_17.
.        5006        EGD_18.
.        5007        EGD_20.
.        5008        EGD_21.
.        5009        EGD_22.
;        5010        EGD_23.
;        5011        EGH_30.
.        5012        EM_0.
;        5013        EM_1.
.        5014        EM_2.
.        5015        EM_3.
;        5016        EM_4.
;        5017        EM_5.
```

```
;       5018        EH_6.
;       5019        EH_7.
;       5020        EH_8.
;       5021        EH_9.
;       5022        EH_10.
.       5023        EH_12.
.       5024        EH_13.
.       5025
.       5026        ERS_01.
;       5027        EBD_10.
.       5028        EBD_12.
.       5029        EBD_13.
.       5030        EBD_14.
.       5031        EBD_18.
.       5032        EBD_19.
.       5033            ELG_00.
.       5034            EX BDR.
.       5035        EX_BDW.
.       5036        EX_LBR.
;       5037        EX_LBW.
.       5038        EX_RBN.
.       5039        EX_CBR.
.       5040        EX_CBW.
.       5041    XX13.
;       5042    XX14.
.       5043    XX15.
.       5044    XX16.
.       5045    XX17.
;       5046    XX18.
.       5047    XX19.
.       5048    XX20.
.       5049    XX21.
.       5050    XX22.
.       5051    XX23.
.       5052    XX24.
        5053    XX25.
.       5054    XX26.
.       5055    XX27.
.       5056    XX29.
        5057    XX30.
.       5058    XX31.
.       5059    XX32.
.       5060    XX33.
.       5061    XX34.
        5062    XX35.
.       5063    !XX36.
        5064    XX37.
.       5065    XX38.
.       5066    XX39.
.       5067    XX40.
        5068    XX41.
        5069    EB DCT.
        5070    EB COMM.
.       5071    EB PKT.
        5072    EB RAL.
        5073    EB ADDR.
```

```
.       EBNEX1,
;       EB_NEX2,
.       EBNEX3,
;       CER_01,
;       CER_02,
;           EX_SEQ,
.           EX_CRD,
;           EX_MTN,
.           EX_DGM,
.           EX_RD,
.           EX_WRT,
.           EX_ACC,
;           EX_ONL,
.           EX_SCC,
;           EX_GDS,
.           EX_ESP,
.           EX_ELP,
;           EX_SDO,
.           EX_RCD,
;           EX_ABP,
;           SC_SDI,
            SC_CON,
;           SC_DUP,
;           SC_ONL,
;           SC_SON,
;           SC_UNK,
;           SC_VOL,
;           SC_IOP,
;           SC_DIS,
;           SC_FER,
;           SC_FE2,
;           SC_ISH,
;           SC_IS2,
.           SC_DST,
;           SC_DS2,
;           SC_ECC,
;           SC_ECD,
;           SC_RCT,
;           SC_FUL,
;           SC_576,
;           SC_FCT,
;           SC_EC1,
;           SC_EC2,
;           SC_EC3,
;           SC_EC4,
;           SC_EC5,
;           SC_EC6,
;           SC_EC7,
;           SC_EC8,
;           SC_EC9,
;           SC_SWP,
;           SC_HWP,
;           SC_ODA,
;           SC_ODB,
;           SC_NXM,
;           SC_PAR,
```

```
;          5130        SC_CTO,
;          5131        SC_SDS,
;          5132        SC_EDC,
;          5133        SC_IDS,
;          5134        SC_SRT,
;          5135        SC_SRI,
;          5136        SC_POE,
;          5137        SC_RDY,
;          5138        SC_CLK,
;          5139        SC_RSP,
;          5140        SC_SUR,
;          5141        SC_PSP,
;          5142    F_1,
;          5143    F_2,
;          5144    F_3,
;          5145    F_4,
;          5146    F_5,
;          5147    F_6,
;          5148    F_7,
;          5149    F_8,
;          5150    F_9,
;          5151    F_10,
;          5152    F_11,
;          5153    F_12,
;          5154    F_13,
;          5155    F_14,
;          5156    F_15,
;          5157    F_16,
;          5158    F_17,
;          5159    F_18,
;          5160    F_19,
;          5161    F_20,
;          5162    F_21,
;          5163        EBH_30,
;          5164        EBH_44,
;          5165        EBH_45,
;          5166        EBH_46,
;          5167        EBH_47,
;          5168        EBH_48,
;          5169        EBH_49,
;          5170        df_0,
;          5171        df_1,
;          5172        df_2,
;          5173        df_3,
;          5174        df_4,
;          5175        df_5,
;          5176        df_6,
;          5177        df_7,
;          5178        T_QUE,
;          5179        T_DEF,
;          5180        T_INF,
;          5181        T_TER,
;          5182        T_FAT,
;          5183        T_SPL,
;          5184        E_UNT,
;          5185        E_BLK,
```

```
;          5186      E_DEV,
;          5187      E_ZER,
;          5188      M_ASC,
;          5189      M_BIN,
;          5190      M_TER,
;          5191      M_COD,
;          5192      M_DAT,
;          5193      M_UR ,
;          5194      M_URP,
;          5195      M_UP,
;          5196      M_UL,
;          5197      CNTR_ERR : vector [23],
;          5198      RDRX_ERR : vector [7],
;          5199      EX_WRD,
;          5200      EX_OP,
;          5201      SPACE4,
;          5202      CRLF,
;          5203      DASH,
;          5204      ASTERISK,
;          5205      SWP_FLAGS : word,
;          5206      L$HIMEM,
;          5207      L$LUN,
;          5208      L$UNIT;
;          5209
.          5210  own
;          5211      TBL_SUC : vector [17] initial (NULL, SC_SDI, SC_CON, NULL, SC_DUP, NULL, NULL,
;          5212                NULL, SC_ONL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, SC_SON),
;          5213      TBL_OFL : vector [9] initial (SC_UNK, SC_VOL, SC_IOP, NULL, SC_DUP, NULL, NULL,
;          5214                NULL, SC_DIS),
;          5215      TBL_MFE : vector [11] initial (SC_FER, NULL, SC_ISM, SC_DST, SC_EC9, SC_576,
;          5216                SC_FCT, SC_ECC, SC_RCT, SC_FUL, SC_EC1),
;          5217      TBL_WPT : vector [3] initial (NULL, SC_SWP, SC_HWP),
;          5218      TBL_DAT : vector [16] initial (SC_FE2, NULL, SC_IS2, SC_DS2, SC_EC9, NULL, NULL,
;          5219                SC_ECD, SC_EC1, SC_EC2, SC_EC3, SC_EC4, SC_EC5, SC_EC6, SC_EC7, SC_EC8),
;          5220      TBL_HST : vector [5] initial (NULL, SC_ODA, SC_ODB, SC_NXM, SC_PAR),
;          5221      TBL_CNT : vector [4] initial (SC_CTO, SC_SDS, SC_EDC, SC_IDS),
;          5222      TBL_DRV : vector [9] initial (NULL, SC_SRT, SC_SRI, SC_POE, SC_RDY, SC_CLK, SC_RSP,
;          5223                SC_SUR, SC_PSP);
```

```
;       5224  %sbttl  TYPE AND DESCRIPTION
;       5225
;       5226  EQUALS;
;       5227
;       5228  DEVTYP (%asciz RQDX1 );                 ! NAME OF DEVICE SUPPORTED BY PROGRAM
;       5229
;       5230  DESCRIPT (%asciz'RD/RX EXERCISER');     ! TEST DESCRIPTION


;       5231  %sbttl 'HARDWARE PARAMETER CODING SECTION'
;       5232
;       5233  !+
;       5234  ! THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
;       5235  ! THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
;       5236  ! MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
;       5237  ! INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
;       5238  ! MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
;       5239  ! WITH THE OPERATOR.
;       5240  !-
;       5241
;       5242  BGNHRD;
;       5243
;       5244  GPRMA (HWQ1, 0, 0, %o'160000', %o'177777', YES, 1);                    ! IP ADDRESS
;       5245  GPRMA (HWQ2, 2, 0, %o'4', %o'774', YES, 1);                            ! VECTOR
;       5246  GPRMD (HWQ3, 4, 0, %o'377', %o'0', %o'7', YES, 1);                    ! BR LEVEL
;       5247  GPRMD (HWQ4, 6, 0, %o'3', %decimal'0', %decimal'3', YES, 1);          ! RDRX DRIVE NUMBER
;       5248  GPRML (HWQ5, 6, %o'4', YES, 1);                                       ! UNIT TYPE
;       5249  XFERF (HW1);
;       5250  GPRML (HWQ10, 6, %o'000010', YES, 1);                                 ! run dup exerciser
;       5251  XFERF (NODU);
;       5252  GPRML (HWQ11, 6, %o'000020', YES, 1);                                 ! WRITE TO DBN'S
;       5253  %L (NODU);
;       5254  GPRMD (HWQ6, 8, 0, %o'177777', %decimal'0', RD_MAX_LBN, YES, 1);      ! STARTING LBN
;       5255  GPRMD (HWQ7, 10, 0, %o'177777', GP$ATLO (8), RD_MAX_LBN, YES, 1);    ! ENDING LBN
;       5256  XFER (HW2);
;       5257  %L (HW1);
;       5258  GPRMD (HWQ6, 8, 0, %o'177777', %decimal'0', RX_MAX_LBN, YES, 1);      ! STARTING LBN
;       5259  GPRMD (HWQ7, 10, 0, %o'177777', GP$ATLO (8), RX_MAX_LBN, YES, 1);    ! ENDING LBN
;       5260  %L (HW2);
;       5261  GPRML (HWQ8, 6, %o'100000', NO, 1);                                  ! EXER ON CUST DATA AREA
;       5262  XFERF (HWDONE);                                                       ! NO - DONE
;       5263  GPRML (HWQ9, 6, %o'100000', NO, 1);                                  ! ** WARNING / CONFIRM
;       5264  %L (HWDONE);
;       5265
;       5266  ENDHRD;
```

```
;        5267  %sbttl 'SOFTWARE PARAMETER CODING SECTION'
;        5268
;        5269  !+
;        5270  ! THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
;        5271  ! THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
;        5272  ! MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
;        5273  ! INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
;        5274  ! MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
;        5275  ! WITH THE OPERATOR.
;        5276  !
;        5277
;        5278  BGNSFT;
;        5279
;        5280  !GPRML (SWQ16, 4, SWF_TRC, YES, 1);                    ! ENABLE DIAGNOSTIC TRACE
;        5281  GPRMD (SWQ1, 0, D, %o'177777', 0, 65535, YES, 1);      ! ERROR LIMIT
;        5282  GPRMD (SWQ2, 2, D, %o'177777', 0, 99, YES, 1);         ! TRANSFER LIMIT
;        5283  GPRMD (SWQ17, 10, D, %o'177777', 0, 100, YES, 1);      ! PERCENT OF RD OPERATIONS
;        5284  GPRMD (SWQ22, 12, D, %o'177777', 0, 144, YES, 1);      ! NUMBER OF DBN'S WRITTEN AT ONE TIME


;        5285  GPRML (SWQ15, 4, SWF_CST, YES, 1);                    ! CLEAR STATISTICAL TABLES ?
;        5286  !GPRML (SWQ20, 4, SWF_FER, YES, 1);                   ! REWRITE BLOCKS WHEN "FORCED ERROR' BIT SET?
;        5287  !GPRML (SWQ21, 4, SWF_HOE, YES, 1);                   ! HALT ON HARD/SOFT ERRORS WITH 'HOE' FLAG SET?
;        5288  GPRML (SWQ4, 4, SWF_RDM, YES, 1);                     ! RANDOM SEEK MODE ?
;        5289  XFERF (SW1);                                          ! IF NO, DO NEXT QUESTION
;        5290  XFER (SW2),
;        5291  $L (SW1);
;        5292  GPRML (SWQ19, 4, SWF_SEQ, YES, 1);                    ! FIXED OR RANDOM SEQUENTAIL STEPPING ?
;        5293  $L (SW2);
;        5294  GPRML (SWQ7, 4, SWF_CRC, YES, 1);                     ! READ-COMPARES AT CONTROLLER ?
;        5295  DISPLAY (SWM1);                                       ! REMAINING QUESTIONS ONLY APPLY ...
;        5296  GPRML (SWQ9, 4, SWF_CWC, YES, 1);                     ! WRITE-COMPARES AT CONTROLLER ?
;        5297  XFERF (SW3);                                          ! IF NO, DO NEXT QUESTION
;        5298  XFER (SW4);
;        5299  $L (SW3);
;        5300  GPRML (SWQ10, 4, SWF_HWC, YES, 1);                    ! CHECK WRITES AT HOST BY READING ?
;        5301  $L (SW4);
;        5302  GPRML (SWQ11, 4, SWF_UDP, YES, 1);                    ! USER-DEFINED DATA PATTERN ?
;        5303  XFERF (SW5);                                          ! IF NO, DO NEXT QUESTION
;        5304  XFER (SW6);
;        5305  $L (SW5);
;        5306  GPRMD (SWQ12, 6, D, %o'177777', 0, DP_CNT, YES, 1);   ! SELECT PRE-DEFINED DATA PATTERN
;        5307  XFER (SW7);                                           ! DONE
;        5308  $L (SW6);
;        5309  GPRMD (SWQ13, 8, D, %o'177777', 1, MAX_UDP_CNT, YES, 1);! NO. OF WORDS IN USER DATA PATTERN
;        5310  GPRMD (SWQ14, 14, O, %o'177777', 0, %o'177777', NO, 8); ! PATTERN VALUES
;        5311  $L (SW7);
;        5312
;        5313  ENDSFT;
```

```
;          5314  %sbttl  REPORT CODING SECTION
;          5315
;          5316  !+
;          5317  ! THE REPORT CODING SECTION CONTAINS THE
;          5318  ! "PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.
;          5319  !
;          5320
;          5321  BGNRPT;
;          5322                                                                  ! PRINTS MSCP DATA
;          5323  PRINTS (RPT1);
;          5324  PRINTS (RPT2);
;          5325  PRINTS (RPT3);
;          5326  PRINTS (RPT4);
;          5327  PRINTS (RPT5);
;          5328  PRINTS (RPT6);
;          5329
;          5330  incr CTLR from 0 to MAX_CTLR - 1 do
;          5331      begin
;          5332      SET_CPAR (.CTLR);
;          5333
;          5334      incr DISK from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;          5335          begin
;          5336          SET_UPAR (.DISK);
;          5337
;          5338          if (.CST_ADDR [.DISK, D_TYPE] eql RX_50) and
;          5339             (.CST_ADDR [.DISK, D_PRES] eql PRESENT)
;          5340          then
;          5341              PRINTS (RPT7, .L$LUN, .CST_ADDR [.DISK, D_DISK_NUM]);
;          5342
;          5343          if (.CST_ADDR [.DISK, D_TYPE] eql RD_51) and
;          5344             (.CST_ADDR [.DISK, D_PRES] eql PRESENT)
;          5345          then
;          5346              PRINTS (RPT8, .L$LUN, .CST_ADDR [.DISK, D_DISK_NUM]);
;          5347
;          5348          if .CST_ADDR [.DISK, D_PRES] eql PRESENT
;          5349          then
;          5350              begin
;        P 5351              PRINTS (RPT9,
;        P 5352                  .T_ADDR [TOT_READS_HI], .T_ADDR [TOT_READS_LO],
;          5353                  .T_ADDR [MTOT_BYT_RED], .T_ADDR [TOT_BYT_RED_HI], .T_ADDR [TOT_BYT_RED_LO]);
;        P 5354              PRINTS (RPT9,
;        P 5355                  .T_ADDR [TOT_WRITES_HI], .T_ADDR [TOT_WRITES_LO],
;          5356                  .T_ADDR [MTOT_BYT_WRT], .T_ADDR [TOT_BYT_WRT_HI], .T_ADDR [TOT_BYT_WRT_LO]);
;        P 5357              PRINTS (RPT10,
;        P 5358                  .T_ADDR [ERR_HRD_SEK], .T_ADDR [ERR_HRD_DAT], .T_ADDR [ERR_HRD_DRV], .T_ADDR [ERR_HRD_HST],
;          5359                  .T_ADDR [ERR_SFT_SEK], .T_ADDR [ERR_SFT_DAT], .T_ADDR [ERR_SFT_DRV], .T_ADDR [ERR_SFT_HST]);
;          5360              end;
;          5361
;          5362          end;
;          5363
;          5364      if .CST [.CTLR, STATE] eql PRESENT
;          5365      then
;          5366          begin
;          5367          PRINTS (RPT11);
;          5368          PRINTS (RPT12, .C_ERR_TBL [.CTLR, C_ERR_HRD], .C_ERR_TBL [.CTLR, C_ERR_SFT]);
;          5369          end;
```

```
;         5370
.         5371      PRINTS (CRLF);
;         5372
;         5373      end;
;         5374  begin                                                          ! PRINTS DUP DATA
.         5375  prints(crlf);
;         5376  PRINTS(RPT13);!!
;         5377  PRINTS(RPT14);
.         5378  PRINTS(RPT15);
;         5379  INCR CTLR FROM 0 TO MAX_CTLR 1 DO
.         5380          BEGIN
;         5381          SET_CPAR(.CTLR);
;         5382          INCR DISK FROM (0+OF_UN) TO (3+UNIT_SIZE+OF UN) BY UNIT SIZE DO
;         5383                  BEGIN
;         5384                  SET_UPAR(.DISK);
;         5385                  IF .CST_ADDR[.DISK, D TYPE] EQLU RD 51 and .CST_ADDR [.DISK, D_PRES] eql PRESENT
;         5386                  THEN
.       P 5387              PRINTS (RPT16,
;       P 5388                  .L$LUN, .CST_ADDR [.DISK, D_DISK_NUM],
;         5389                  .T_ADDR [T_DBN_RD], .T_ADDR [T_BLK_RD], .T_ADDR [T_DBN_WT], .T ADDR [T_BLK_WT]);
;         5390                  END;
;         5391          END;
;         5392  end;
;         5393
;         5394  PRINTS (CRLF);
.         5395
.         5396  ENDRPT;
```

```
                                .TITLE  NRQAM2 RD/RX EXERCISER
                                .IDENT  /V01.2/
                                .ENABL  AMA

000000                          .PSECT  $CODE$,  RO
000000  122  121  104    L$DVTYP::
                                .ASCII  /RQD/
000003  130  061  000            .ASCII  /X1/<00>
000006                          .BLKB   2
000010  122  104  057    L$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$21:: .WORD   2130
000342  000000                          .WORD   SWQ7
000344  000004                          .WORD   4
000346  000003                  GP$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  00    2                 GP$27::  .WORD   7222
000444  000   0G                         .WORD   SWQ14
000446  1777  7                          .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.DNL
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, DU.MSG, DU.RSN, ERR.COD
.GLOBL  ELG.FMT, HWQ1, HWQ2, HWQ3, HWQ4
.GLOBL  HWQ5, HWQ6, HWQ7, HWQ8, HWQ9, HWQ10
.GLOBL  HWQ11, SWQ1, SWQ2, SWQ4, SWQ7
.GLOBL  SWQ9, SWQ10, SWQ11, SWQ12, SWQ13
.GLOBL  SWQ14, SWQ15, SWQ17, SWQ19, SWQ22
.GLOBL  SWM1, NULL, MSG.01, MSG.02, MSG.03
.GLOBL  RPT1, RPT2, RPT3, RPT4, RPT5, RPT6
.GLOBL  RPT7, RPT8, RPT9, RPT10, RPT11
.GLOBL  RPT12, RPT13, RPT14, RPT15, RPT16
.GLOBL  EGS.01, EGS.02, EGD.10, EGD.11
.GLOBL  EGD.12, EGD.13, EGD.14, EGD.15
.GLOBL  EGD.16, ECD.17, EGD.18, EGD.20
.GLOBL  EGD.21, EGD.22, EGD.23, EGH.30
.GLOBL  EH.0, EH.1, EH.2, EH.3, EH.4, EH.5
.GLOBL  EH.6, EH.7, EH.8, EH.9, EH.10
.GLOBL  EH.12, EH.13, EBS.01, EBD.10, EBD.12
.GLOBL  EBD.13, EBD.14, EBD.18, EBD.19
.GLOBL  ELG.00, EX.BDR, EX.BDW, EX.LBR
.GLOBL  EX.LBW, EX.RBN, EX.CBR, EX.CBW
.GLOBL  XX13, XX14, XX15, XX16, XX17, XX18
.GLOBL  XX19, XX20, XX21, XX22, XX23, XX24
.GLOBL  XX25, XX26, XX27, XX29, XX30, XX31
.GLOBL  XX32, XX33, XX34, XX35, XX37, XX38
.GLOBL  XX39, XX40, XX41, EB.DCT, EB.COMM
.GLOBL  EB.PKT, EB.RAL, EB.ADDR, EBNEX1
.GLOBL  EB.NEX2, EBNEX3, CER.01, CER.02
.GLOBL  EX.SEQ, EX.CRD, EX.MTN, EX.DGM
.GLOBL  EX.RD, EX.WRT, EX.ACC, EX.ONL
.GLOBL  EX.SCC, EX.GDS, EX.ESP, EX.ELP
.GLOBL  EX.SDD, EX.RCD, EX.ABP, SC.SDI
.GLOBL  SC.CON, SC.DUP, SC.ONL, SC.SON
.GLOBL  SC.UNK, SC.VOL, SC.IOP, SC.DIS
.GLOBL  SC.FER, SC.FE2, SC.ISH, SC.IS2
.GLOBL  SC.DST, SC.DS2, SC.ECC, SC.ECD
.GLOBL  SC.RCT, SC.FUL, SC.576, SC.FCT
.GLOBL  SC.EC1, SC.EC2, SC.EC3, SC.EC4
.GLOBL  SC.EC5, SC.EC6, SC.EC7, SC.EC8
.GLOBL  SC.EC9, SC.SWP, SC.HWP, SC.ODA
.GLOBL  SC.ODB, SC.NXM, SC.PAR, SC.CTO
.GLOBL  SC.SDS, SC.EDC, SC.IDS, SC.SRT
.GLOBL  SC.SRI, SC.POE, SC.RDY, SC.CLK
.GLOBL  SC.RSP, SC.SUR, SC.PSP, F.1, F.2
.GLOBL  F.3, F.4, F.5, F.6, F.7, F.8, F.9
.GLOBL  F.10, F.11, F.12, F.13, F.14, F.15
.GLOBL  F.16, F.17, F.18, F.19, F.20, F.21
.GLOBL  EBH.30, EBH.44, EBH.45, EBH.46
.GLOBL  EBH.47, EBH.48, EBH.49, DF.0, DF.1
.GLOBL  DF.2, DF.3, DF.4, DF.5, DF.6, DF.7
.GLOBL  T.QUE, T.DEF, T.INF, T.TER, T.FAT
.GLOBL  T.SPL, E.UNT, E.BLK, E.DEV, E.ZER
.GLOBL  M.ASC, M.BIN, M.TER, M.COD, M.DAT
```

```
                        .GLOBL  M.UR, M.URP, M.UP, M.UL, CNTR.ERR
                        .GLOBL  RDRX.ERR, EX.WRD, EX.OP, SPACE4
                        .GLOBL  CRLF, DASH, ASTERISK, SWP.FLAGS
                        .GLOBL  L$HIMEM, L$LUN, 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                  IOE=:           40000
        100000                  MOE=:           100000
        000034                  L$HARD=:        L$HRDLN*2
        000256                  L$SOFT=:        L$SFTLN*2


                                        .SBTTL  LRPT REPORT CODING SECTION
                                        .PSECT  $CODE$,  RO
000042

000000  004157  000000G       LRPT:   JSR     R1,$SAVE3                       ;                               5313
000004  012746  000000G               MOV     #RPT1,-(SP)                     ;                               5323
000010  012746  000001                MOV     #1,-(SP)
000014  010600                        MOV     SP,R0                           ; SP,*
000016  104416                        TRAP    16
000020  012716  000000G               MOV     #RPT2,(SP)                      ;                               5324
000024  012746  000001                MOV     #1,-(SP)
000030  010600                        MOV     SP,R0                           ; SP,*
000032  104416                        TRAP    16
000034  012716  000000G               MOV     #RPT3,(SP)                      ;                               5325
000040  012746  000001                MOV     #1,-(SP)
000044  010600                        MOV     SP,R0                           ; SP,*
000046  104416                        TRAP    16
000050  012716  000000G               MOV     #RPT4,(SP)                      ;                               5326
000054  012746  000001                MOV     #1,-(SP)
000060  010600                        MOV     SP,R0                           ; SP,*
000062  104416                        TRAP    16
000064  012716  000000G               MOV     #RPT5,(SP)                      ;                               5327
000070  012746  000001                MOV     #1,-(SP)
000074  010600                        MOV     SP,R0                           ; SP,*
000076  104416                        TRAP    16
000100  012716  000000G               MOV     #RPT6,(SP)                      ;                               5328
000104  012746  000001                MOV     #1,-(SP)
000110  010600                        MOV     SP,R0                           ; SP,*
000112  104416                        TRAP    16
000114  005003                        CLR     R3                              ; CTLR                          5330
000116  010316          1$:           MOV     R3,(SP)                         ; CTLR,*                        5332
000120  004737  000000V               JSR     PC,SET.CPAR
000124  012702  000003                MOV     #3,R2                           ; *,DISK                        5334
000130  010216          2$:           MOV     R2,(SP)                         ; DISK,*                        5336
000132  004737  000000V               JSR     PC,SET.UPAR
000136  010201                        MOV     R2,R1                           ; DISK,*                        5338
000140  006301                        ASL     R1
000142  010100                        MOV     R1,R0
000144  063700  000000G               ADD     CST.ADDR,R0
000150  132710  000004                BITB    #4,(R0)
000154  001020                        BNE     3$
000156  032710  040000                BIT     #40000,(R0)                     ;                               5339
000162  001415                        BEQ     3$
000164  111016                        MOVB    (R0),(SP)                       ;                               5341
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$:     MOV     R1,R0                   ;                               5343
000220  063700  000000G                 ADD     CST.ADDR,R0
000224  132710  000004                  BITB    #4,(R0)
000230  001420                          BEQ     4$
000232  032710  040000                  BIT     #40000,(R0)             ;                               5344
000236  001415                          BEQ     4$
000240  111016                          MOVB    (R0),(SP)               ;                               5346
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                   ;                               5348
000274  063700  000000G                 ADD     CST.ADDR,R0
000300  032710  040000                  BIT     #40000,(R0)
000304  001506                          BEQ     5$
000306  013700  000000G                 MOV     T.ADDR,R0               ;                               5353
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               ;                               5356
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               ;                               5359
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                  ;                       5350
000522  062702  000005          5$:     ADD     #5,R2                   ;*,DISP                 5334
000526  020227  000022                  CMP     R2,#22                  ;DISP,*
000532  003002                          BGT     6$
000534  000137  000612                  JMP     2$
000540  010316                  6$:     MOV     R3,(SP)                 ;CTLR,*                 5364
000542  012746  000056                  MOV     #56.,(SP)
000546  004737  000000G                 JSR     PC,BL$MUL
000552  005726                          TST     (SP)+
000554  005760  000002G                 TST     CST+2(R0)
000560  100026                          BPL     7$
000562  012716  000000G                 MOV     #RPT11,(SP)             ;                       5367
000566  012746  000001                  MOV     #1,-(SP)
000572  010600                          MOV     SP,R0                   ;SP,*
000574  104416                          TRAP    16
000576  010300                          MOV     R3,R0                   ;CTLR,*                 5368
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                  ;                       5366
000636  012716  000000G          7$:     MOV     #CRLF,(SP)              ;                       5371
000642  012746  000001                  MOV     #1,-(SP)
000646  010600                          MOV     SP,R0                   ;SP,*
000650  104416                          TRAP    16
000652  005726                          TST     (SP)+                   ;                       5331
000654  005203                          INC     R3                      ;CTLR                   5330
000656  000243                          .WORD CLV!CLC

000660  003002                          BGT     8$
000662  000137  000600'                 JMP     1$
000666  012716  000000G          8$:     MOV     #CRLF,(SP)              ;                       5375
000672  012746  000001                  MOV     #1,-(SP)
000676  010600                          MOV     SP,R0                   ;SP,*
000700  104416                          TRAP    16
000702  012716  000000G                 MOV     #RPT13,(SP)             ;                       5376
000706  012746  000001                  MOV     #1,-(SP)
000712  010600                          MOV     SP,R0                   ;SP,*
000714  104416                          TRAP    16
000716  012716  000000G                 MOV     #RPT14,(SP)             ;                       5377
000722  012746  000001                  MOV     #1,-(SP)
000726  010600                          MOV     SP,R0                   ;SP,*
000730  104416                          TRAP    16
000732  012716  000000G                 MOV     #RPT15,(SP)             ;                       5378
000736  012746  000001                  MOV     #1,-(SP)
000742  010600                          MOV     SP,R0                   ;SP,*
000744  104416                          TRAP    16
```

```
000746  005002                          CLR     R2                      ; CTLR                  5379
000750  010216                  9$:     MOV     R2,(SP)                 ; CTLR,*                5381
000752  004737  000000V                 JSR     PC,SET.CPAR
000756  012701  000003                  MOV     #3,R1                   ; *,DISK                5382
000762  010116                  10$:    MOV     R1,(SP)                 ; DISK,*                5384
000764  004737  000000V                 JSR     PC,SET.UPAR
000770  010100                          MOV     R1,R0                   ; DISK,*                5385
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               ;                       5389
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                5382
001076  020127  000022                  CMP     R1,#22                  ; DISK,*
001102  003727                          BLE     10$
001104  005202                          INC     R2                      ; CTLR                  5379
001106  000243                          .WORD CLV!CLC

001110  003717                          BLE     9$
001112  012716  000000G                 MOV     #CRLF,(SP)              ;                       5394
001116  012746  000001                  MOV     #1,-(SP)
001122  010600                          MOV     SP,R0                   ; SP,*
001124  104416                          TRAP    16
001126  062706  000030                  ADD     #30,SP                  ;                       5313
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                 ;                       5394
000004  104425                          TRAP    25
000006  000207                          RTS     PC

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

```
;          5397  %sbttl  INITIALIZE SECTION'
;          5398
;          5399  BGNINIT;
;          5400
;          5401  local
;          5402      DELAY_MULT : word,
;          5403      FLAG : byte,
;          5404      TEMP : word,
;          5405      HWPT_REF : ref block [HWPT LEN, word] field (HWP FIELDS),
;          5406      CLEAR_TABLES : byte;
;          5407
;          5408  SETPRI (PRIO7);                          ! PRIORITY 7   NO INTERRUPTS ALLOWED DURING INIT
;          5409
;          5410  if READEF (EF_NEW)                       ! IS THIS A NEW PASS?
;          5411  then
;          5412      begin
;          5413      ENTRY_REASON = NEW_PASS;
;          5414
;          5415      if (.SWP_FLAGS and SWF_CST) neq SWF_CST
;          5416      then
;          5417          CLEAR_TABLES = FALSE
;          5418      else
;          5419          CLEAR TABLES = TRUE;
;          5420
;          5421      end;
;          5422
;          5423  if READEF (EF_START)                     ! IS THIS A START?
;          5424  then
;          5425      begin
;          5426      BRESET;
;          5427      ENTRY_REASON = START;
;          5428      CLEAR_TABLES = TRUE;
;          5429      end;
;          5430
;          5431  if READEF (EF_RESTART)                   ! IS THIS A RESTART?
;          5432  then
;          5433      begin
;          5434      ENTRY_REASON = RESTART;
;          5435      CLEAR_TABLES = TRUE;
;          5436      end;
;          5437
;          5438  if READEF (EF_CONTINUE)                  ! IS THIS A CONTINUE?
;          5439  then
;          5440      begin
;          5441      ENTRY_REASON = CONT;
;          5442
;          5443      if (.SWP_FLAGS and SWF_CST) neq SWF_CST
;          5444      then
;          5445          CLEAR_TABLES = FALSE
;          5446      else
;          5447          CLEAR_TABLES = TRUE;
;          5448
;          5449      end;
;          5450
;          5451  if READEF (EF_PWR)                       ! ARE WE HERE BECAUSE OF POWER FAIL?
;          5452  then
```

```
;          5453      begin
;          5454      ENTRY_REASON = PWR_FAIL;
;          5455      CLEAR_TABLES = TRUE;
;          5456      PRINTF (MSG_01);                        ! "POWER DELAY   WAITING"
;          5457
;          5458      incr COUNT from 0 to 60 do              ! WAIT APPROX. 60 SECONDS
;          5459          begin
;          5460          DELAY_MULT = 333;
;          5461          DELAY (.DELAY_MULT);
;          5462          BREAK;                              ! BREAK FOR ACT
;          5463          end;
;          5464
;          5465      end;
;          5466
;          5467  SETVEC (0_TVEC, 0_BRK, PRI07);             ! SET ODT TRAP VECTOR !JSD REV A
;          5468
;          5469  !+
;          5470  !       MAKE SURE THAT NOT MORE THAN MAX_UNITS HAVE BEEN SPECIFIED.
;          5471  !       IF THERE ARE TOO MANY, NOTIFY USER AND RETURN TO SUPERVISOR.
;          5472  !       (DIAGNOSTIC IS ABORTED).
;          5473  !
;          5474
;          5475  if .L$UNIT gtru MAX_UNITS
;          5476  then
;          5477      begin
;          5478      ERRSF (1, EGS_01, EMS_01);
;          5479      DOCLN;
;          5480      end;
;          5481
;          5482  !+
;          5483  !       THE FOLLOWING CODE IS EXECUTED FOR ALL ENTRY REASONS EXCEPT NEW_PASS.
;          5484  !       ALL RUN-TIME CONTROLLER STATUS TABLES (CSTs) ARE CLEARED TO 0, THEN
;          5485  !       LOADED WITH CONFIGURATION DATA FROM THE HARDWARE P-TABLES.
;          5486  !
;          5487
;          5488  if (.ENTRY_REASON neq NEW_PASS)
;          5489  then
;          5490      begin
;          5491
;          5492      incr COUNT from 0 to ((MAX_CTLR * CST_LEN * 2)   2) by 2 do
;          5493          (CST + .COUNT) = 0;
;          5494
;          5495      incr UNIT from 0 to (.L$UNIT - 1) do        ! LOOP THROUGH ALL UNITS
;          5496
;          5497          if GPHARD (.UNIT, HWPT_REF) neqa 0      ! IF HWP TABLE FOUND
;          5498          then
;          5499              begin
;          5500              FLAG = NOT_FOUND;
;          5501
;          5502              incr CTLR from 0 to (MAX_CTLR - 1) do       ! LOOP THROUGH ALL CSTs
;          5503
;          5504                  if .CST [.CTLR, IP_ADDR] eqla .HWPT REF [HWP_IP ADDR]
;          5505                  then
;          5506
;          5507                      if .CST [.CTLR, .HWPT REF [HWP DISK_NUM] * UNIT SIZE + OF UN, D PRES] eql NOT_PRESENT
;          5508                      then
```

```
;       5509                        begin                                                   ! IF EMPTY SLOT FOUND
;       5510                        TEMP = .HWPT_REF [HWP_DISK_NUM] * UNIT_SIZE + OF_UN;
;       5511                        CST [.CTLR, .TEMP, D_ALL] = .HWPT_REF [HWP_DISK];       ! COPY DISK ADDR AND PROT BIT
;       5512                        CST [.CTLR, .TEMP, D_UNIT] = .UNIT;
;       5513                        CST [.CTLR, .TEMP, D_FATAL] = FALSE;
;       5514                        CST [.CTLR, .TEMP, D_PRES] = PRESENT;
;       5515                        CST [.CTLR, .TEMP + 1, D_BEG] = .HWPT_REF [HWP_BEG_TRK];
;       5516                        CST [.CTLR, .TEMP + 2, D_END] = .HWPT_REF [HWP_END_TRK];
;       5517                        CST [.CTLR, .TEMP + 3, D_DBN] = 0;
;       5518                        CST [.CTLR, .TEMP + 3, NODUPMEDIA] = NOT(.HWPT_REF [HWP_DUPEX]);
;       5519                        CST [.CTLR, .TEMP + 3, DUPWRITE] = (.HWPT_REF [HWP_DUPWT]);
;       5520                        CST [.CTLR, .TEMP + 4, D_COUNT] = 0;
;       5521
;       5522                        if (.CST [.CTLR, .TEMP, D_TYPE] eql RX_50) and
;       5523                           (.CST [.CTLR, .TEMP + 2, D_END] gtru RX_MAX_LBN)
;       5524                        then
;       5525                            CST [.CTLR, .TEMP + 2, D_END] = RX_MAX_LBN;
;       5526
;       5527                        FLAG = FOUND;
;       5528                        exitloop;
;       5529                        end
;       5530                    else
;       5531                        begin                       ! DUPLICATE UNIT
;       5532                        PRINTF (CER_01, .HWPT_REF [HWP_DISK_NUM], .HWPT_REF [HWP_IP_ADDR]);
;       5533                                                    ! "DUPLICATE UNIT: XX AT IP: XXXXXX"
;       5534                        DUR [.UNIT] = DU_CONF;      ! CONFIGURATION ERROR
;       5535                        DODU (.UNIT);               ! DROP UNIT
;       5536                        FLAG = FOUND;
;       5537                        exitloop;
;       5538                        end;
;       5539
;       5540                ;f .FLAG eql NOT_FOUND              ! IF NO IP ADDR MATCH TO EXISTING CST
;       5541                then
;       5542                    begin
;       5543
;       5544                    incr CTLR from 0 to (MAX_CTLR - 1) do   ! LOOP THROUGH EACH CST
;       5545
;       5546                        if .CST [.CTLR, IP_ADDR] eqla 0     ! IF EMPTY CST FOUND
;       5547                        then
;       5548                            begin
;       5549                            FLAG = FOUND;
;       5550                            CST [.CTLR, IP_ADDR] = .HWPT_REF [HWP_IP_ADDR];
;       5551                            CST [.CTLR, VEC_ADDR] = .HWPT_REF [HWP_VECTOR];
;       5552                            CST [.CTLR, BR_LEV] = .HWPT_REF [HWP_BR_LEVEL];
;       5553                            TEMP = .HWPT_REF [HWP_DISK_NUM] * UNIT_SIZE + OF_UN;
;       5554                            CST [.CTLR, .TEMP, D_ALL] = .HWPT_REF [HWP_DISK];       !COPY DISK ADDR AND PROT BIT
;       5555                            CST [.CTLR, .TEMP, D_UNIT] = .UNIT;
;       5556                            CST [.CTLR, .TEMP, D_FATAL] = FALSE;
;       5557                            CST [.CTLR, .TEMP, D_PRES] = PRESENT;
;       5558                            CST [.CTLR, .TEMP + 1, D_BEG] = .HWPT_REF [HWP_BEG_TRK];
;       5559                            CST [.CTLR, .TEMP + 2, D_END] = .HWPT_REF [HWP_END_TRK];
;       5560                            CST [.CTLR, .TEMP + 3, D_DBN] = 0;
;       5561                            CST [.CTLR, .TEMP + 3, NODUPMEDIA] = NOT(.HWPT_REF [HWP_DUPEX]);
;       5562                                    ! CHECK TO SEE IF PROGRAMMER WANTS TO NOT WRITE TO DBNs.
;       5563
;       5564                            CST [.CTLR, .TEMP + 3, DUPWRITE] = (.HWPT_REF [HWP_DUPWT]);
```

```
;          5565                        CST [.CTLR, .TEMP + 4, D_COUNT] = 0;
;          5566
;          5567                        if (.CST [.CTLR, .TEMP, D_TYPE] eql RX 50) and
;          5568                           (.CST [.CTLR, .TEMP + 2, D END] gtru RX MAX LBN)
;          5569                        then
;          5570                            CST [.CTLR, .TEMP + 2, D END] = RX MAX LBN;
;          5571
;          5572                        exitloop;
;          5573                        end;                        ! IF EMPTY CST FOUND
;          5574
;          5575                    if .FLAG eql NOT FOUND          ! IF NO FMPTY CST FOUND
;          5576                    then
;          5577                        begin
;          5578                        PRINTF (CER_02, MAX CTLR);  ! 'MORE THAN X DIFFERENT IP ADDRESSES.'
;          5579                        DUR [.UNIT] = DU_CONF;      ! CONFIGURATION ERROR
;          5580                        DODU (.UNIT);               ! DROP UNIT
;          5581                        end;
;          5582
;          5583                    end;                            ! IF NO IP ADDR MATCH IN CST
;          5584
;          5585                end;                                ! IF GPHARD RETURNS A HWP TABLE
;          5586        !
;          5587        ! CONFIGURATON CHECK FOR LEGAL RDRX UNIT MIX BECAUSE WE HAVE DIFFERENT
;          5588        ! DRIVES :  THE RD51, AND RX50.
;          5589        ! (NEEDED?)
;          5590        !
;          5591        end;                                        ! END OF "NON NEW_PASS" INITIALIZATION
;          5592
;          5593    if .ENTRY_REASON eql NEW_PASS
;          5594    then
;          5595        begin
;          5596
;          5597        incr UNIT from 0 to (.L$UNIT - 1) do
;          5598            GPHARD (.UNIT, HWPT_REF);                   ! DUMMY GPHARDs FOR NEW PASS
;          5599
;          5600        incr CTLR from 0 to (MAX_CTLR - 1) do
;          5601            begin
;          5602            CST [.CTLR, U_CNT] = 0;                     ! REINITIALIZE UNIT COUNT
;          5603
;          5604            incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;          5605                CST [.CTLR, .OFFSET, D_STAT] = OFFLINE; ! START EACH UNIT AS OFFLINE
;          5606
;          5607            end;
;          5608
;          5609        end;
;          5610
;          5611    if .ENTRY_REASON eql START
;          5612    then
;          5613        begin
;          5614        CTLR_CNT = 0;                               ! NUMBER OF CONFIGURED CONTROLLERS
;          5615
;          5616        incr CTLR from 0 to (MAX CTLR   1) do
;          5617
;          5618            if .CST [.CTLR, IP_ADDR] neqa 0         ! IF CONTROLLER IS PRESENT
;          5619            then
;          5620                CTLR_CNT = .CTLR_CNT + 1;           ! INCREMENT CONTROLLER COUNT
```

```
;          5621
;          5622      MEMORY (FREE MEM_ADDR);                    ! GET START OF FREE MEMORY
;          5623
;          5624      end;                                       ! END OF "START" INITIALIZATION
;          5625
;          5626  !+
;          5627  !       CLEAR STATISTICS TABLES
;          5628  !
.          5629
;          5630  incr UNITS from 0 to MAX_UNITS   1 do         ! CLEAR CURRENT STATISTICS
;          5631      incr COUNT from 0 to TALLY_CLEAR - 1 do
;          5632          TALLY [.UNITS * TALLY_LEN + .COUNT] = 0;
;          5633
;          5634  if .CLEAR_TABLES                              ! IF CLEAR TABLES ON EVERY PASS
;          5635  then
;          5636      incr UNITS from 0 to MAX_UNITS - 1 do
;          5637          incr COUNT from TALLY_CLEAR to TALLY_LEN   1 do        ! INITIALIZE TOTALS
;          5638              TALLY [.UNITS * TALLY_LEN + .COUNT] = 0;           !
;          5639
;          5640  if .CLEAR_TABLES
;          5641  then
;          5642      incr CTLR from 0 to MAX_CTLR - 1 do
;          5643          begin
;          5644          C_ERR_TBL [.CTLR, C_ERR_HRD] = 0;      ! INITALIZE CONTROLLER ERRORS
;          5645          C_ERR_TBL [.CTLR, C_ERR_SFT] = 0;      !
.          5646          end;
;          5647
;          5648  !+
;          5649  !       MISCELLANEOUS INITIALIZATON
;          5650  !-
;          5651
;          5652  incr CTLR from 0 to (MAX_CTLR - 1) do         ! INITIALIZE NO. OF OUTSTANDING QIOS
;          5653      QIO [.CTLR] = 0;
;          5654
;          5655  CRN_LOW = CRN_HIGH = 0;                       ! INITIALIZE COMMAND REF NUMBER
;          5656
;          5657  SETPRI (PRI00);                               ! SET PROGRAM PRIORITY TO 0
           5658
.          5659  ENDINIT;


                                    .GLOBL  L$DLY


                                    .SBTTL  LINIT INITIALIZE SECTION
000000  004137  000000G        LINIT:  JSR     R1,$SAVE5                       ;                       5396
000004  162706  000020                  SUB     #20,SP
000010  012700  000340                  MOV     #340,R0                         ;                       5408
000014  104441                          TRAP    41
000016  012700  000035                  MOV     #35,R0                          ;                       5410
000022  104447                          TRAP    47
000024  103012                          BHIS    2$
000026  112737  000005  000000G         MOVB    #5,ENTRY.REASON                 ;                       5413
000034  105737  000000G                 TSTB    SWP.FLAGS                       ;                       5415
000040  100402                          BMI     1$
000042  105005                          CLRB    R5                              ; CLEAR.TABLES          5417
```

```
000044  000402                          BR      2$                      ;                               5415
000046  112705  000001          1$:     MOVB    #1,R5                   ; *,CLEAR.TABLES                5419
000052  012700  000040          2$:     MOV     #40,R0                  ;                               5423
000056  104447                          TRAP    47
000060  103006                          BHIS    3$
000062  104433                          TRAP    33                      ;                               5425
000064  112737  000001  000000G        MOVB    #1,ENTRY.REASON         ;                               5427
000072  112705  000001                  MOVB    #1,R5                   ; *,CLEAR.TABLES                5428
000076  012700  000037          3$:     MOV     #37,R0                  ;                               5431
000102  104447                          TRAP    47
000104  103005                          BHIS    4$
000106  112737  000002  000000G        MOVB    #2,ENTRY.REASON         ;                               5434
000114  112705  000001                  MOVB    #1,R5                   ; *,CLEAR.TABLES                5435
000120  012700  000036          4$:     MOV     #36,R0                  ;                               5438
000124  104447                          TRAP    47
000126  103012                          BHIS    6$
000130  112737  000003  000000G        MOVB    #3,ENTRY.REASON         ;                               5441
000136  105737  000000G                TSTB    SWP.FLAGS               ;                               5443
000142  100402                          BMI     5$
000144  105005                          CLRB    R5                      ; CLEAR.TABLES                  5445
000146  000402                          BR      6$                      ;                               5443
000150  112705  000001          5$:     MOVB    #1,R5                   ; *,CLEAR.TABLES                5447
000154  012700  000034          6$:     MOV     #34,R0                  ;                               5451
000160  104447                          TRAP    47
000162  103036                          BHIS    12$
000164  112737  000004  000000G        MOVB    #4,ENTRY.REASON         ;                               5454
000172  112705  000001                  MOVB    #1,R5                   ; *,CLEAR.TABLES                5455
000176  012746  000000G                MOV     #MSG.01,-(SP)           ;                               5456
000202  012746  000001                  MOV     #1,-(SP)
000206  010600                          MOV     SP,R0                   ; SP,*
000210  104417                          TRAP    17
000212  012702  000075                  MOV     #75,R2                  ; *,COUNT                       5458
000216  012703  000515          7$:     MOV     #515,R3                 ; *,DELAY.MULT                  5460
000222  010301                          MOV     R3,R1                   ; DELAY.MULT,$$TMP2             5461
000224  001411                  8$:     BEQ     11$
000226  013700  000000G                MOV     L$DLY,R0                ; *,$$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$:    TRAP    22
000252  005302                          DEC     R2                      ; COUNT                         5458
000254  001360                          BNE     7$
000256  022626                          CMP     (SP)+,(SP)+             ;                               5453
000260  012746  000340          12$:    MOV     #340,-(SP)              ;                               5467
000264  012746  170000                  MOV     #-10000,-(SP)
000270  012746  000140                  MOV     #140,-(SP)
000274  012746  000003                  MOV     #3,-(SP)
000300  104437                          TRAP    37
000302  023727  000000G 000004          CMP     L$UNIT,#4               ;                               5475
000310  101405                          BLOS    13$
000312  104454                          TRAP    54                      ;                               5478
000314  000001                          .WORD   1
000316  000000G                         .WORD   EGS.01
```

```
000320  000000V                                 .WORD   EMS.01
000322  104444                                  TRAP    44
000324  123727  000000G 000005          13$:    CMPB    ENTRY.REASON,#5         ;                               5488
000332  001002                                  BNE     14$
000334  000137  003530'                          JMP     37$
000340  005000                          14$:    CLR     R0                      ; COUNT                         5492
000342  005060  000000G                 15$:    CLR     CST(R0)                 ; *(COUNT)                      5493
000346  062700  000002                          ADD     #2,R0                   ; *,COUNT                       5492
000352  020027  000054                          CMP     R0,#54                  ; COUNT,*
000356  003771                                  BLE     15$
000360  013766  000000G 000024                  MOV     L$UNIT,24(SP)           ;                               5495
000366  005066  000020                          CLR     20(SP)                  ; UNIT
000372  000137  003504'                          JMP     35$
000376  016600  000020                  16$:    MOV     20(SP),R0               ; UNIT,*                        5497
000402  104442                                  TRAP    42
000404  010066  000022                          MOV     R0,22(SP)               ; *,HWPT.REF
000410  001002                                  BNE     17$
000412  000137  003500'                          JMP     34$
000416  105066  000016                  17$:    CLRB    16(SP)                  ; FLAG                          5500
000422  005004                                  CLR     R4                      ; CTLR                          5502
000424  010416                          18$:    MOV     R4,(SP)                 ; CTLR,*                        5504
000426  012746  000056                          MOV     #56.,-(SP)
000432  004737  000000G                         JSR     PC,BL$MUL
000436  005726                                  TST     (SP)+
000440  026076  000000G 000022                  CMP     CST(R0),@22(SP)         ; *,HWPT.REF
000446  001402                                  BEQ     19$
000450  000137  002744'                          JMP     24$
000454  012766  000001  000012      19$:    MOV     #1,12(SP)               ;                               5527
000462  112766  000001  000016              MOVB    #1,16(SP)               ; *,FLAG
000470  012700  000006                          MOV     #6,R0                   ;                               5507
000474  066600  000022                          ADD     22(SP),R0               ; HWPT.REF,*
000500  010066  000014                          MOV     R0,14(SP)
000504  111016                                  MOVB    (R0),(SP)
000506  042716  177774                          BIC     #177774,(SP)
000512  012746  000005                          MOV     #5,-(SP)
000516  004737  000000G                         JSR     PC,BL$MUL
000522  010003                                  MOV     R0,R3
000524  005726                                  TST     (SP)+
000526  010416                                  MOV     R4,(SP)                 ; CTLR,*
000530  012746  000027                          MOV     #27.,-(SP)
000534  004737  000000G                         JSR     PC,BL$MUL
000540  010001                                  MOV     R0,R1
000542  005726                                  TST     (SP)+
000544  060300                                  ADD     R3,R0
000546  006300                                  ASL     R0
000550  032760  040000  000006G                BIT     #40000,CST+6(R0)
000556  001132                                  BNE     23$
000560  010302                                  MOV     R3,R2                   ; *,TEMP                        5510
000562  062702  000003                          ADD     #3,R2                   ; *,TEMP
000566  010100                                  MOV     R1,R0                   ;                               5511
000570  060200                                  ADD     R2,R0                   ; TEMP,*
000572  006300                                  ASL     R0
000574  012703  000000G                         MOV     #CST,R3
000600  060003                                  ADD     R0,R3
000602  017613  000014                          MOV     @14(SP),(R3)
000606  016600  000020                          MOV     20(SP),R0               ; UNIT,*                        5512
```

```
000612  000300                          SWAB    R0
000614  042700  170377                  BIC     #170377,R0
000620  042713  007400                  BIC     #7400,(R3)
000624  050013                          BIS     R0,(R3)
000626  042713  010000                  BIC     #10000,(R3)                     ;                               5513
000632  052713  040000                  BIS     #40000,(R3)                     ;                               5514
000636  010100                          MOV     R1,R0                           ;                               5515
000640  060200                          ADD     R2,R0                           ; TEMP,*
000642  006300                          ASL     R0
000644  016646  000022                  MOV     22(SP), (SP)                    ; HWPT.REF,*
000650  062716  000010                  ADD     #10,(SP)
000654  013660  000002G                 MOV     @(SP)+,CST+2(R0)
000660  010100                          MOV     R1,R0                           ;                               5516
000662  060200                          ADD     R2,R0                           ; TEMP,*
000664  006300                          ASL     R0
000666  012766  000004G 000010          MOV     #CST+4,10(SP)
000674  060066  000010                  ADD     R0,10(SP)
000700  016600  000022                  MOV     22(SP),R0                       ; HWPT.REF,*
000704  016076  000012  000010          MOV     12(R0),@10(SP)                  ; *(HWPT.REF),*
000712  010100                          MOV     R1,R0                           ;                               5517
000714  060200                          ADD     R2,R0                           ; TEMP,*
000716  006300                          ASL     R0
000720  062700  000006G                 ADD     #CST+6,R0
000724  105010                          CLRB    (R0)
000726  005046                          CLR     -(SP)                           ;                               5518
000730  132776  000010  000016          BITB    #10,@16(SP)
000736  001401                          BEQ     20$
000740  005216                          INC     (SP)
000742  005116                  20$:    COM     (SP)
000744  011646                          MOV     (SP),-(SP)
000746  042710  100000                  BIC     #100000,(R0)
000752  006026                          ROR     (SP)+
000754  103002                          BCC     21$
000756  052710  100000                  BIS     #100000,(R0)
000762  117616  000016          21$:    MOVB    @16(SP),(SP)                    ;                               5519
000766  042710  010000                  BIC     #10000,(R0)
000772  032726  000020                  BIT     #20,(SP)+
000776  001402                          BEQ     22$
001000  052710  010000                  BIS     #10000,(R0)
001004  010100                  22$:    MOV     R1,R0                           ;                               5520
001006  060200                          ADD     R2,R0                           ; TEMP,*
001010  006300                          ASL     R0
001012  005060  000010G                 CLR     CST+10(R0)
001016  132713  000004                  BITB    #4,(R3)                         ;                               5522
001022  001042                          BNE     25$
001024  027627  000010  001437          CMP     @10(SP),#1437                   ;                               5523
001032  101436                          BLOS    25$
001034  012776  001437  000010          MOV     #1437,@10(SP)                   ;                               5525
001042  000432                          BR      25$                             ;                               5509
001044  017616  000022          23$:    MOV     @22(SP),(SP)                    ; HWPT.REF,*                    5532
001050  117646  000014                  MOVB    @14(SP),-(SP)
001054  042716  177774                  BIC     #177774,(SP)
001060  012746  000000G                 MOV     #CER.01,-(SP)
001064  012746  000003                  MOV     #3,-(SP)
001070  010600                          MOV     SP,R0                           ; SP,*
001072  104417                          TRAP    17
```

```
0C1074  062706  000006                    ADD     #6,SP
001100  016600  000020                    MOV     20(SP),R0           ; UNIT,*                    5534
001104  112760  000001  000000G           MOVB    #1,DUR(R0)          ; *,*(UNIT)
001112  104451                            TRAP    51                  ;                           5535
001114  000405                            BR      25$                 ;                           5531
001116  005204                    24$:    INC     R4                  ; CTLR                      5502
001120  000243                            .WORD CLV'CLC

001122  003002                            BGT     25$
001124  000137  002252'                    JMP     18$
001130  105766  000016                25$:TSTB    16(SP)              ; FLAG                      5540
001134  001402                            BEQ     26$
001136  000137  003500'                    JMP     34$
001142  005004                    26$:    CLR     R4                  ; CTLR                      5544
001144  010416                    27$:    MOV     R4,(SP)             ; CTLR,*                    5546
001146  012746  000056                    MOV     #56.,-(SP)
001152  004737  000000G                   JSR     PC,BL$MUL
001156  005726                            TST     (SP)+
001160  005760  000000G                   TST     CST(R0)
001164  001402                            BEQ     28$
001166  000137  003422'                    JMP     32$
001172  112766  000001  000016    28$:    MOVB    #1,16(SP)           ; *,FLAG                    5549
001200  017660  000022  000000G           MOV     @22(SP),CST(R0)     ; HWPT.REF,*                5550
001206  016603  000022                    MOV     22(SP),R3           ; HWPT.REF,*                5551
001212  016301  000002                    MOV     2(R3),R1            ; *(HWPT.REF),*
001216  042701  177000                    BIC     #177000,R1
001222  042760  000777  000002G           BIC     #777,CST+2(R0)
001230  050160  000002G                   BIS     R1,CST+2(R0)
001234  010301                            MOV     R3,R1               ; HWPT.REF,*                5552
001236  116160  000004  000004G           MOVB    4(R1),CST+4(R0)     ; *(HWPT.REF),*
001244  012700  000006                    MOV     #6,R0               ;                           5553
001250  060300                            ADD     R3,R0               ; HWPT.REF,*
001252  010066  000014                    MOV     R0,14(SP)
001256  111016                            MOVB    (R0),(SP)
001260  042716  177774                    BIC     #177774,(SP)
001264  012746  000005                    MOV     #5.,-(SP)
001270  004737  000000G                   JSR     PC,BL$MUL
001274  005726                            TST     (SP)+
001276  010002                            MOV     R0,R2               ; *,TEMP
001300  062702  000003                    ADD     #3,R2               ; *,TEMP
001304  010416                            MOV     R4,(SP)             ; CTLR,*                    5554
001306  012746  000027                    MOV     #27.,-(SP)
001312  004737  000000G                   JSR     PC,BL$MUL
001316  010001                            MOV     R0,R1
001320  060200                            ADD     R2,R0               ; TEMP,*
001322  006300                            ASL     R0
001324  012703  000000G                   MOV     #CST,R3
001330  060003                            ADD     R0,R3
001332  017613  000016                    MOV     @16(SP),(R3)
001336  016600  000022                    MOV     22(SP),R0           ; UNIT,*                    5555
001342  000300                            SWAB    R0
001344  042700  170377                    BIC     #170377,R0
001350  042713  007400                    BIC     #7400,(R3)
001354  050013                            BIS     R0,(R3)
001356  042713  010000                    BIC     #10000,(R3)         ;                           5556
001362  052713  040000                    BIS     #40000,(R3)         ;                           5557
```

```
001360  010100                          MOV     R1,R0                   ;                               5559
001370  060200                          ADD     R2,R0                   ; TEMP,*
001372  006300                          ASL     R0
001374  016616  000024                  MOV     24(SP),(SP)             ; MUPT.REF,*
001400  062716  000010                  ADD     #10.,(SP)
001404  013660  000020                  MOV     @(SP)+,CST+2(R0)
001410  010100                          MOV     R1,R0                   ;                               5559
001412  060200                          ADD     R2,R0                   ; TEMP,*
001414  006300                          ASL     R0
001416  012766  000040  000012          MOV     #CST+4,12(SP)
001424  060066  000012                  ADD     R0,12(SP)
001430  016600  000022                  MOV     22(SP),R0               ; MUPT.REF,*
001434  016076  000012  000012          MOV     12(R0),@12(SP)          ; *(MUPT.REF),*
001442  010100                          MOV     R1,R0                   ;                               5560
001444  060200                          ADD     R2,R0                   ; TEMP,*
001446  006300                          ASL     R0
001450  062700  000006                  ADD     #CST+6,R0
001454  105010                          CLRB    (R0)
001456  005046                          CLR     -(SP)                   ;                               5561
001460  132776  000010  000016          BITB    #10,@16(SP)
001466  001401                          BEQ     29$
001470  005216                          INC     (SP)
001472  005116                  29$:    COM     (SP)
001474  011646                          MOV     (SP),-(SP)
001476  042710  100000                  BIC     #100000,(R0)
001502  006026                          ROR     (SP)+
001504  103002                          BCC     30$
001506  052710  100000                  BIS     #100000,(R0)
001512  117616  000016          30$:    MOVB    @16(SP),(SP)            ;                               5564
001516  042710  010000                  BIC     #10000,(R0)
001522  032726  000020                  BIT     #20,(SP)+
001526  001402                          BEQ     31$
001530  052710  010000                  BIS     #10000,(R0)
001534  010100                  31$:    MOV     R1,R0                   ;                               5565
001536  060200                          ADD     R2,R0                   ; TEMP,*
001540  006300                          ASL     R0
001542  005060  000010                  CLR     CST+10(R0)
001546  132713  000004                  BITB    #4,(R3)                 ;                               5567
001552  001015                          BNE     33$
001554  027627  000012  001437          CMP     @12(SP),#1437           ;                               5568
001562  101411                          BLOS    33$
001564  012776  001437  000012          MOV     #1437,@12(SP)           ;                               5570
001572  000405                          BR      33$                     ;                               5548
001574  005204                  32$:    INC     R4                      ; CTLR                          5544
001576  000243                          .WORD CLV!CLC

001600  003002                          BGT     33$
001602  000137  002772                  JMP     27$
001606  105766  000016          33$:    TSTB    16(SP)                  ; FLAG                          5575
001612  001017                          BNE     34$
001614  012716  000001                  MOV     #1,(SP)                 ;                               5578
001620  012746  000000                  MOV     #CER.02,-(SP)
001624  012746  000002                  MOV     #2,-(SP)
001630  010600                          MOV     SP,R0                   ; SP,*
001632  104417                          TRAP    17
001634  016600  000024                  MOV     24(SP),R0               ; UNIT,*                        5579
```

```
001640  112760  000001  000000G          MOVB    #1,DUR(R0)              ; *,*(UNIT)
001646  104451                           TRAP    51                      ;
001650  022626                           CMP     (SP)+,(SP)+             ;
001652  005266  000020                34$: INC     20(SP)                  ; UNIT                          5495
001656  026666  000020  000024       35$: CMP     20(SP),24(SP)           ; UNIT,*
001664  002002                           BGE     36$
001666  000137  002224'                  JMP     16$
001672  123727  000000G 000005       36$: CMPB    ENTRY.REASON,#5         ;                               5593
001700  001051                           BNE     42$
001702  013701  000000G              37$: MOV     L$UNIT,R1               ;                               5597
001706  005002                           CLR     R2                      ; UNIT
001710  000405                           BR      39$
001712  010200                       38$: MOV     R2,R0                   ; UNIT,*                        5598
001714  104442                           TRAP    42
001716  010066  000022                   MOV     R0,22(SP)               ; *,HWPT.REF
001722  005202                           INC     R2                      ; UNIT                          5597
001724  020201                       39$: CMP     R2,R1                   ; UNIT,*
001726  002771                           BLT     38$
001730  005004                           CLR     R4                      ; CTLR                          5600
001732  010416                       40$: MOV     R4,(SP)                 ; CTLR,*                        5602
001734  012746  000056                   MOV     #56,-(SP)
001740  004737  000000G                  JSR     PC,BL$MUL
001744  105060  000005G                  CLRB    CST+5(R0)
001750  010416                           MOV     R4,(SP)                 ; CTLR,*                        5605
001752  012746  000027                   MOV     #27,-(SP)
001756  004737  000000G                  JSR     PC,BL$MUL
001762  012702  000003                   MOV     #3,R2                   ; *,OFFSET                      5604
001766  010001                       41$: MOV     R0,R1                   ;                               5605
001770  060201                           ADD     R2,R1                   ; OFFSET,*
001772  006301                           ASL     R1
001774  042761  020000  000000G          BIC     #20000,CST(R1)
002002  062702  000005                   ADD     #5,R2                   ; *,OFFSET                      5604
002006  020227  000022                   CMP     R2,#22                  ; OFFSET,*
002012  003765                           BLE     41$
002014  022626                           CMP     (SP)+,(SP)+             ;                               5601
002016  005204                           INC     R4                      ; CTLR                          5600
002020  000243                           .WORD CLV!CLC

002022  003743                           BLE     40$
002024  123727  000000G 000001       42$: CMPB    ENTRY.REASON,#1         ;                               5611
002032  001017                           BNE     45$
002034  005037  000000G                  CLR     CTLR.CNT                ;                               5614
002040  005000                           CLR     R0                      ; CTLR                          5616
002042  005760  000000G              43$: TST     CST(R0)                 ; *(CTLR)                       5618
002046  001402                           BEQ     44$
002050  005237  000000G                  INC     CTLR.CNT                ;                               5620
002054  062700  000056              44$: ADD     #56,R0                  ; *,CTLR                        5616
002060  000243                           .WORD CLV!CLC

002062  003767                           BLE     43$
002064  104431                           TRAP    31                      ;                               5622
002066  010037  000000G                  MOV     R0,FREE.MEM.ADDR
002072  005002                       45$: CLR     R2                      ; UNITS                         5630
002074  005001                       46$: CLR     R1                      ; COUNT                         5631
002076  010100                       47$: MOV     R1,R0                   ; COUNT,*                       5632
002100  060200                           ADD     R2,R0                   ; UNITS,*
```

```
002102  006300                          ASL     R0
002104  005060  000000G                 CLR     TALLY(R0)
002110  005201                          INC     R1                      ; COUNT                 5631
002112  020127  000006                  CMP     R1,#6                   ; COUNT,*
002116  003767                          BLE     47$
002120  062702  000034                  ADD     #34,R2                  ; *,UNITS               5630
002124  020227  000124                  CMP     R2,#124                 ; UNITS,*
002130  003761                          BLE     46$
002132  032705  000001                  BIT     #1,R5                   ; *,CLEAR.TABLES        5634
002136  001421                          BEQ     50$
002140  005002                          CLR     R2                      ; UNITS                 5636
002142  012701  000007          48$:    MOV     #7,R1                   ; *,COUNT               5637
002146  010100          49$:    MOV     R1,R0                   ; COUNT,*               5638
002150  060200                          ADD     R2,R0                   ; UNITS,*
002152  006300                          ASL     R0
002154  005060  000000G                 CLR     TALLY(R0)
002160  005201                          INC     R1                      ; COUNT                 5637
002162  020127  000033                  CMP     R1,#33                  ; COUNT,*
002166  003767                          BLE     49$
002170  062702  000034                  ADD     #34,R2                  ; *,UNITS               5636
002174  020227  000124                  CMP     R2,#124                 ; UNITS,*
002200  003760                          BLE     48$
002202  006005          50$:    ROR     R5                      ; CLEAR.TABLES          5640
002204  103011                          BCC     52$
002206  005000                          CLR     R0                      ; CTLR                  5642
002210  105060  000000G         51$:    CLRB    C.ERR.TBL(R0)           ; *(CTLR)               5644
002214  105060  000001G                 CLRB    C.ERR.TBL+1(R0)         ; *(CTLR)               5645
002220  062700  000002                  ADD     #2,R0                   ; *,CTLR                5642
002224  000243                          .WORD CLV!CLC

002226  003770                          BLE     51$
002230  005000          52$:    CLR     R0                      ; CTLR                  5652
002232  105060  000000G         53$:    CLRB    QIO(R0)                 ; *(CTLR)               5653
002236  005200                          INC     R0                      ; CTLR                  5652
002240  000243                          .WORD CLV!CLC

002242  003773                          BLE     53$
002244  005037  000000G                 CLR     CRN.HIGH                ;                       5655
002250  005037  000000G                 CLR     CRN.LOW
002254  005000                          CLR     R0                      ;                       5657
002256  104441                          TRAP    41
002260  062706  000030                  ADD     #30,SP                  ;                       5396
002264  000207                          RTS     PC

; Routine Size:  603 words,     Routine Base:  $CODE$ + 1626
; Maximum stack depth per invocation:  23 words

                                        .SBTTL  L$INIT INITIALIZE SECTION
000000  004737  001626'         L$INIT::JSR     PC,LINIT                ;                       5657
000004  104411                          TRAP    11
000006  000207                          RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4114
; Maximum stack depth per invocation:  2 words
```

```
;          5660  $sbttl  AUTODROP SECTION'
;          5661
;          5662  !+
;          5663  ! THIS CODE IS EXECUTED IMMEDIATELY AFTER THE INITIALIZE CODE IF
;          5664  ! THE 'ADR" FLAG WAS SET.  THE UNIT(S) UNDER TEST ARE CHECKED TO
;          5665  ! SEE IF THEY WILL RESPOND.  THOSE THAT DON'T ARE IMMEDIATELY
;          5666  ! DROPPED FROM TESTING.
;          5667  !
;          5668
;          5669  BGNAUTO;
;          5670
;          5671  !if (.SWP_FLAGS and SWF_TRC) eql SWF_TRC
;          5672  !then   PRINTF (DBM3);
;          5673
;          5674  return;          ! ADDED TO PREVENT COMPILATION WARNING  !JSD REV A
;          5675
;          5676  ENDAUTO;


                                        .SBTTL  LAUTO AUTODROP SECTION
000000  000207                  LAUTO:  RTS     PC                              ;  5659

; Routine Size:  1 word,        Routine Base:  $CODE$ + 4124
; Maximum stack depth per invocation:  0 words



                                        .SBTTL  L$AUTO AUTODROP SECTION
000000  004737  004124'         L$AUTO::JSR     PC,LAUTO                        ;  5674
000004  104461                          TRAP    61
000006  000207                          RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4126
; Maximum stack depth per invocation:  2 words
```

```
;         5677  %sbttl  CLEANUP CODING SECTION
;         5678
;         5679  !+
;         5680  ! THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
;         5681  ! AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.
;         5682  !
;         5683
.         5684  BGNCLN;
.         5685
;         5686  DORPT;
.         5687
;         5688  !CLRVEC (O_TVEC);                              ! RETURN ODT TRAP TO DIAGNOSTIC SUPERVISER
.         5689
.         5690  incr CTLR from 0 to (MAX_CTLR - 1) do          ! FOR EACH CONTROLLER
.         5691
.         5692      if (RDRX_ADDR = .CST [.CTLR, IP_ADDR]) neqa 0       ! IF CONTROLLER EXISTS
;         5693      then
.         5694          begin
;         5695          CLRVEC (.CST[.CTLR, VEC_ADDR]);         ! RETURN CONTROLLER'S TRAP VECTOR TO SUPERVISOR
;         5696          WRT_RDRX (RCIP, RC_ALL, ALL_ONES);      ! WRITE IP TO STOP DEVICE
;         5697          end;
;         5698
.         5699  ENDCLN;

                                        .SBTTL  LCLEAN CLEANUP CODING SECTION
000000  010146                  LCLEAN: MOV     R1, (SP)                        ;                               5676
000002  104424                          TRAP    24                              ;                               5684
000004  005001                          CLR     R1                              ; CTLR                          5690
000006  016137  000000G 000000G  1$:     MOV     CST(R1),RDRX.ADDR               ; *(CTLR),*                     5692
000014  001411                          BEQ     2$
000016  016100  000002G                 MOV     CST+2(R1),R0                    ; *(CTLR),*                     5695
000022  042700  177000                  BIC     #177000,R0
000026  104436                          TRAP    36
000030  012700  177777                  MOV     #-1,R0                          ; *,RC.REG                      5696
000034  010077  000000G                 MOV     R0,@RDRX.ADDR                   ; RC.REG,*
000040  062701  000056          2$:     ADD     #56,R1                          ; *,CTLR                        5690
000044  000243                          .WORD CLV!CLC

000046  003757                          BLE     1$
000050  012601                          MOV     (SP)+,R1                        ;                               5676
000052  000207                          RTS     PC

; Routine Size:  22 words,      Routine Base:  $CODE$ + 4136
; Maximum stack depth per invocation:  3 words

                                        .SBTTL  L$CLEAN CLEANUP CODING SECTION
000000  004737  004136'         L$CLEAN::
                                        JSR     PC,LCLEAN                       ;                               5697
000004  104412                          TRAP    12
000006  000207                          RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4212
; Maximum stack depth per invocation:  2 words
```

```
;          5700  %sbttl  DROP UNIT SECTION'
;          5701
;          5702  !+
;          5703  ! THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
;          5704  ! TO NO LONGER BE TESTED.
;          5705  !-
;          5706
;          5707  BGNDU;
;          5708
;          5709  local
;          5710      UNIT : word,                                    ! NUMBER OF UNIT BEING DROPPED
;          5711      PRINT : byte initial (byte (FALSE));            ! NO PRINTING OF DROP UNIT MESSAGE
;          5712
;          5713  label
;          5714      SEARCH;
;          5715
;          5716  begin
;          5717
;          5718  register
;          5719      INPUT = 0;                                      ! UNIT NUMBER APPEARS IN R0 UPON ENTRY
;          5720
;          5721  UNIT = .INPUT;                                      ! GET UNIT NUMBER
;          5722  end;                                                ! UNDECLARE REGISTER
;          5723
;          5724  !if (.SWP_FLAGS and SWF_TRC) eql SWF_TRC            !JSD REV A
;          5725  !then                                               !JSD REV A
;          5726  !    PRINTF (DBM5, .UNIT);                          !JSD REV A
;          5727
;          5728  SEARCH :                                            ! BEGIN SEARCH BLOCK
;          5729  begin
;          5730
;          5731  incr CTLR from 0 to (MAX_CTLR - 1) do               ! FOR EACH CONTROLLER
;          5732
;          5733      incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do     ! FOR EACH UNIT ENTRY IN CST
;          5734
;          5735          if (.CST [.CTLR, .OFFSET, D_UNIT] eql .UNIT) and                        ! IF UNIT MATCHES CST ENTRY
;          5736             (.CST [.CTLR, .OFFSET, D_PRES] eql PRESENT)
;          5737          then
;          5738              begin
;          5739
;          5740              if (.CST [.CTLR, .OFFSET, D_STAT] eql ONLINE) or    ! IF UNIT IS STILL ALIVE
;          5741                 (.DUR [.UNIT] eql DU_ONLINE) or
;          5742                 (.DUR [.UNIT] eql DU_PROTECT)
;          5743              then
;          5744                  begin
;          5745                  PRINT = TRUE;                                   ! O.K. TO PRINT MESSAGE
;          5746
;          5747                  if (.CST [.CTLR, U_CNT] gtru 0) and
;          5748                     (.CST [.CTLR, .OFFSET, D_STAT] eql ONLINE)
;          5749                  then
;          5750                      CST [.CTLR, U_CNT] = .CST [.CTLR, U_CNT] - 1;! DECREMENT UNIT COUNT
;          5751
;          5752                  if (.CST [.CTLR, U_CNT] eql 0) and
;          5753                     (.CST [.CTLR, .OFFSET, D_STAT] eql ONLINE)
;          5754                  then
;          5755                      EOP_FLAG = TRUE;                            ! DECLARE END OF PASS IF ALL UNITS OFFLINE
```

```
;          5756
;          5757                CST [.CTLR, .OFFSET, D STAT] = OFFLINE;          ! MARK UNIT OFFLINE
;          5758                end;                                             ! IF UNIT WAS STILL ALIVE
;          5759
;          5760              leave SEARCH;                                      ! EXIT SEARCH BLOCK
;          5761              end;                                               ! IF UNIT FOUND IN CST
;          5762
;          5763  end;
;          5764
;          5765  if .PRINT or                                                   ! IF OK TO PRINT MESSAGE
;          5766     (.DUR [.UNIT] eql DU_CONF) or
;          5767     (.DUR [.UNIT] eql DU_INIT) or
;          5768     (.DUR [.UNIT] eql DU_ONLINE) or
;          5769     (.DUR [.UNIT] eql DU_AV) or
;          5770     (.DUR [.UNIT] eql DU_PROTECT)
;          5771  then
;          5772      begin
;          5773      PRINTF (DU_MSG, .UNIT);                                    ! "UNIT XX DROPPED
;          5774      PRINTF (.DU_RSN [.DUR [.UNIT]]);                           ! REASON
;          5775      end;
;          5776
;          5777  ENDDU;


                                        .SBTTL  LDU DROP UNIT SECTION
000000  004137  000000G          LDU:  JSR     R1,$SAVE5                        ;                                  5699
000004  024646                        CMP     -(SP),-(SP)
000006  105005                        CLRB    R5                               ; PRINT
000010  010001                        MOV     R0,R1                            ; INPUT.UNIT                       5721
000012  005066  000002                CLR     2(SP)                            ; CTLR                             5731
000016  016646  000002          1$:   MOV     2(SP),-(SP)                      ; CTLR,*                           5735
000022  012746  000027                MOV     #27.,-(SP)
000026  004737  000000G               JSR     PC,BL$MUL
000032  010066  000004                MOV     R0,4(SP)
000036  012703  000003                MOV     #3,R3                            ; *,OFFSET                         5733
000042  010300                  2$:   MOV     R3,R0                            ; OFFSET,*                         5735
000044  066600  000004                ADD     4(SP),R0
000050  006300                        ASL     R0
000052  012704  000000G               MOV     #CST,R4
000056  060004                        ADD     R0,R4
000060  010102                        MOV     R1,R2                            ; UNIT,*
000062  011400                        MOV     (R4),R0
000064  000300                        SWAB    R0
000066  042700  177760                BIC     #177760,R0
000072  020002                        CMP     R0,R2
000074  001055                        BNE     8$
000076  032714  040000                BIT     #40000,(R4)                      ;                                  5736
000102  001452                        BEQ     8$
000104  005002                        CLR     R2                               ;                                  5740
000106  032714  020000                BIT     #20000,(R4)
000112  001402                        BEQ     3$
000114  005202                        INC     R2
000116  000410                        BR      4$
000120  126127  000000G 000007  3$:   CMPB    DUR(R1),#7                       ; *(UNIT),*                        5741
000126  001404                        BEQ     4$
000130  126127  000000G 000011        CMPB    DUR(R1),#11                      ; *(UNIT),*                        5742
```

```
000136  001032                          BNE     7$
000140  112705  000001          4$:     MOVB    #1,R5                   ; *,PRINT                       5745
000144  016616  000006                  MOV     6(SP),(SP)              ; CTLR,*                        5747
000150  012746  000056                  MOV     #56, (SP)
000154  004737  000000G                 JSR     PC,BL$MUL
000160  005726                          TST     (SP)+
000162  062700  000004G                 ADD     #CST+4,R0
000166  105760  000001                  TSTB    1(R0)
000172  001404                          BEQ     5$
000174  006002                          ROR     R2                      ;                               5748
000176  105660  000001                  SBCB    1(R0)                   ;                               5750
000202  001006                          BNE     6$                      ;                               5752
000204  032714  020000          5$:     BIT     #20000,(R4)             ;                               5753
000210  001403                          BEQ     6$
000212  112737  000001  000000G         MOVB    #1,EOP.FLAG             ;                               5755
000220  042714  020000          6$:     BIC     #20000,(R4)             ;                               5757
000224  022626                  7$:     CMP     (SP)+,(SP)+             ;                               5738
000226  000412                          BR      9$
000230  062703  000005          8$:     ADD     #5,R3                   ; *,OFFSET                      5733
000234  020327  000022                  CMP     R3,#22                  ; OFFSET,*
000240  003700                          BLE     2$
000242  022626                          CMP     (SP)+,(SP)+
000244  005266  000002                  INC     2(SP)                   ; CTLR                          5731
000250  000243                          .WORD CLV!CLC

000252  003661                          BLE     1$
000254  006005          9$:     ROR     R5                              ; PRINT                         5765
000256  103424                          BLO     10$
000260  126127  000000G 000001          CMPB    DUR(R1),#1              ; *(UNIT),*                     5766
000266  001420                          BEQ     10$
000270  126127  000000G 000002          CMPB    DUR(R1),#2              ; *(UNIT),*                     5767
000276  001414                          BEQ     10$
000300  126127  000000G 000007          CMPB    DUR(R1),#7              ; *(UNIT),*                     5768
000306  001410                          BEQ     10$
000310  126127  000000G 000013          CMPB    DUR(R1),#13             ; *(UNIT),*                     5769
000316  001404                          BEQ     10$
000320  126127  000000G 000011          CMPB    DUR(R1),#11             ; *(UNIT),*                     5770
000326  001024                          BNE     11$
```

```
000330  010146                          10$:    MOV     R1, (SP)                        ; UNIT,*                                5773
000332  012746  000000G                        MOV     #DU.MSG, (SP)
000336  012746  000002                        MOV     #2, (SP)
000342  010600                                MOV     SP,R0                           ; SP,*
000344  104417                                TRAP    17
000346  116101  000000G                        MOVB    DUR(R1),R1                      ; *(UNIT),*                             5774
000352  042701  177400                        BIC     #177400,R1
000356  006301                                ASL     R1
000360  016116  000000G                        MOV     DU.RSN(R1),(SP)
000364  012746  000001                        MOV     #1,-(SP)
000370  010600                                MOV     SP,R0                           ; SP,*
000372  104417                                TRAP    17
000374  062706  000010                        ADD     #10,SP                          ;                                       5772
000400  022626                          11$:    CMP     (SP)+,(SP)+                     ;                                       5699
000402  000207                                RTS     PC

; Routine Size:  130 words,     Routine Base:  $CODE$ + 4222
; Maximum stack depth per invocation:  14 words


                                        .SBTTL  L$DU DROP UNIT SECTION
000000  004737  004222'                  L$DU::  JSR     PC,LDU                          ;                                       5775
000004  104453                                TRAP    53
000006  000207                                RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4626
; Maximum stack depth per invocation:  2 words
```

```
;   %sbttl 'ADD UNIT SECTION'
;
;   !+
;   ! THE ADD-UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
;   !  TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
;   !  TO THE TEST CYCLE.
;   !
;
;   BGNAU;
;
;   local
;       STINDX : word,
;       ENDIDX : word;
;
;   register
;       UNIT = 0;                               ! UNIT NUMBER APPEARS IN RO UPON ENTRY
;
;   if (.SWP_FLAGS and SWF_CST) eql SWF CST
;   then
;       begin                                   ! IF CLEAR STAT. TABLES TRUE....
;       STINDX = .UNIT * TALLY_LEN;             ! ZERO OUT
;       ENDIDX = .STINDX + TALLY_LEN - 1;       !    ADDED
;
;       incr COUNT from .STINDX to .ENDIDX do   !    UNIT'S
;           TALLY [.COUNT] = 0;                 !    STATISTICS
;
;       end;
;
;   ENDAU;
```

```
                                        .SBTTL  LAU ADD UNIT SECTION
000000  004137  000000G        LAU:    JSR     R1,$SAVE2                       ;                               5777
000004  105737  000000G                TSTB    SWP.FLAGS                       ;                               5795
000010  100023                        BPL     3$
000012  010046                        MOV     R0, (SP)                        ; UNIT,*                        5798
000014  012746  000034                MOV     #34, (SP)
000020  004737  000000G                JSR     PC,BL$MUL
000024  010002                        MOV     R0,R2                           ; STINDX,ENDIDX                 5799
000026  062702  000033                ADD     #33,R2                          ; *,ENDIDX
000032  010001                        MOV     R0,R1                           ; STINDX,COUNT                  5801
000034  005301                        DEC     R1                              ; COUNT
000036  000404                        BR      2$
000040  010100                1$:     MOV     R1,R0                           ; COUNT,*                       5802
000042  006300                        ASL     R0
000044  005060  000000G                CLR     TALLY(R0)
000050  005201                2$:     INC     R1                              ; COUNT                         5801
000052  020102                        CMP     R1,R2                           ; COUNT,ENDIDX
000054  003771                        BLE     1$
000056  022626                        CMP     (SP)+,(SP)+                     ;                               5797
000060  000207                3$:     RTS     PC                              ;                               5777

; Routine Size:  25 words,      Routine Base:  $CODE$ + 4636
; Maximum stack depth per invocation:  6 words


                                        .SBTTL  L$AU ADD UNIT SECTION
000000  004737  004636'        L$AU::  JSR     PC,LAU                          ;                               5804
000004  104452                        TRAP    52
000006  000207                        RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4720
; Maximum stack depth per invocation:  2 words
```

```
;        5807  $sbttl  NON EXISTENT MEMORY TRAP HANDLER
;        5808
;        5809  !+
;        5810  !       THIS TRAP HANDLER IS VECTORED FROM LOCATION 4 FOR ALL UNIBUS TIMEOUT
;        5811  !       ERRORS, INDICATING THAT AN ATTEMPT WAS MADE TO REFERENCE A NON EXISTENT
;        5812  !       MEMORY LOCATION. ITS MAIN PURPOSE IS TO SET A FLAG FOR THE RDRX
;        5813  !       REGISTER EXISTENCE TEST, INDICATING THE ABSENCE OF A DEVICE REGISTER.
;        5814  !
;        5815
;        5816  BGNSRV (NEX_TRAP);
;        5817
;        5818  NEX = TRUE;                                  ! NEX TRAP OCCURRED
;        5819
;        5820  ENDSRV;


                                        .SBTTL  NEX.TRAP NON EXISTENT MEMORY TRAP HANDLER
000000  012737  000001  000000G      NEX.TRAP::
                                        MOV     #1,NEX                          ;                       5818
000006  000002                          RTI                                     ;                       5816

; Routine Size:  4 words,       Routine Base:  $CODE$ + 4730
; Maximum stack depth per invocation:  0 words
```

```
;       5821  %sbttl 'GLOBAL ROUTINES'
;       5822
;       5823  global routine SET_CPAR (CTLR) : novalue =
;       5824
;       5825  !+
;       5826  !       THIS ROUTINE SETS UP THE COMMONLY-USED CONTROLLER-RELATED DATA ITEMS
;       5827  !       FOR THE GIVEN CONTROLLER NUMBER.
;       5828  !
;       5829  !       INPUTS:
;       5830  !               CTLR - CONTROLLER NUMBER
;       5831  !
;       5832  !       IMPLICIT OUTPUTS:
;       5833  !               CCTLR - CURRENT CONTROLLER NUMBER
;       5834  !               CST_ADDR - ADDRESS OF CONTROLLER'S STATUS TABLE
;       5835  !               DCT_ADDR - ADDRESS OF CONTROLLER'S DRIVER TABLE
;       5836  !               RDRX_ADDR - ADDRESS OF CONTROLLER'S IP REGISTER
;       5837  !-
;       5838
;       5839      begin
;       5840      CCTLR = .CTLR;                          ! SET CURRENT CONTROLLER NUMBER
;       5841      CST_ADDR = CST + (.CTLR * CST_LEN * 2);     ! CALCULATE ADDRESS OF CONTROLLER'S CST
;       5842      DCT_ADDR = DCT + (.CTLR * DCT_LEN * 2);     ! CALCULATE ADDRESS OF CONTROLLER'S DCT
;       5843      RDRX_ADDR = .CST_ADDR [IP_ADDR];            ! GET CONTROLLER'S DEVICE ADDRESS
;       5844      end;

                                        .SBTTL  SET.CPAR GLOBAL ROUTINES
000000  010146                    SET.CPAR::
                                        MOV     R1,-(SP)                ;                       5823
000002  016601  000004                  MOV     4(SP),R1                ; CTLR,*                5840
000006  010137  000000G                 MOV     R1,CCTLR
000012  010146                          MOV     R1,-(SP)                ;                       5841
000014  012746  000056                  MOV     #56,-(SP)
000020  004737  000000G                 JSR     PC,BL$MUL
000024  062700  000000G                 ADD     #CST,R0
000030  010037  000000G                 MOV     R0,CST.ADDR
000034  010116                          MOV     R1,(SP)                 ;                       5842
000036  012746  000022                  MOV     #22.,-(SP)
000042  004737  000000G                 JSR     PC,BL$MUL
000046  062700  000000G                 ADD     #DCT,R0
000052  010037  000000G                 MOV     R0,DCT.ADDR
000056  017737  000000G 000000G         MOV     @CST.ADDR,RDRX.ADDR     ;                       5843
000064  062706  000006                  ADD     #6,SP                   ;                       5839
000070  012601                          MOV     (SP)+,R1                ;                       5823
000072  000207                          RTS     PC

; Routine Size:  30 words,      Routine Base:  $CODE$ + 4740
; Maximum stack depth per invocation:  5 words
```

```
global routine SET UPAR (OFFSET) : novalue =

!+
!       THIS ROUTINE SETS UP THE COMMONLY USED UNIT RELATED C        MS FOR
!       THE CURRENT CONTROLLER AND GIVEN CST OFFSET.
!
!       INPUTS:
!               OFFSET   WORD OFFSET INTO CURRENT CONTROLLER        WHICH
!                                DESCRIBES A UNIT
!
!       IMPLICIT INPUTS:
!               CST_ADDR   ADDRESS OF CURRENT CONTROLLER S CST
!
!       IMPLICIT OUTPUTS:
!               CUOFF - CURRENT UNIT S CST OFFSET
!               CDISK - CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
!               L$LUN   CURRENT UNIT NUMBER (DRS UNIT NUMBER)
!               T_ADDR - ADDRESS OF CURRENT UNIT S STATISTICS BLOCK (TALLY)
!

    begin
    CUOFF = .OFFSET;
    CDISK = .CST_ADDR [.OFFSET, D_DISK_NUM];
    L$LUN = .CST_ADDR [.OFFSET, D_UNIT];
    T_ADDR = TALLY + (.L$LUN * TALLY_LEN * 2);
    end;
```

```
                                        .SBTTL  SET.UPAR GLOBAL ROUTINES
000000  010146                    SET.UPAR::
                                        MOV     R1,-(SP)                ;                         5845
000002  016637  000004  000000G         MOV     4(SP),CUOFF             ; OFFSET,*                5866
000010  016600  000004                  MOV     4(SP),R0                ; CUOFF,*                 5867
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                 ;                         5868
000036  000301                          SWAB    R1
000040  042701  177760                  BIC     #177760,R1
000044  010137  000000G                 MOV     R1,L$LUN
000050  010146                          MOV     R1,-(SP)                ; L$LUN,*                 5869
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)+             ;                         5865
000074  012601                          MOV     (SP)+,R1                ;                         5845
000076  000207                          RTS     PC

; Routine Size:  32 words,      Routine Base:  $CODE$ + 5034
; Maximum stack depth per invocation:  4 words
```

```
;        5872  global routine GET_PKT (CTLR) =
;        5873
;        5874  !+
;        5875  !       THIS ROUTINE SEARCHES THE MSCP PACKET POOL ALLOCATION TABLE (PKT_USE)
;        5876  !       FOR A FREE MSCP PACKET TO ALLOCATE TO THE GIVEN CONTROLLER. IF ONE IS
;        5877  !       FOUND, THE PACKET IS ZEROED OUT, AND THE PACKET INDEX IS RETURNED
;        5878  !       TO THE CALLER. OTHERWISE, A -1 IS RETURNED INDICATING NONE AVAILABLE.
;        5879  !
;        5880  !       INPUTS:
;        5881  !               CTLR   CONTROLLER NUMBER REQUESTING ALLOCATION
;        5882  !-
;        5883
;        5884      begin
;        5885
;        5886      local
;        5887          index : signed word initial ( -1),
;        5888          RING_ADDR : word,
;        5889          PACKET_OWNED : byte,
;        5890          NEXT_PACKET : byte;
;        5891
;        5892      NEXT_PACKET = .NEXT_PKT_USE;                ! NEXT PACKET TO TRY
;        5893
;        5894      incr COUNT from 0 to (PKT_CNT - 1) do       ! FOR EACH ENTRY IN ALLOCATION TABLE
;        5895          begin
;        5896          PACKET_OWNED = FALSE;
;        5897
;        5898          if .PKT_USE [.NEXT_PACKET] lss 0        ! IF ENTRY INDICATES FREE PACKET
;        5899          then
;        5900              begin
;        5901              RING_ADDR = .DCT_ADDR [RR_BEG];     ! FIRST RESPONSE PACKET'S ADDRESS
;        5902
;        5903              incr I from 1 to (RRING_LEN + CRING_LEN) do ! FOR EACH PACKET ADDRESS
;        5904                  begin
;        5905
;        5906                  if (..RING_ADDR eqla .MSCP_PKT [.NEXT_PACKET, PKT_LO]) and
;        5907                     (((.RING_ADDR + 2) and ED_OWN) eql ED_OWN)
;        5908                  then
;        5909                      begin                       ! CHECK ADDRESS AND OWNERSHIP
;        5910                      PACKET_OWNED = TRUE;        ! PACKET OWNED BY CONTROLLER
;        5911                      exitloop;
;        5912                      end
;        5913                  else
;        5914                      RING_ADDR = .RING_ADDR + 4; ! ADDRESS OF NEXT PACKET IN RING
;        5915
;        5916                  end;
;        5917
;        5918              if not .PACKET_OWNED                ! IF NOT ALREADY USED
;        5919              then
;        5920                  begin
;        5921                  PKT_USE [.NEXT_PACKET] = .CTLR; ! ALLOCATE PACKET TO CONTROLLER
;        5922                  index = .NEXT_PACKET;
;        5923
;        5924                  incr J from 2 to (PKT_LEN - 1) do       ! ZERO OUT PACKET
;        5925                      MSCP_PKT [.NEXT_PACKET, .J, 0, 16, 0] = 0;
;        5926
;        5927                  exitloop;                       ! DONE
```

```
;          5928                    end;
;          5929
;          5930                end;
;          5931
;          5932            NEXT_PACKET = .NEXT_PACKET + 1;          ! TRY NEXT PACKET IN RING
;          5933
;          5934            if .NEXT_PACKET gequ PKT_CNT
;          5935            then
;          5936                NEXT_PACKET = 0;                     ! IF BEYOND ALL PACKETS, START AT THE TOP
;          5937
;          5938            end;
;          5939
;          5940        if (.index geq 0) and                        ! IF PACKET FOUND
;          5941           (.PKT_USE [.index] geq 0)
;          5942        then
;          5943            begin
;          5944            MSCP_PKT [.index, MSGLEN] = SZ_GEN;      ! PACKET SIZE   ONLY ONLINE AND SCC CHANGE IT
;          5945            MSCP_PKT [.index, CREDITS] = 1;          ! CREDIT SIZE
;          5946            NEXT_PKT_USE = .NEXT_PACKET + 1;         ! NEXT PACKET TO ALLOCATE
;          5947
;          5948            if.NEXT_PKT_USE gequ PKT_CNT
;          5949            then
;          5950                NEXT_PKT_USE = 0;                    ! CYCLE BACK TO BEGINNING IF AT END
;          5951
;          5952            end;
;          5953
;          5954        return .index;
;          5955        end;

                        .SBTTL  GET.PKT GLOBAL ROUTINES
000000  004137  000000G       GET.PKT::
                        JSR     R1,$SAVE5                ;                                   5872
000004  162706  000006          SUB     #6,SP
000010  012704  177777          MOV     #-1,R4           ; *.INDEX                          5884
000014  113705  000000G         MOVB    NEXT.PKT.USE,R5  ; *.NEXT.PACKET                    5892
000020  012766  000014  000004  MOV     #14,4(SP)        ; *.COUNT                          5894
000026  105066  000002      1$: CLRB    2(SP)            ; PACKET.OWNED                     5896
000032  005002                  CLR     R2               ;                                  5898
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      ;                                  5901
000050  016016  000004          MOV     4(R0),(SP)       ; *.RING.ADDR
000054  010246                  MOV     R2,-(SP)         ;                                  5906
000056  012746  000104          MOV     #104,-(SP)
000062  004737  000000G         JSR     PC,BL$MUL
000066  012703  000010          MOV     #10,R3           ; *.I                              5903
000072  027660  000004  000000G 2$: CMP     @4(SP),MSCP.PKT(R0)  ; RING.ADDR,*              5906
000100  001016                  BNE     3$
000102  012746  000002          MOV     #2,-(SP)         ;                                  5907
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                5916
000134  000405                          BR      4$                      ;                               5909
000136  062766  000004  000004  3$:     ADD     #4,4(SP)                ; *,RING.ADDR                   5914
000144  005303                          DEC     R3                      ; I                             5903
000146  001351                          BNE     2$
000150  032766  000001  000006  4$:     BIT     #1,6(SP)                ; *,PACKET.OWNED                5918
000156  001027                          BNE     6$
000160  116662  000030  000000G         MOVB    30(SP),PKT.USE(R2)      ; CTLR,*                        5921
000166  010204                          MOV     R2,R4                   ; *,INDEX                       5922
000170  010216                          MOV     R2,(SP)                 ;                               5925
000172  012746  000042                  MOV     #42,-(SP)
000176  004737  0000000G                JSR     PC,BL$MUL
000202  005726                          TST     (SP)+
000204  012701  000002                  MOV     #2,R1                   ; *,J                           5924
000210  010003                  5$:     MOV     R0,R3                   ;                               5925
000212  060103                          ADD     R1,R3                   ; J,*
000214  006303                          ASL     R3
000216  005063  000000G                 CLR     MSCP.PKT(R3)
000222  005201                          INC     R1                      ; J                             5924
000224  020127  000041                  CMP     R1,#41                  ; J,*
000230  003767                          BLE     5$
000232  022626                          CMP     (SP)+,(SP)+             ;                               5920
000234  000411                          BR      9$
000236  022626                  6$:     CMP     (SP)+,(SP)+             ;                               5900
000240  105205                  7$:     INCB    R5                      ; NEXT.PACKET                   5932
000242  120527  000014                  CMPB    R5,#14                  ; NEXT.PACKET,*                 5934
000246  103401                          BLO     8$
000250  105005                          CLRB    R5                      ; NEXT.PACKET                   5936
000252  005366  000004  8$:     DEC     4(SP)                   ; COUNT                         5894
000256  001263                          BNE     1$
000260  005704                  9$:     TST     R4                      ; INDEX                         5940
000262  002434                          BLT     11$
000264  105764  000000G                 TSTB    PKT.USE(R4)             ; *(INDEX)                      5941
000270  002431                          BLT     11$
000272  010446                          MOV     R4,-(SP)                ; INDEX,*                       5944
000274  012746  000104                  MOV     #104,-(SP)
000300  004737  0000000G                JSR     PC,BL$MUL
000304  012760  000040  000006G         MOV     #40,MSCP.PKT+6(R0)
000312  142760  000017  000010G         BICB    #17,MSCP.PKT+10(R0)     ;                               5945
000320  152760  000001  000010G         BISB    #1,MSCP.PKT+10(R0)
000326  005000                          CLR     R0                      ;                               5946
000330  150500                          BISB    R5,R0                   ; NEXT.PACKET,*
000332  005200                          INC     R0
000334  110037  000000G                 MOVB    R0,NEXT.PKT.USE
000340  120027  000014                  CMPB    R0,#14                  ; NEXT.PKT.USE,*                5948
000344  103402                          BLO     10$
000346  105037  000000G                 CLRB    NEXT.PKT.USE            ;                               5950
000352  022626                  10$:    CMP     (SP)+,(SP)+             ;                               5943
000354  010400                  11$:    MOV     R4,R0                   ; INDEX,*                       5884
000356  062706  000006                  ADD     #6,SP                   ;                               5872
000362  000207                          RTS     PC

; Routine Size:  122 words,    Routine Base:  $CODE$ + 5134
; Maximum stack depth per invocation:  13 words

;         5956
```

```
;          5957  global routine PUT_PKT (index) : novalue =
;          5958
;          5959  !+
;          5960  !       THE MSCP PACKET DESIGNATED BY 'INDEX' IS RETURNED TO THE POOL BY THIS
;          5961  !       ROUTINE.
;          5962  !
;          5963
;          5964      begin
;          5965
;          5966      local
;          5967          RING_ADDR : word,
;          5968          OWNER : word;
;          5969
;          5970      RING_ADDR = .DCT_ADDR [RR_BEG];                     ! ADDRESS IN FIRST RESPONSE RING
;          5971
;          5972      incr COUNT from 1 to (RRING_LEN + CRING_LEN) do     ! FOR EACH ADDRESS IN THE RINGS
;          5973          begin
;          5974
;          5975          if .MSCP_PKT [.index, PKT_LO] eqla ..RING_ADDR  ! IF ADDRESS MATCHES
;          5976          then
;          5977              begin
;          5978              OWNER = .RING_ADDR + 2;                     ! ADDRESS OF OWNERSHIP WORD
;          5979              .OWNER = ..OWNER and (not (ED_OWN)) and (not (ED_FLAG)) ; ! GIVE OWNERSHIP TO HOST
;          5980              end;
;          5981
;          5982          RING_ADDR = .RING_ADDR + 4;                     ! LOOK AT NEXT PACKET ADDRESS IN RING
;          5983          end;
;          5984
;          5985      PKT_USE [.index] = -1;
;          5986      end;

                                        .SBTTL  PUT.PKT GLOBAL ROUTINES
000000  004137  000000G           PUT.PKT::
                                        JSR     R1,$SAVE4               ;                                       5957
000004  013700  000000G                 MOV     DCT.ADDR,R0             ;                                       5970
000010  016003  000004                  MOV     4(R0),R3                ; *,RING.ADDR
000014  016601  000014                  MOV     14(SP),R1               ; INDEX,*                               5975
000020  010146                          MOV     R1,-(SP)
000022  012746  000104                  MOV     #104,-(SP)
000026  004737  000000G                 JSR     PC,BL$MUL
000032  012702  000010                  MOV     #10,R2                  ; *,COUNT                               5972
000036  026013  000000G           1$:   CMP     MSCP.PKT(R0),(R3)       ; *,RING.ADDR                           5975
000042  001005                          BNE     2$
000044  012704  000002                  MOV     #2,R4                   ; *,OWNER                               5978
000050  060304                          ADD     R3,R4                   ; RING.ADDR,OWNER
000052  042714  140000                  BIC     #140000,(R4)            ; *,OWNER                               5979
000056  062703  000004           2$:    ADD     #4,R3                   ; *,RING.ADDR                           5982
000062  005302                          DEC     R2                      ; CO'                                   5972
000064  001364                          BNE     1$
000066  112761  000377  000000G        MOVB    #377,PKT.USE(R1)        ;                                       5985
000074  022626                          CMP     (SP)+,(SP)+             ;                                       5964
000076  000207                          RTS     PC                      ;                                       5957

; Routine Size:  32 words,      Routine Base:  $CODE$ + 5520
; Maximum stack depth per invocation:  8 words
```

```
;          5987  routine PUTA_PKT (CTLR) : novalue
;          5988
;          5989  !*
;          5990  !       THIS ROUTINE DEALLOCATES ALL MSCP PACKETS WHICH HAVE BEEN ALLOCATED
;          5991  !       TO A PARTICULAR CONTROLLER.
;          5992  !
;          5993  !       INPUTS:
;          5994  !               CTLR - CONTROLLER NUMBER
;          5995  !
;          5996
;          5997      incr COUNT from 0 to (PKT_CNT - 1) do       ! FOR EACH ENTRY IN ALLOCATION TABLE
;          5998
;          5999          if .PKT_USE [.COUNT] eql .CTLR          ! IF PACKET IS ALLOCATED TO GIVEN CONTROLLER
;          6000          then
;          6001              PKT_USE [.COUNT] = 1;               ! DEALLOCATE IT


                                        .SBTTL  PUTA.PKT GLOBAL ROUTINES
000000  010146                  PUTA.PKT:
                                        MOV     R1,-(SP)                ;                                       5987
000002  005000                          CLR     R0                      ; COUNT                                 5997
000004  116001  000000G         1$:     MOVB    PKT.USE(R0),R1          ; *(COUNT),*                            5999
000010  020166  000004                  CMP     R1,4(SP)                ; *,CTLR
000014  001003                          BNE     2$
000016  112760  000377  000000G         MOVB    #377,PKT.USE(R0)        ; *,*(COUNT)                            6001
000024  005200                  2$:     INC     R0                      ; COUNT                                 5997
000026  020027  000013                  CMP     R0,#13                  ; COUNT,*
000032  003764                          BLE     1$
000034  012601                          MOV     (SP)+,R1                ;                                       5987
000036  000207                          RTS     PC

; Routine Size:  16 words,      Routine Base:  $CODE$ + 5620
; Maximum stack depth per invocation:  2 words
```

```
;       6002  global routine GET_RETPKT (CTLR)
;       6003
;       6004  !+
;       6005  !       THIS ROUTINE SEARCHES THE RETURN PACKET POOL ALLOCATION TABLE (RP_USE)
;       6006  !       FOR A FREE RETURN PACKET TO ALLOCATE TO THE GIVEN CONTROLLER. IF ONE IS
;       6007  !       FOUND, THE PACKET IS ZEROED OUT, AND THE PACKET INDEX IS RETURNED TO
;       6008  !       THE CALLER. OTHERWISE, A -1 IS RETURNED INDICATING NONE AVAILABLE.
;       6009  !
;       6010  !       INPUTS:
;       6011  !               CTLR - CONTROLLER NUMBER REQUESTING ALLOCATION
;       6012  !-
;       6013
;       6014      begin
;       6015
;       6016      local
;       6017          index : signed word initial ( -1);          ! ASSUME NONE AVAILABLE
;       6018
;       6019      incr COUNT from 0 to (RP_CNT - 1) do            ! FOR EACH ENTRY IN TABLE
;       6020
;       6021          if .RP_USE [.COUNT] lss 0                   ! IF FREE RETPKT IS FOUND
;       6022          then
;       6023              begin
;       6024              RP_USE [.COUNT] = .CTLR;                ! ALLOCATE RETURN PACKET TO CONTROLLER
;       6025              index = .COUNT;
;       6026
;       6027              incr J from 0 to (RP_LEN - 1) do        ! ZERO OUT RETPKT
;       6028                  RETPKT [.COUNT, .J, 0, 16, 0] = 0;
;       6029
;       6030              exitloop;                               ! DONE
;       6031              end;
;       6032
;       6033      return .index;                                  ! RETURN PACKET INDEX (OR -1) TO CALLER
;       6034      end;
```

```
                                        .SBTTL  GET.RETPKT GLOBAL ROUTINES
000000  004137  000000G            GET.RETPKT::
                                        JSR     R1,$SAVE4                       ;                               6002
000004  012704  177777                  MOV     #-1,R4                          ; *,INDEX                       6014
000010  005003                          CLR     R3                              ; COUNT                         6019
000012  105763  000000G          1$:     TSTB    RP.USE(R3)                      ; *(COUNT)                      6021
000016  002025                          BGE     3$
000020  116663  000014  000000G         MOVB    14(SP),RP.USE(R3)               ; CTLR,*(COUNT)                 6024
000026  010304                          MOV     R3,R4                           ; COUNT,INDEX                   6025
000030  010346                          MOV     R3,-(SP)                        ; COUNT,*                       6028
000032  012746  000030                  MOV     #30,-(SP)
000036  004737  000000G                JSR     PC,BL$MUL
000042  022626                          CMP     (SP)+,(SP)+
000044  005002                          CLR     R2                              ; J                             3027
000046  010001                 2$:      MOV     R0,R1                           ;                               6028
000050  060201                          ADD     R2,R1                           ; J,*
000052  006301                          ASL     R1
000054  005061  000000G                CLR     RETPKT(R1)
000060  005202                          INC     R2                              ; J                             6027
000062  020227  000027                  CMP     R2,#27                          ; J,*
000066  003767                          BLE     2$
000070  000404                          BR      4$                              ;                               6023
000072  005203                 3$:      INC     R3                              ; COUNT                         6019
000074  020327  000003                  CMP     R3,#3                           ; COUNT,*
000100  003744                          BLE     1$
000102  010400                 4$:      MOV     R4,R0                           ; INDEX,*                       6014
000104  000207                          RTS     PC                              ;                               6002

; Routine Size:  35 words,      Routine Base:  $CODE$ + 5660
; Maximum stack depth per invocation:  8 words
```

```
;          6035  global routine PUT_RETPKT (index) : novalue =
;          6036
;          6037  !+
;          6038  !       THE RETURN PACKET DESIGNATED BY "INDEX" IS RETURNED TO THE POOL BY THIS
;          6039  !       ROUTINE.
;          6040  !-
;          6041
;          6042      RP_USE [.index] = -1;


                                          .SBTTL  PUT.RETPKT GLOBAL ROUTINES
000000  016600  000002                PUT.RETPKT::
                                          MOV     2(SP),R0                        ; INDEX,*                       6042
000004  112760  000377  000000G            MOVB    #377,RP.USE(R0)
000012  000207                            RTS     PC                              ;                               6035

; Routine Size:  6 words,       Routine Base:  $CODE$ + 5766
; Maximum stack depth per invocation:  0 words
```

```
;       6043  global routine GET_IO_BUFF (ADDR) : novalue =
;       6044
;       6045  !+
;       6046  !       THIS ROUTINE HANDLES THE ALLOCATION OF AN I/O BUFFER FROM THE BUFFER
;       6047  !       POOL.
;       6048  !
;       6049  !       INPUTS:
;       6050  !               ADDR   ADDRESS TO STORE THE 2 WORD BUFFER DESCRIPTOR
;       6051  !
;       6052  !       IMPLICIT INPUTS:
;       6053  !               CCTLR - CURRENT CONTROLLER NUMBER
;       6054  !
;       6055  !       OUTPUTS:
;       6056  !               THE ALLOCATED BUFFER'S DESCRIPTOR IS LOADED INTO THE TWO
;       6057  !               WORDS AT "ADDR" AND "ADDR + 2'. OTHERWISE, A ZERO IS RETURNED
;       6058  !               AT "ADDR" IF NO BUFFERS ARE AVAILABLE.
;       6059  !-
;       6060
;       6061      begin
;       6062      .ADDR = 0;                                  ! ASSUME FAILURE
;       6063
;       6064      incr COUNT from 0 to (QIO_PER_CTLR * MAX_CTLR - 1) do       ! FOR EACH ENTRY IN BUFFER TABLE
;       6065
;       6066          if .BUFF_OWN [.COUNT] lss 0             ! IF BUFFER IS FREE
;       6067          then
;       6068              begin
;       6069              BUFF_OWN [.COUNT] = .CCTLR;         ! ALLOCATE BUFFER TO CONTROLLER
;       6070              .ADDR = .BUFF_ADDR [.COUNT];        ! RETURN BUFFER DESCRIPTOR
;       6071              exitloop;                           ! DONE
;       6072              end;
;       6073
;       6074      end;                                        ! ROUTINE GET_IO_BUFF


                                        .SBTTL  GET.IO.BUFF GLOBAL ROUTINES
000000  010146                  GET.IO.BUFF::
                                        MOV     R1,-(SP)                ;                               6043
000002  005076  000004                  CLR     @4(SP)                  ; ADDR                          6062
000006  005001                          CLR     R1                      ; COUNT                         6064
000010  105761  000000G         1$:     TSTB    BUFF.OWN(R1)            ; *(COUNT)                      6066
000014  002011                          BGE     2$
000016  113761  000000G 000000G         MOVB    CCTLR,BUFF.OWN(R1)      ; *,*(COUNT)                    6069
000024  010100                          MOV     R1,R0                   ; COUNT,*                       6070
000026  006300                          ASL     R0
000030  016076  000000G 000004          MOV     BUFF.ADDR(R0),@4(SP)    ; *,ADDR
000036  000404                          BR      3$                      ;                               6068
000040  005201                  2$:     INC     R1                      ; COUNT                         6064
000042  020127  000007                  CMP     R1,#7                   ; COUNT,*
000046  003760                          BLE     1$
000050  012601                  3$:     MOV     (SP)+,R1                ;                               6043
000052  000207                          RTS     PC

; Routine Size:  22 words,      Routine Base:  $CODE$ + 6002
; Maximum stack depth per invocation:  2 words
```

```
;          6075  global rout'ne PUT IO BUFF (ADDR) : novalue =
;          6076
;          6077  !+
;          6078  !       THIS ROUTINE HANDLES THE DEALLOCATION OF AN I/O BUFFER, RETURNING IT
;          6079  !       TO THE BUFFER POOL.
;          6080  !
;          6081  !       INPUTS:
;          6082  !               ADDR    ADDRESS OF THE 2-WORD BUFFER DESCRIPTOR TO BE
;          6083  !                       DEALLOCATED
;          6084  !
;          6085
;          6086      incr COUNT from 0 to (QIO_PER_CTLR * MAX_CTLR - 1) do       ! FOR EACH ENTRY IN BUFFER TABLE
;          6087
;          6088          if .BUFF_ADDR [.COUNT] eqla ..ADDR      ! IF THIS IS THE BUFFER'S ENTRY
;          6089          then
;          6090              begin
;          6091              BUFF_OWN [.COUNT] =  1;             ! DEALLOCATE BUFFER
;          6092              exitloop;                           ! DONE
;          6093              end;


                                        .SBTTL  PUT.IO.BUFF GLOBAL ROUTINES
000000  010146                  PUT.IO.BUFF::
                                        MOV     R1,-(SP)                        ;                                       6075
000002  005001                          CLR     R1                              ; COUNT                                 6086
000004  010100                  1$:     MOV     R1,R0                           ; COUNT,*                               6088
000006  006300                          ASL     R0
000010  026076  000000G 000004          CMP     BUFF.ADDR(R0),@4(SP)            ; *,ADDR
000016  001004                          BNE     2$
000020  112761  000377  000000G         MOVB    #377,BUFF.OWN(R1)               ; *,*(COUNT)                            6091
000026  000404                          BR      3$                              ;                                       6090
000030  005201                  2$:     INC     R1                              ; COUNT                                 6086
000032  020127  000007                  CMP     R1,#7                           ; COUNT,*
000036  003762                          BLE     1$
000040  012601                  3$:     MOV     (SP)+,R1                        ;                                       6075
000042  000207                          RTS     PC

; Routine Size:  18 words,      Routine Base:  $CODE$ + 6056
; Maximum stack depth per invocation:  2 words
```

```
;          6094  global routine PUTA BUFF : novalue =
;          6095
;          6096  !+
;          6097  !       THIS ROUTINE DEALLOCATES ALL I/O BUFFERS WHICH HAVE BEEN ALLOCATED TO
;          6098  !       THE CURRENT CONTROLLER (CCTLR).
;          6099  !-
;          6100
;          6101      incr COUNT from 0 to (QIO PER CTLR * MAX CTLR - 1) do       ! FOR EACH ENTRY IN BUFFER TABLE
;          6102
;          6103          if .BUFF_OWN [.COUNT] eql .CCTLR        ! IF THIS BUFFER IS ALLOCATED TO THE CURRENT CONTROLLER
;          6104          then
;          6105              BUFF_OWN [.COUNT] = -1;             ! DEALLOCATE IT


                                        .SBTTL  PUTA.BUFF GLOBAL ROUTINES
000000  010146                  PUTA.BUFF::
                                        MOV     R1,-(SP)                        ;                                       6094
000002  005000                          CLR     R0                              ; COUNT                                 6101
000004  116001  000000G         1$:     MOVB    BUFF.OWN(R0),R1                 ; *(COUNT),*                            6103
000010  020137  000000G                 CMP     R1,CCTLR
000014  001003                          BNE     2$
000016  112760  000377  000000G        MOVB    #377,BUFF.OWN(R0)               ; *,*(COUNT)                            6105
000024  005200                  2$:     INC     R0                              ; COUNT                                 6101
000026  020027  000007                  CMP     R0,#7                           ; COUNT,*
000032  003764                          BLE     1$
000034  012601                          MOV     (SP)+,R1                        ;                                       6094
000036  000207                          RTS     PC

; Routine Size:  16 words,      Routine Base:  $CODE$ + 6122
; Maximum stack depth per invocation:  2 words
```

```
;        6106  global routine OUT_IODQ =
;        6107
;        6108  !+
;        6109  !       THIS ROUTINE RETURNS TO THE CALLER THE NEXT RETPKT INDEX TO BE
;        6110  !       PROCESSED FROM THE I/O DONE QUEUE (IODQ). THE "OUT" POINTER TO THE
;        6111  !       QUEUE IS ALSO UPDATED.
;        6112  !
;        6113  !       INPUTS:
;        6114  !               NONE
;        6115  !
;        6116  !       OUTPUTS:
;        6117  !               THE INDEX OF THE NEXT RETPKT TO BE PROCESSED.
;        6118  !-
;        6119
;        6120      begin
;        6121
;        6122      local
;        6123          index : word;
;        6124
;        6125      index = .IODQ [.IODQ_OUT];                ! GET NEXT RETPKT INDEX
;        6126      IODQ_OUT = .IODQ_OUT + 1;                 ! ADVANCE "OUT" POINTER
;        6127
;        6128      if .IODQ_OUT gequ IODQ_LEN                ! IF BEYOND END OF QUEUE
;        6129      then
;        6130          IODQ_OUT = 0;                         ! SET POINTER TO BEGINNING OF QUEUE
;        6131
;        6132      return .index;                            ! RETURN INDEX TO CALLER
;        6133      end;

                                .SBTTL  OUT.IODQ GLOBAL ROUTINES
000000  013700  000000G        OUT.IODQ::
                                MOV     IODQ.OUT,R0                     ;                       6125
000004  116000  000000G                MOVB    IODQ(R0),R0             ; *,INDEX
000010  042700  177400                 BIC     #177400,R0              ; *,INDEX
000014  005237  000000G                INC     IODQ.OUT                ;                       6126
000020  023727  000000G 000004         CMP     IODQ.OUT,#4             ;                       6128
000026  103402                         BLO     1$
000030  005037  000000G                CLR     IODQ.OUT                ;                       6130
000034  000207                 1$:     RTS     PC                      ;                       6106

; Routine Size:  15 words,      Routine Base:  $CODE$ + 6162
; Maximum stack depth per invocation:  0 words
```

```
;        6134  global routine IN_IODQ ( index ) : novalue =
;        6135
;        6136  !+
;        6137  !       THIS ROUTINE INSERTS A RETURN PACKET INDEX INTO THE I/O DONE QUEUE, AND
;        6138  !       UPDATES THE IODQ IN POINTER.
;        6139  !
;        6140
;        6141      if ((.IODQ_IN + 1) eql .IODQ_OUT) or
;        6142         (.IODQ_IN   (IODQ_LEN   1) eql .IODQ_OUT)
;        6143      then
;        6144          return
;        6145      else
;        6146          begin
;        6147          IODQ[.IODQ_IN] = .index;                ! LOAD INDEX INTO QUEUE
;        6148          IODQ_IN = .IODQ_IN + 1;                 ! ADVANCE "IN" POINTER
;        6149
;        6150          if .IODQ_IN gequ IODQ_LEN               ! IF BEYOND END OF QUEUE
;        6151          then
;        6152              IODQ_IN = 0;                        ! CYCLE BACK TO BEGINNING OF QUEUE
;        6153
;        6154          end;                                    ! IF IODQ IS NOT FULL


                                .SBTTL  IN.IODQ GLOBAL ROUTINES
000000  010146                IN.IODQ::
                                MOV     R1, (SP)                ;                       6134
000002  013701  000000G         MOV     IODQ.IN,R1              ;                       6141
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                   ;                       6142
000022  162700  000003          SUB     #3,R0
000026  020037  000000G         CMP     R0,IODQ.OUT
000032  001413                  BEQ     1$                      ;                       6144
000034  116661  000004  000000G MOVB    4(SP),IODQ(R1)          ; INDEX,*               6147
000042  005237  000000G         INC     IODQ.IN                 ;                       6148
000046  023727  000000G 000004  CMP     IODQ.IN,#4              ;                       6150
000054  103402                  BLO     1$
000056  005037  000000G         CLR     IODQ.IN                 ;                       6152
000062  012601                1$: MOV     (SP)+,R1                ;                       6154
000064  000207                  RTS     PC

; Routine Size:  27 words,    Routine Base:  $CODE$ + 6220
; Maximum stack depth per invocation:  2 words
```

```
;        6155  global routine DROP_CTLR (CTLR, REASON) : novalue =
;        6156
;        6157  !+
;        6158  !       THIS ROUTINE DROPS ALL UNITS ASSOCIATED WITH THE CONTROLLER DESIGNATED
;        6159  !       BY "CTLR". THE REASON FOR DROPPING THE DEVICE IS LOADED INTO THE DUR
;        6160  !       VECTOR FOR EACH ATTACHED UNIT. THIS DATA IS THEN USED BY THE DROP UNIT
;        6161  !       SECTION.
;        6162  !
;        6163
;        6164      begin
;        6165
;        6166      local
;        6167          UNIT;
;        6168
;        6169      incr N from (0 * OF UN) to (3 * UNIT_SIZE * OF UN) by UNIT_SIZE do  ! FOR EACH UNIT IN CST
;        6170
;        6171          if .CST [.CTLR, .N, D_PRES] eql PRESENT ! IF UNIT IS CONFIGURED
;        6172          then
;        6173              begin
;        6174              UNIT = .CST [.CTLR, .N, D_UNIT];    ! GET DRS UNIT NUMBER
;        6175              DUR [.UNIT] = .REASON;              ! SET REASON FOR DROPPING UNIT
;        6176              DODU (.UNIT);                       ! DROP UNIT
;        6177              end;
;        6178
;        6179      end;


                                        .SBTTL  DROP.CTLR GLOBAL ROUTINES
000000  004137  000000G           DROP.CTLR::
                                  JSR     R1,$SAVE3                       ;                       6155
000004  016646  000014            MOV     14(SP),-(SP)                    ; CTLR,*                6171
000010  012746  000027            MOV     #27,-(SP)
000014  004737  000000G           JSR     PC,BL$MUL
000020  010003                    MOV     R0,R3
000022  012702  000003            MOV     #3,R2                           ; *,N                   6169
000026  010300                1$: MOV     R3,R0                           ;                       6171
000030  060200                    ADD     R2,R0                           ; N,*
000032  006300                    ASL     R0
000034  032760  040000  000000G   BIT     #40000,CST(R0)
000042  001412                    BEQ     2$
000044  016001  000000G           MOV     CST(R0),R1                      ; *,UNIT                6174
000050  000301                    SWAB    R1                              ; UNIT
000052  042701  177760            BIC     #177760,R1                      ; *,UNIT
000056  116661  000016  000000G   MOVB    16(SP),DUR(R1)                  ; REASON,*(UNIT)        6175
000064  010100                    MOV     R1,R0                           ; UNIT,*                6176
000066  104451                    TRAP    51
000070  062702  000005        2$: ADD     #5,R2                           ; *,N                   6169
000074  020227  000022            CMP     R2,#22                          ; N,*
000100  003752                    BLE     1$
000102  022626                    CMP     (SP)+,(SP)+                     ;                       6164
000104  000207                    RTS     PC                              ;                       6155

; Routine Size:  35 words,      Routine Base:  $CODE$ + 6306
; Maximum stack depth per invocation:  8 words
```

```
;         6180  global routine DRV_CTLERR (CTLR) : novalue =
;         6181
;         6182  !+
;         6183  !       THIS ROUTINE IS CALLED BY DRV_TIMCHK AND FATAL_ERROR WHENEVER AN
;         6184  !       UNRECOVERABLE CONTROLLER ERROR HAS BEEN DETECTED. ITS PURPOSE IS TO
;         6185  !       CLEAN UP ALL CONTROLLER-RELATED DATA IN THE "DRIVER" PORTION OF THE
;         6186  !       PROGRAM. THIS INCLUDES MARKING THE CONTROLLER OFFLINE, CLEARING THE
;         6187  !       C-RING COUNT, AND DEALLOCATING MSCP PACKETS DESCRIBED IN THE RESPONSE
;         6188  !       RING.
;         6189  !
;         6190  !       INPUTS:
;         6191  !               CTLR - DYING CONTROLLER NUMBER
;         6192  !
;         6193
;         6194      begin
;         6195
;         6196      local
;         6197          D_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS);  ! CONTROLLER'S DCT ADDRESS
;         6198
;         6199      D_ADDR = DCT + (.CTLR * DCT_LEN * 2);       ! GET CONTROLLER'S DCT ADDR
;         6200      D_ADDR [WORD0] = OFFLINE;                   ! MARK DCT OFFLINE AND CLEAR CRING_CNT
;         6201      PUTA_PKT (.CTLR);                           ! RELEASE ALL PACKETS ALLOCATED TO CONTROLLER
;         6202      DROP_CTLR (.CTLR, DU_CFATAL);               ! DROP ALL UNITS ON THE CONTROLLER
;         6203      end;                                        ! ROUTINE DRV_CTLERR


                                        .SBTTL  DRV.CTLERR GLOBAL ROUTINES
000000  010146                  DRV.CTLERR::
                                        MOV     R1, (SP)                        ;                               6180
000002  016601  000004                  MOV     4(SP),R1                        ; CTLR,*                        6199
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                        6200
000026  010116                          MOV     R1,(SP)                         ;                               6201
000030  004737  005620'                 JSR     PC,PUTA.PKT
000034  010116                          MOV     R1,(SP)                         ;                               6202
000036  012746  000006                  MOV     #6,-(SP)
000042  004737  006306'                 JSR     PC,DROP.CTLR
000046  062706  000006                  ADD     #6,SP                           ;                               6194
000052  012601                          MOV     (SP)+,R1                        ;                               6180
000054  000207                          RTS     PC

; Routine Size:  23 words,      Routine Base:  $CODE$ + 6414
; Maximum stack depth per invocation:  5 words
```

```
;       6204  global routine SEND (index) =
;       6205
;       6206  !+
;       6207  !       IF THE CURRENT RDRX IS ONLINE AND ITS CRING IS NOT FULL, THEN THIS
;       6208  !       ROUTINE "SENDS" A COMMAND TO THE RDRX BY LOADING THE PACKET
;       6209  !       DESCRIPTOR OF AN MSCP PACKET INTO THE COMMAND RING AND READING THE
;       6210  !       DEVICE'S IP REGISTER.  IF THE
;       6211  !       CURRENT RDRX IN NOT ON_LINE, THEN A FAILURE INDICATION IS RETURNED TO
;       6212  !       THE CALLER, AND NO ACTION IS TAKEN.
;       6213  !
;       6214  !       INPUTS:
;       6215  !               INDEX   INDEX OF MSCP PACKET CONTAINING THE COMMAND TO
;       6216  !                       BE SENT
;       6217  !
;       6218  !       IMPLICIT INPUTS:
;       6219  !               CCTLR - CURRENT CONTROLLER NUMBER
;       6220  !               DCT_ADDR   ADDRESS OF CURRENT CONTROLLER'S DCT
;       6221  !-
;       6222
;       6223      begin
;       6224
;       6225      local
;       6226          SLOT_ADDR,
;       6227          TEMP : word;
;       6228
;       6229      if ((.DCT_ADDR [STAT] eql ONLINE) and                       ! IF DEVICE IS ONLINE AND
;       6230          (.DCT_ADDR [CRING_CNT] lssu CRING_LEN)) or              ! ITS CRING IS NOT FULL
;       6231         ((.MSCP_PKT [.index, OPCODE] eql OP_SCC) and             ! OR IT IS A SET-CTRL CHAR COMMAND
;       6232          (.DCT_ADDR [CRING_CNT] lssu CRING_LEN))                 !
;       6233      then
;       6234
;       6235          if (not ((.MSCP_PKT [.index, OPCODE] eql OP_ACC) or
;       6236                   (.MSCP_PKT [.index, OPCODE] eql OP_ONL) or
;       6237                   (.MSCP_PKT [.index, OPCODE] eql OP_RD) or
;       6238                   (.MSCP_PKT [.index, OPCODE] eql OP_SCC) or
;       6239                   (.MSCP_PKT [.index, OPCODE] eql OP_WPT) OR  !+
;       6240                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_SDD) OR
;       6241                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_RCD) OR
;       6242                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_GDS) OR
;       6243                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_ELP) OR
;       6244                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_ABP) OR
;       6245                   (.MSCP_PKT[.INDEX, OPCODE] EQL OP_ESP) ) )
;       6246          then
;       6247              begin
;       6248  !           PRINTF (DBM107, .MSCP_PKT [.index, OPCODE]);        !JSD REV A
;       6249              return FAILURE;
;       6250              end
;       6251          else
;       6252              begin
;       6253
;       6254              do
;       6255                  BREAK                                           ! LOOP TILL CREDIT BALANCE POSITIVE
;       6256              until ((.MSCP_PKT [.index, CMD_TYPE] eql IMM_CMD) and
;       6257                     (.CREDIT_BAL gequ 1)) or
;       6258                    (.CREDIT_BAL gtru 1);
;       6259
```

```
;       6260              MSCP_PKT [.index, CRN_LO] = (CRN_LOW = .CRN_LOW + 1);       ! ASSIGN CMD REF NUM
;       6261
;       6262              if .CRN_LOW eql 0
;       6263              then  CRN_HIGH = .CRN_HIGH + 1;
;       6264
;       6265              MSCP_PKT [.index, CRN_HI] = .CRN_HIGH;
;       6266
;       6267              SLOT_ADDR = .DCT_ADDR [CR_NEXT];                    ! ADDR OF NEXT COMMAND SLOT
;       6268
;       6269              DO BREAK
;       6270              UNTIL ((.(.SLOT_ADDR + 2) and ED_OWN) eql 0);
;       6271
;       6272              SETPRI (PRIO7);
;       6273
;       6274              .SLOT_ADDR = .MSCP_PKT [.index, PKT_LO];            ! LOAD BUFF DESC (LO) INTO COMMAND SLOT
;       6275              SLOT_ADDR = .SLOT_ADDR + 2;                         ! ADVANCE TO NEXT WORD
;       6276              .SLOT_ADDR = .MSCP_PKT [.index, PKT_HI];            ! LOAD BUFF DESC (HI) INTO COMMAND SLOT
;       6277              .SLOT_ADDR = ..SLOT_ADDR and (not (ED_FLAG));       ! CLEAR INTERRUPT FLAG IN CASE SET
;       6278              .SLOT_ADDR = ..SLOT_ADDR or ED_OWN;                 ! GIVE OWNERSHIP TO CONTROLLER
;       6279              SLOT_ADDR = .SLOT_ADDR + 2;                         ! ADVANCE TO NEXT COMMAND SLOT
;       6280
;       6281              if .SLOT_ADDR gtra .DCT_ADDR [CR_END]               ! IF BEYOND END OF CRING
;       6282              then
;       6283                  SLOT_ADDR = .DCT_ADDR [CR_BEG];                 ! CYCLE BACK TO BEGINNING
;       6284
;       6285              DCT_ADDR [CR_NEXT] = .SLOT_ADDR;                    ! RESTORE CR_NEXT POINTER IN DCT
;       6286              DCT_ADDR [CRING_CNT] = .DCT_ADDR [CRING_CNT] + 1;   ! INCR # OF COMMANDS IN CRING
;       6287              CREDIT_BAL = .CREDIT_BAL - 1;                       ! DECREMENT CREDIT BALANCE
;       6288              TEMP = .RDRX_ADDR [RCIP, RC_ALL];                   ! READ IP TO FORCE PORT TO POLL
;       6289              SETPRI (PRIOO);                                     ! LOWER PRIORITY
;       6290              return SUCCESS;
;       6291                  end
;       6293          else
;       6294              return FAILURE;                                     ! IF DEVICE IS NOT ONLINE
;       6295
;       6296          end;                                                    ! ROUTINE SEND


                                        .SBTTL  SEND GLOBAL ROUTINES
000000  004137  000000G          SEND::  JSR     R1,$SAVE2                   ;                       6204
000004  005746                          TST     -(SP)
000006  013701  000000G                  MOV     DCT.ADDR,R1                 ;                       6229
000012  005711                          TST     (R1)
000014  100003                          BPL     1$
000016  121127  000004                  CMPB    (R1),#4                     ;                       6230
000022  103416                          BLO     2$
000024  016646  000012          1$:     MOV     12(SP),-(SP)                ; INDEX,*               6231
000030  012746  000104                  MOV     #104,-(SP)
000034  004737  000000G                  JSR     PC,BL$MUL
000040  022626                          CMP     (SP)+,(SP)+
000042  126027  000022G 000004          CMPB    MSCP.PKT+22(R0),#4
000050  001156                          BNE     9$
000052  121127  000004                  CMPB    (R1),#4                     ;                       6232
000056  103153                          BHIS    9$
000060  016646  000012          2$:     MOV     12(SP),-(SP)                ; INDEX,*               6235
```

```
000064  012746  000104                    MOV     #104, (SP)
000070  004737  000000G                   JSR     PC,BL$MUL
000074  010002                            MOV     R0,R2
000076  022626                            CMP     (SP)+,(SP)+
000100  005000                            CLR     R0
000102  156200  000022G                   BISB    MSCP.PKT+22(R2),R0
000106  020027  000020                    CMP     R0,#20
000112  001436                            BEQ     3$
000114  020027  000011                    CMP     R0,#11                  ;                       6236
000120  001433                            BEQ     3$
000122  020027  000041                    CMP     R0,#41                  ;                       6237
000126  001430                            BEQ     3$
000130  020027  000004                    CMP     R0,#4                   ;                       6238
000134  001425                            BEQ     3$
000136  020027  000042                    CMP     R0,#42                  ;                       6239
000142  001422                            BEQ     3$
000144  020027  000004                    CMP     R0,#4                   ;                       6240
000150  001417                            BEQ     3$
000152  020027  000005                    CMP     R0,#5                   ;                       6241
000156  001414                            BEQ     3$
000160  020027  000001                    CMP     R0,#1                   ;                       6242
000164  001411                            BEQ     3$
000166  020027  000003                    CMP     R0,#3                   ;                       6243
000172  001406                            BEQ     3$
000174  020027  000006                    CMP     R0,#6                   ;                       6244
000200  001403                            BEQ     3$
000202  020027  000002                    CMP     R0,#2                   ;                       6245
000206  001077                            BNE     4$                      ;                       6235
000210  104422                    3$:     TRAP    22                      ;                       6254
000212  005762  000004G                   TST     MSCP.PKT+4(R2)          ;                       6256
000216  001003                            BNE     4$
000220  005737  000000G                   TST     CREDIT.BAL              ;                       6257
000224  001004                            BNE     5$
000226  023727  000000G 000001    4$:     CMP     CREDIT.BAL,#1           ;                       6258
000234  101765                            BLOS    3$
000236  013700  000000G             5$:     MOV     CRN.LOW,R0              ;                       6260
000242  005200                            INC     R0
000244  010037  000000G                   MOV     R0,CRN.LOW
000250  010062  000012G                   MOV     R0,MSCP.PKT+12(R2)
000254  001002                            BNE     6$                      ;                       6262
000256  005237  000000G                   INC     CRN.HIGH                ;                       6263
000262  013762  000000G 000014G    6$:     MOV     CRN.HIGH,MSCP.PKT+14(R2)  ;                     6265
000270  013700  000000G                   MOV     DCT.ADDR,R0             ;                       6267
000274  016001  000020                    MOV     20(R0),R1               ; *,SLOT.ADDR
000300  104422                    7$:     TRAP    22                      ;                       6269
000302  032761  100000  000002            BIT     #-100000,2(R1)          ; *,*(SLOT.ADDR)        6270
000310  001373                            BNE     7$
000312  012700  000340                    MOV     #340,R0                 ;                       6272
000316  104441                            TRAP    41
000320  016221  000000G                   MOV     MSCP.PKT(R2),(R1)+      ; *,SLOT.ADDR           6274
000324  016211  000002G                   MOV     MSCP.PKT+2(R2),(R1)     ; *,SLOT.ADDR           6276
000330  042711  040000                    BIC     #40000,(R1)             ; *,SLOT.ADDR           6277
000334  052721  100000                    BIS     #100000,(R1)+           ; *,SLOT.ADDR           6278
000340  013700  000000G                   MOV     DCT.ADDR,R0             ;                       6281
000344  020160  000012                    CMP     R1,12(R0)               ; SLOT.ADDR,*
000350  101402                            BLOS    8$
```

```
000352  016001  000010                  MOV     10(R0),R1                       ; *.SLOT.ADDR                   6283
000356  010160  000020          8$:     MOV     R1,20(R0)                       ; SLOT.ADDR,*                   6285
000362  105210                          INCB    (R0)                            ;                               6286
000364  005337  000000G                 DEC     CREDIT.BAL                      ;                               6287
000370  017716  000000G                 MOV     @RDRX.ADDR,(SP)                 ; *,RC.REG                      6288
000374  005000                          CLR     R0                              ;                               6289
000376  104441                          TRAP    41
000400  012700  000001                  MOV     #1,R0                           ;                               6235
000404  000401                          BR      10$                             ;                               6223
000406  005000          9$:     CLR     R0
000410  005726          10$:    TST     (SP)+                           ;                               6204
000412  000207                          RTS     PC

; Routine Size:  134 words,     Routine Base:  $CODE$ + 6472
; Maximum stack depth per invocation:  7 words
```

```
;          6297  global routine WAIT : novalue =
;          6298
;          6299  !+
;          6300  !       THE PURPOSE OF THIS ROUTINE IS TO KILL TIME UNTIL AN RDRX INTERRUPT
;          6301  !       RESULTS IN A RETURN PACKET INDEX BEING DEPOSITED INTO THE I/O DONE
;          6302  !       QUEUE (IODQ).
;          6303  !
;          6304
;          6305      do
;          6306          BREAK                                   ! BREAK FOR ACT
;          6307      until .IODQ IN neq .IODQ_OUT;


                                        .SBTTL  WAIT GLOBAL ROUTINES
000000  104422                    WAIT::
000000                            1$:   TRAP    22                        ;                6305
000002  023737  000000G 000000G         CMP     IODQ.IN,IODQ.OUT          ;                6307
000010  001773                        BEQ     1$
000012  000207                        RTS     PC                        ;                6297

; Routine Size:  6 words,      Routine Base:  $CODE$ + 7106
; Maximum stack depth per invocation:  2 words
```

```
;       6308  %sbttl  ERROR MESSAGE SUBROUTINES
;       6309
;       6310  routine EMS_SA : novalue *
;       6311
;       6312  !+
;       6313  !       THIS ROUTINE PRINTS (EXTENDED) THE GLOBAL DATUM "SA_REG" WHICH CONTAINS
;       6314  !       THE CONTENTS OF THE SA REGISTER.
;       6315  !
;       6316
;       6317      if .SA_REG eql %o'177777'                                   ! IF CONTROLLER TIME OUT
;       6318      then
;       6319          begin
;       6320          PRINTX (CRLF);
;       6321          PRINTX (ASTERISK);
;       6322          PRINTX (.CNTR_ERR [0]);
;       6323          end
;       6324      else
;       6325
;       6326          if (.SA_REG and %o'003777') lequ 22                     ! IF GENERIC CONTROLLER ERROR
;       6327          then
;       6328              begin
;       6329              PRINTX (CRLF);
;       6330              PRINTX (ASTERISK);
;       6331              PRINTX (.CNTR_ERR [.SA_REG and %o'003777']);
;       6332              end
;       6333          else
;       6334
;       6335              if ((.SA_REG and %o'003777') - 400) lequ 6          ! IF RDRX SPECIFIC CONTROLLER ERROR
;       6336              then
;       6337                  begin
;       6338                  PRINTX (CRLF);
;       6339                  PRINTX (ASTERISK);
;       6340                  PRINTX (.RDRX_ERR [(.SA_REG and %o'003777') - 400]);
;       6341                  end
;       6342              else
;       6343                  PRINTX (XX14, .SA_REG);                         ! JUST PRINT CONTENTS OF SA


                                .SBTTL  EMS.SA ERROR MESSAGE SUBROUTINES
000000  010146                EMS.SA: MOV     R1,-(SP)                ;               6310
000002  013701  000000G               MOV     SA.REG,R1               ;               6317
000006  020127  177777               CMP     R1,#-1
000012  001023                       BNE     1$
000014  012746  000000G               MOV     #CRLF,-(SP)             ;               6320
000020  012746  000001               MOV     #1,-(SP)
000024  010600                       MOV     SP,R0                   ; SP,*
000026  10  15                       TRAP    15
000030  012716  000000G               MOV     #ASTERISK,(SP)          ;               6321
000034  012746  000001               MOV     #1,-(SP)
000040  010600                       MOV     SP,R0                   ; SP,*
000042  104415                       TRAP    15
000044  013716  000000G               MOV     CNTR.ERR,(SP)           ;               6322
000050  012746  000001               MOV     #1,-(SP)
000054  010600                       MOV     SP,R0                   ; SP,*
000056  104415                       TRAP    15
000060  000475                       BR      3$                      ;               6319
```

```
000062  010100                          1$      MOV     R1,R0                           ;                               6326
000064  042700  174000                          BIC     #174000,R0
000070  020027  000026                          CMP     R0,#26
000074  101030                                  BHI     2$
000076  012746  000000G                         MOV     #CRLF,-(SP)                     ;                               6329
000102  012746  000001                          MOV     #1,-(SP)
000106  010600                                  MOV     SP,R0                           ; SP,*
000110  104415                                  TRAP    15
000112  012716  000000G                         MOV     #ASTERISK,(SP)                  ;                               6330
000116  012746  000001                          MOV     #1,-(SP)
000122  010600                                  MOV     SP,R0                           ; SP,*
000124  104415                                  TRAP    15
000126  013700  000000G                         MOV     SA.REG,R0                       ;                               6331
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$                              ;                               6328
000156  010100                          2$:     MOV     R1,R0                           ;                               6335
000160  042700  174000                          BIC     #174000,R0
000164  162700  000620                          SUB     #620,R0
000170  020027  000006                          CMP     R0,#6
000174  101031                                  BHI     4$
000176  012746  000000G                         MOV     #CRLF,-(SP)                     ;                               6338
000202  012746  000001                          MOV     #1,-(SP)
000206  010600                                  MOV     SP,R0                           ; SP,*
000210  104415                                  TRAP    15
000212  012716  000000G                         MOV     #ASTERISK,(SP)                  ;                               6339
000216  012746  000001                          MOV     #1,-(SP)
000222  010600                                  MOV     SP,R0                           ; SP,*
000224  104415                                  TRAP    15
000226  013700  000000G                         MOV     SA.REG,R0                       ;                               6340
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$:     TST     (SP)+                           ;                               6337
000256  000407                                  BR      5$                              ;                               6335
000260  010146                          4$:     MOV     R1,-(SP)                        ;                               6343
000262  012746  000000G                         MOV     #XX14,-(SP)
000266  012746  000002                          MOV     #2,-(SP)
000272  010600                                  MOV     SP,R0                           ; SP,*
000274  104415                                  TRAP    15
000276  062706  000006                  5$:     ADD     #6,SP                           ;                               631~
000302  012601                                  MOV     (SP)+,R1                        ;                               6310
000304  000207                                  RTS     PC

; Routine Size:  99 words,      Routine Base:  $CODE$ + 7122
; Maximum stack depth per invocation:  7 words
```

```
;       6344  routine EMS_SBC : novalue =
;       6345
;       6346  !+
;       6347  !       THIS ROUTINE PRINTS  THE GLOBAL DATUM 'SB_CODE" (SUB-CODE) IF
;       6348  !       EITHER THE STATUS CODE (ST_CODE) OR THE SUB-CODE IS NON ZERO. (A
;       6349  !       NON-ZERO SUB-CODE ALWAYS HAS SIGNIFICANCE, WHEREAS A ZERO SUB CODE ONLY
;       6350  !       HAS MEANING WITH A NON-ZERO STATUS CODE).
;       6351  !
;       6352
;       6353      begin
;       6354
;       6355      if (.ST_CODE or .SB_CODE) neq 0                                   ! PRINT SUB-CODE ONLY ON ERROR
;       6356      then
;       6357          begin
;       6358          PRINTB (XX16);                                                ! SUB CODE :
;       6359
;       6360          case .ST_CODE from ST_SUC to ST_DRV of
;       6361              set
;       6362
;       6363              [ST_SUC]:           if .SB_CODE lequ 16                   ! SUCCESS SUB CODES
;       6364                                  then
;       6365                                      PRINTB (.TBL_SUC [.SB_CODE]);
;       6366
;       6367              [ST_CMD]:           PRINTB (EX_OP, .SB_CODE / 8);         ! INVALID COMMAND
;       6368
;       6369              [ST_ABO]:           ;                                     ! COMMAND ABORTED
;       6370
;       6371              [ST_OFL]:           if .SB_CODE lequ 8                    ! UNIT OFFLINE
;       6372                                  then
;       6373                                      PRINTB (.TBL_OFL [.SB_CODE]);
;       6374
;       6375              [ST_AVL]:           ;                                     ! UNIT AVAILABLE
;       6376
;       6377              [ST_MFE]:           if .SB_CODE lequ 10                   ! MEDIA FORMAT ERROR
;       6378                                  then
;       6379                                      PRINTB (.TBL_MFE [.SB_CODE]);
;       6380
;       6381              [ST_WPT]:           if (.SB_CODE / 128) lequ 2            ! WRITE PROTECTED
;       6382                                  then
;       6383                                      PRINTB (.TBL_WPT [(.SB_CODE / 128)]);
;       6384
;       6385              [ST_CMP]:           ;                                     ! COMPARE ERROR
;       6386
;       6387              [ST_DAT]:           if .SB_CODE lequ 15                   ! DATA ERROR
;       6388                                  then
;       6389                                      PRINTB (.TBL_DAT [.SB_CODE]);
;       6390
;       6391              [ST_HST]:           if .SB_CODE lequ 4                    ! HOST ACCESS ERROR
;       6392                                  then
;       6393                                      PRINTB (.TBL_HST [.SB_CODE]);
;       6394
;       6395              [ST_CNT]:           if .SB_CODE lequ 3                    ! CONTROLLER ERROR
;       6396                                  then
;       6397                                      PRINTB (.TBL_CNT [.SB_CODE]);
;       6398
;       6399              [ST_DRV]:           if .SB_CODE lequ 8                    ! DRIVE ERROR
```

```
;        6400                                    then
;        6401                                        PRINTB (.TBL_DRV [.SB_CODE]);
;        6402
;        6403                [outrange]:         PRINTB (EX_OP, .SB_CODE);              ! JUST PRINT SUB CODE IF NO MATCH
;        6404                tes;
;        6405
;        6406            end;
;        6407
;        6408        end;


                                        .SBTTL  EMS.SBC ERROR MESSAGE SUBROUTINES
000000  013700  000000G          EMS.SBC:MOV     ST.CODE,R0                      ;                                       6355
000004  053700  000000G                  BIS     SB.CODE,R0
000010  001001                           BNE     1$
000012  000207                           RTS     PC
000014  012746  000000G          1$:     MOV     #XX16, (SP)                     ;                                       6358
000020  012746  000001                   MOV     #1,-(SP)
000024  010600                           MOV     SP,R0                           ; SP,*
000026  104414                           TRAP    14
000030  013700  000000G                  MOV     ST.CODE,R0                      ;                                       6360
000034  020027  000013                   CMP     R0,#13
000040  101003                           BHI     3$
000042  006300                           ASL     R0
000044  066007  000000'                   ADD     P.AAA(R0),PC                    ; Case dispatch
000050  013716  000000G          3$:     MOV     SB.CODE,(SP)                    ;                                       6403
000054  012746  000000G                  MOV     #EX.OP,-(SP)
000060  012746  000002                   MOV     #2,-(SP)
000064  010600                           MOV     SP,R0                           ; SP,*
000066  104414                           TRAP    14
000070  022626                           CMP     (SP)+,(SP)+
000072  000435                           BR      6$                              ;                                       6360
000074  023727  000000G 000020   4$:     CMP     SB.CODE,#20                     ;                                       6363
000102  101165                           BHI     14$
000104  013700  000000G                  MOV     SB.CODE,R0                      ;                                       6365
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$:     MOV     SB.CODE,(SP)                    ;                                       6367
000134  012746  000010                   MOV     #10,-(SP)
000140  004737  000000G                  JSR     PC,BL$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$:     BR      16$                             ;                                       6360
000170  023727  000000G 000010   7$:     CMP     SB.CODE,#10                     ;                                       6371
000176  101142                           BHI     16$
000200  013700  000000G                  MOV     SB.CODE,R0                      ;                                       6373
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$:     CMP     SB.CODE,#12             ;                       6377
000232  101124                                  BHI     16$
000234  013700  000000G                         MOV     SB.CODE,R0              ;                       6379
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$:     MOV     SB.CODE,(SP)            ;                       6381
000264  012746  000200                          MOV     #200,(SP)
000270  004737  000000G                         JSR     PC,BL$DIV
000274  005726                                  TST     (SP)+
000276  020027  000002                          CMP     R0,#2
000302  101100                                  BHI     16$
000304  006300                                  ASL     R0                      ;                       6383
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$:    CMP     SB.CODE,#17             ;                       6387
000332  101064                                  BHI     16$
000334  013700  000000G                         MOV     SB.CODE,R0              ;                       6389
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$:    CMP     SB.CODE,#4              ;                       6391
000366  101046                                  BHI     16$
000370  013700  000000G                         MOV     SB.CODE,R0              ;                       6393
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$:    CMP     SB.CODE,#3              ;                       6395
000422  101030                                  BHI     16$
000424  013700  000000G                         MOV     SB.CODE,R0              ;                       6397
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$:    CMP     SB.CODE,#10             ;                       6399
000456  101012                          14$:    BHI     16$
000460  013700  000000G                         MOV     SB.CODE,R0              ;                       6401
000464  006300                                  ASL     R0
```

```
0004[illegible]  016016  000202                    MOV     TBL.DRV(R0),(SP)
0004[illegible]  012746  000001                    MOV     #1,(SP)
000476  010600                                     MOV     SP,R0                   ; SP,R0
000500  104414                                     TRAP    14
000502  005726                             15$:    TST     (SP)+
000504  022626                             16$:    CMP     (SP)+,(SP)+             ;                       6357
000506  000207                                     RTS     PC                      ;                       6344

; Routine Size:  164 words,    Routine Base:  $CODE$ + 7430
; Maximum stack depth per invocation:  7 words


000000                                           .PSECT  $PLIT$,  RO ,  D

                                         P.AAA:                                  ; CASE Table for EMS.SBC+0044   6360
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$]
```

```
;       6409  GLOBAL routine EMSCMD : novalue =
;       6410
;       6411  !+
;       6412  !       THIS ROUTINE PRINTS THE ENTIRE RETURN PACKET INCLUDING OPCODE,
;       6413  !       STAUTUS, SUB STATUS,  MODFIERS OR FLAGS, AND ETC.
;       6414  !       THESE FIELDS ARE "TRANSLATED" INTO ENGLISH TEXT IF POSSIBLE
;       6415  !       RATHER THAN PRINTED AS RAW NUMBERS.
;       6416  !
;       6417  !       IMPLICIT INPUTS:
;       6418  !               RP_ADDR   ADDRESS OF THE CURRENT RETURN PACKET
;       6419  !-
;       6420  begin
;       6421
;       6422  OWN
;       6423          EBH_TB1 : VECTOR [7] INITIAL (EBH_30,EBH_44,EBH_45,
;       6424                                        EBH_46,EBH 47,EBH_48,EBH_49);
;       6425
;       6426                  ! TABLE OF BASIC, HARD ERROR MESSAGE ADDRESSES, INDEXED BY STATUS CODE
;       6427
;       6428  PRINTB (XX13, .CDISK);                  ! "DISK XXX"
;       6429  !PRINTB (XX36, .CRN_LOW);                                ! EXPECTED CRN : XXXXXX
;       6430  PRINTX (XX35, .RP_ADDR [CRF_LO]);                       ! RECEIVED CRN : XXXXXX
;       6431  printX (xx29);                                          ! "message type:"
;       6432  SELECTU (.RP_ADDR [MESTYP]) OF
;       6433                  SET
;       6434                  [%o'0']: PRINTX (EX_SEQ);       ! "SEQUENTIAL"
;       6435                  [%o'1']: PRINTX (EX_DGM);       ! "DATAGRAM"
;       6436                  [%o'2']: PRINTX (EX_CRD);       ! "CREDIT NOTIFICATION"                 PACKET TYPE
;       6437                  [%o'15']: PRINTX (EX_MTN);      ! "MAINTENANCE"
;       6438                  [OTHERWISE]: PRINTX (XX37, .RP_ADDR [MESTYP]);  ! UNKOWN MESSAGE TYPE
;       6439                  TES;
;       6440
;       6441  PRINTB (XX17);                          ! "COMMAND: "
;       6442
;       6443  SELECTU (.RP_ADDR [conid]) OF
;       6444  SET
;       6445  [%o'2']:
;       6446      BEGIN
;       6447      PRINTB (XX18);                                      ! PRINTS  DUP-
;       6448      SELECTU (.RP_ADDR [ENDCOD]) OF
;       6449                  SET
;       6450                  [%o'201']: PRINTB (EX_GDS);
;       6451                  [%o'202']: PRINTB (EX_ESP);             ! PRINTS A COMMAND
;       6452                  [%o'203']: PRINTB (EX_ELP);
;       6453                  [%o'204']: PRINTB (EX_RCD);
;       6454                  [%o'205']: PRINTB (EX_SDD);
;       6455                  [%o'206']: PRINTB (EX_ABP);
;       6456                  [OTHERWISE]: PRINTB (EX_OP, .RP_ADDR [ENDCOD]); ! PRINT ENDCODE VALUE
;       6457                  TES;
;       6458      printb (xx15);                                              ! "status:"
;       6459      IF (.RP_ADDR [STSCOD] GEQU 0) AND (.RP_ADDR [STSCOD] LEQU 7)        ! IF STATUS CODE IS WITHIN RANGE
;       6460      THEN        PRINTB (.EBH_TB1 [.RP_ADDR [STSCOD]])                   ! PRINTB APPROPRIATE MESSAGE
;       6461      ELSE    PRINTB (ex_op, .RP_ADDR [STSCOD]);              ! JUST PRINT STATUS CODE
;       6462
;       6463      IF .RP_ADDR [ENDCOD] EQL %O'204' or
;       6464         .RP_ADDR [ENDCOD] EQL %O'205'                        ! IF A SEND DATA OR RECEIVE DATA COMMAND THEN
```

```
;       6465        then
;       6466            begin
;       6467            PRINTX (XX25, .RP_ADDR [BCNT_LO]);                  ! FOR ANY "ACTUAL # OF BYTES TRANSFERRED: xxxxx."
;       6468            PRINTX (XX26, .RP_ADDR [BUFF_1], .RP_ADDR [BUFF_0]);    ! "I/O BUFFER DESCRIPTOR: xxxxxx xxxxxx"
;       6469            EMS_DUP ();                                             ! prints contents of dup packet
;       6470            end;
;       6471        IF .RP_ADDR [ENDCOD] EQL %O'201'
;       6472        then
;       6473            begin
;       6474            if BIT_TST (RP_ADDR [9, 8, 1, 0], 1)
;       6475                then PRINTB (df_0);
;       6476            if BIT_TST (RP_ADDR [9, 9, 1, 0], 1)
;       6477                then PRINTB (df_1);
;       6478            if BIT_TST (RP_ADDR [9, 10, 1, 0], 1)
;       6479                then PRINTB (df_2);
;       6480            if BIT_TST (RP_ADDR [9, 11, 1, 0], 1)
;       6481                then PRINTB (df_3);
;       6482            end;
;       6483
;       6484        IF .RP_ADDR [ENDCOD] EQL %O'203'                        ! IF A GET DUST STATUS OR EXEC. LOCAL PRG COMMAND TH
EN
;       6485        then
;       6486            begin
;       6487            if BIT_TST (RP_ADDR [9, 8, 1, 0], 1)
;       6488                then PRINTB (df_4);
;       6489            if BIT_TST (RP_ADDR [9, 9, 1, 0], 1)
;       6490                then PRINTB (df_5);
;       6491            if BIT_TST (RP_ADDR [9, 10, 1, 0], 1)
;       6492                then PRINTB (df_6);
;       6493            if BIT_TST (RP_ADDR [9, 11, 1, 0], 1)
;       6494                then PRINTB (df_7);
;       6495            end;
;       6496        PRINTX (XX23, .CST_ADDR [.CUOFF + 3, D_DBN], .CST_ADDR [.CUOFF + 3, D_DBN]);        ! "DBN: XXXXXX."
;       6497        END;
;       6498    [%o'0']:
;       6499        BEGIN
;       6500        PRINTB (XX19);                                  !PRINTS -MSCP                                   !MSC
P
;       6501        SELECTU (.RP_ADDR [ENDCOD]) OF
;       6502                    SET
;       6503                    [%o'204']: PRINTB (EX_SCC);
;       6504                    [%o'211']: PRINTB (EX_ONL);
;       6505                    [%o'220']: PRINTB (EX_ACC);
;       6506                    [%o'241']: PRINTB (EX_RD);                      ! PRINTS THE COMMAND
;       6507                    [%o'242']: PRINTB (EX_WRT);
;       6508                    [OTHERWISE]: PRINTB (EX_OP, .RP_ADDR [ENDCOD]); ! PRINT ENDCODE VALUE
;       6509                    TES;
;       6510        if .RP_ADDR [CMDMOD] eql MD_CMP THEN PRINTB (XX20); ! PRINTS THE MODIFIER IF NECESSARY
;       6511
;       6512        PRINTB (XX15);                                  ! STATUS:
;       6513        if (.ST_CODE gtru 0) and                        ! IF STATUS CODE IS WITHIN RANGE
;       6514           (.ST_CODE lequ 11)
;       6515        then
;       6516            PRINTB (.ERR_COD [.ST_CODE   1])                ! PRINTB APPROPRIATE MESSAGE
;       6517        else
;       6518
;       6519            if .ST_CODE eql ST_DIA
;       6520            then
```

```
;       6521                    PRINTB (.ERR COD [11])                  ! MESSAGE FROM INTERNAL DIAGNOSTICS
;       6522                ELSE
;       6523                    PRINTB (EX_OP, .ST CODE);               ! JUST PRINT STATUS CODE WHEN NO MATCH
;       6524
;       6525            EMS SBC ();                                     ! PRINTS STATUS SUB CODE
;       6526
;       6527            IF .RP ADDR [ENDCOD] EQLU %o'220' OR                                                    ! MSCP acces
s command
;       6528                   .RP_ADDR [ENDCOD] EQLU %o'241' OR                                                ! mscp read
;       6529                   .RP_ADDR [ENDCOD] EQLU %o'242'                                                   ! mscp write
;       6530            THEN
;       6531                    begin
;       6532                    printX (XX24, .rp_addr [CBCNT_LO]);             ! "BYTE COUNT IN COMMAND: XXXXXXX"
;       6533                    PRINTX (XX25, .RP_ADDR [BCNT LO]);              ! FOR ANY "ACTUAL # OF BYTES TRANSFERRED: XXXXX.'
;       6534                    PRINTX (XX26, .RP_ADDR [BUFF_1],  .RP_ADDR [BUFF_0]);   ! "I/O BUFFER DESCRIPTOR: XXXXXX XXXXXX"
;       6535                    if BIT_TST (RP_ADDR [FLAGS], EF_0)      ! IF BAD BLOCK REPORTED
;       6536                        then
;       6537                            PRINTB (XX21, .RP_ADDR [BBLK_LO])       ! "BAD BLOCK REPORTED: XXXXXX."
;       6538                        else
;       6539                            printX (XX22, .RP_ADDR [LBN_LO], .RP ADDR [LBN_LO]);            ! "LBN: XXXXXX'
;       6540                    PRINTB (XX41);
;       6541                    if BIT_TST (RP_ADDR [FLAGS], EF_1)      ! IF BAD BLOCK UNREPORTED
;       6542                        then PRINTB (F_1);
;       6543                    if BIT_TST (RP_ADDR [FLAGS], EF_2)      ! IF ERROR LOG GENERATED
;       6544                        then PRINTB (F_2);
;       6545                    if BIT_TST (RP_ADDR [FLAGS], EF_3)      ! IF SERIOUS EXCEPTION
;       6546                        then PRINTB (F_3);
;       6547                    END;
;       6548            IF .RP_ADDR [ENDCOD] EQLU %o'204'                                                       ! MSCP SET CTLR CHAR
 command
;       6549            THEN
;       6550                begin
;       6551                    PRINTB (XX39);
;       6552                    if BIT_TST (RP_ADDR [BCNT_HI], EF_4)    ! IF
;       6553                        then PRINTB (F_4);
;       6554                    if BIT_TST (RP_ADDR [BCNT_HI], EF_5)    ! IF
;       6555                        then PRINTB (F_5);
;       6556                    if BIT_TST (RP_ADDR [BCNT_HI], EF_6)    ! IF
;       6557                        then PRINTB (F_6);
;       6558                    if BIT_TST (RP_ADDR [BCNT_HI], EF_7)    ! IF
;       6559                        then PRINTB (F_7);
;       6560                    if BIT_TST (RP_ADDR [BCNT_HI], EF_8)    ! IF
;       6561                        then PRINTB (F_8);
;       6562                    if BIT_TST (RP_ADDR [BCNT_HI], EF_9)    ! IF
;       6563                        then PRINTB (F_9);
;       6564                    if BIT_TST (RP_ADDR [BCNT_HI], EF_10)   ! IF
;       6565                        then PRINTB (F_10);
;       6566                end;
;       6567            IF .RP_ADDR [ENDCOD] EQLU %o'211'
;       6568            THEN                                                                    ! MSCP ONLINE comman
d
;       6569                begin
;       6570                    PRINTB (XX40);
;       6571                    if BIT_TST (RP_ADDR [BCNT_HI], EF_11)   ! IF
;       6572                        then PRINTB (F_11);
;       6573                    if BIT_TST (RP_ADDR [BCNT_HI], EF_12)   ! IF
;       6574                        then PRINTB (F_12);
;       6575                    if BIT_TST (RP_ADDR [BCNT_HI], EF_13)   ! IF
;       6576                        then PRINTB (F_13);
```

```
;       6577                    if BIT_TST (RP_ADDR [BCNT_HI], EF_14)   ! IF
;       6578                        then PRINTB (F_14);
;       6579                    if BIT_TST (RP_ADDR [BCNT_HI], EF_15)   ! IF
;       6580                        then PRINTB (F_15);
;       6581                    if BIT_TST (RP_ADDR [BCNT_HI], EF_16)   ! IF
;       6582                        then PRINTB (F_16);
;       6583                    if BIT_TST (RP_ADDR [BCNT_HI], EF_17)   ! IF
;       6584                        then PRINTB (F_17);
;       6585                    if BIT_TST (RP_ADDR [BCNT_HI], EF_18)   ! IF
;       6586                        then PRINTB (F_18);
;       6587                    if BIT_TST (RP_ADDR [BCNT_HI], EF_19)   ! IF
;       6588                        then PRINTB (F_19);
;       6589                    if BIT_TST (RP_ADDR [BCNT_HI], EF_20)   ! IF
;       6590                        then PRINTB (F_20);
;       6591                    if BIT_TST (RP_ADDR [BCNT_HI], EF_21)   ! IF
;       6592                        then PRINTB (F_21);
;       6593                end;
;       6594           END;
;       6595       [otherwise]:
;       6596           BEGIN                                                                    ! UNKNOWN CO
NNID
;       6597           PRINTB (XX38, .RP_ADDR [CONID]);                     ! PRINTS UNKNOWN CONNECTION ID
;       6598           SELECTU (.RP_ADDR [ENDCOD]) OF
;       6599                   SET
;       6600                   [%o'204']: PRINTB (EX_SCC);
;       6601                   [%o'211']: PRINTB (EX_ONL);                  ! PRINTS THE COMMAND IF RECOGNIZED
;       6602                   [%o'220']: PRINTB (EX_ACC);
;       6603                   [%o'241']: PRINTB (EX_RD);
;       6604                   [%o'242']: PRINTB (EX_WRT);
;       6605                   [%o'211']: PRINTB (EX_ONL);
;       6606                   [%o'220']: PRINTB (EX_ACC);
;       6607                   [%o'241']: PRINTB (EX_RD);
;       6608                   [%o'242']: PRINTB (EX_WRT);
;       6609                   [OTHERWISE]: PRINTB (EX_OP, .RP_ADDR [ENDCOD]); ! PRINT ENDCODE VALUE
;       6610                   TES;
;       6611           if .RP_ADDR [CMDMOD] eql MD_CMP THEN PRINTB (XX20); ! PRINTS MODIFIER IF NECESSARY
;       6612           printb (xx15);
;       6613           PRINTB (ex_op, .RP_ADDR [STSCOD]);                   ! PRINTS STATUS CODE IN OCTA' FORM
;       6614           printb (xx16);
;       6615           PRINTB (ex_op, .RP_ADDR [SUBCOD]);                   ! PRINTS STATUS SUB-CODE IN OCTAL FORM
;       6616
;       6617                           !+
;       6618                           !       THIS ROUTINE PRINTS (EXTENDED) BOTH BYTE COUNT FIELDS OF THE CURRENT
;       6619                           !       RETURN PACKET: THE BYTE COUNT FROM THE COMMAND ENVELOPE AND THE
;       6620                           !       ACTUAL NUMBER OF BYTES TRANSFERRED (FROM THE RESPONSE ENVELOPE).
;       6621                           !-
;       6622
;       6623           IF .RP_ADDR [ENDCOD] EQLU %o'220' OR                 ! MSCP access command
;       6624               .RP_ADDR [ENDCOD] EQLU %o'241' OR                ! mscp read
;       6625               .RP_ADDR [ENDCOD] EQLU %o'242'                   ! mscp write
;       6626           THEN
;       6627               begin
;       6628                       printX (XX24, .rp_addr [CBCNT_LO]);      ! "BYTE COUNT IN COMMAND: XXXXXXX"
;       6629                       PRINTX (XX25, .RP_ADDR [BCNT_LO]);       ! FOR ANY "ACTUAL # OF BYTES TRANSFERRED: XXXXX."
;       6630                       PRINTX (XX26, .RP_ADDR [BUFF_1], .RP_ADDR [BUFF_0]);   ! "I/O BUFFER DESCRIPTOR: XXXXXX XXXXXX'
;       6631                       if BIT_TST (RP_ADDR [FLAGS], EF_0)       ! IF BAD BLOCK REPORTED
;       6632                           then
```

```
;       6633                        PRINTB (XX21, .RP_ADDR [BBLK_LO])       ! "BAD BLOCK REPORTED: XXXXXX."
;       6634                    else
;       6635                        printX (xx22, .RP_ADDR [LBN_LO], .RP_ADDR [LBN_LO]);            ! "LBN: XXXXXX"
;       6636            end;
;       6637        IF .RP_ADDR [ENDCOD] EQLU %o'204' OR                                ! dup receive data
;       6638            .RP_ADDR [ENDCOD] EQLU %o'205'                  ! dup send data
;       6639        THEN
;       6640            begin
;       6641            PRINTX (XX25, .RP_ADDR [BCNT_LO]);                              ! FOR ANY "ACTUAL # OF BYTES TRANSFERRED: XXXXX."
;       6642            PRINTX (XX26, .RP_ADDR [BUFF_1], .RP_ADDR [BUFF_0]);    ! 'I/O BUFFER DESCRIPTOR: XXXXXX XXXXXX"
;       6643            PRINTX (XX23, .CST_ADDR [.CUOFF + 3, D_DBN], .CST_ADDR [.CUOFF + 3, D_DBN]);        ! "DBN. XXXXXX."
;       6644            ems_dup ();                                                     ! prints dup packet material
;       6645            end;
;       6646        END;
;       6647    tes;                        !+
;       6648                                !       PRINTS RETURN OR MESSAGE PACKET IN A BLOCK OF OCTAL WORDS
;       6649                                !-
;       6650
;       6651    !PRINTX (XX27);                             ! "CONTENTS OF PACKET:"
;       6652    !EMS_BLK (.RP_ADDR, PKT_LEN);               ! PRINT BLOCK OF WORDS AS LONG AS A MESSAGE PACKET INCASE A PACKET IS USED
;       6653                                                ! INSTEAD OF A RETURN PACKET
;       6654
;       6655    end;                        ! ROUTINE EMSCMD


010140                                  .PSECT  $CODE$,  RO
010140  000000G                EBH.TB1:.WORD   EBH.30
010142  000000G                        .WORD   EBH.44
010144  000000G                        .WORD   EBH.45
010146  000000G                        .WORD   EBH.46
010150  000000G                        .WORD   EBH.47
010152  000000G                        .WORD   EBH.48
010154  000000G                        .WORD   EBH.49


                                    .SBTTL  EMSCMD ERROR MESSAGE SUBROUTINES
000000  004137  000000G    EMSCMD::JSR     R1,$SAVE4                       ;                       6409
000004  013746  000000G            MOV     CDISK,-(SP)                     ;                       6428
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                      ;                       6430
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)                      ;                       6431
000054  012746  000001            MOV     #1,-(SP)
000060  010600                    MOV     SP,R0                           ; SP,*
000062  104415                    TRAP    15
000064  013700  000000G            MOV     RP.ADDR,R0                      ;                       6432
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)            ;                       6434
000126  012746  000001                  MOV     #1,-(SP)
000132  010600                          MOV     SP,R0                   ; SP,*
000134  104415                          TRAP    15
000136  005726                          TST     (SP)+
000140  020227  000001          1$:     CMP     R2,#1                   ;                       6432
000144  001010                          BNE     2$
000146  005001                          CLR     R1
000150  012716  000000G                 MOV     #EX.DGM,(SP)            ;                       6435
000154  012746  000001                  MOV     #1,-(SP)
000160  010600                          MOV     SP,R0                   ; SP,*
000162  104415                          TRAP    15
000164  005726                          TST     (SP)+
000166  020227  000002          2$:     CMP     R2,#2                   ;                       6432
000172  001010                          BNE     3$
000174  005001                          CLR     R1
000176  012716  000000G                 MOV     #EX.CRD,(SP)            ;                       6436
000202  012746  000001                  MOV     #1,-(SP)
000206  010600                          MOV     SP,R0                   ; SP,*
000210  104415                          TRAP    15
000212  005726                          TST     (SP)+
000214  020227  000015          3$:     CMP     R2,#15                  ;                       6432
000220  001010                          BNE     4$
000222  005001                          CLR     R1
000224  012716  000000G                 MOV     #EX.MTN,(SP)            ;                       6437
000230  012746  000001                  MOV     #1,-(SP)
000234  010600                          MOV     SP,R0                   ; SP,*
000236  104415                          TRAP    15
000240  005726                          TST     (SP)+
000242  005701                  4$:     TST     R1                      ;                       6432
000244  001422                          BEQ     5$
000246  013700  000000G                 MOV     RP.ADDR,R0              ;                       6438
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)              ;                       6441
000316  012746  000001                  MOV     #1,-(SP)
000322  010600                          MOV     SP,R0                   ; SP,*
000324  104414                          TRAP    14
```

```
000326  013700  000000G                MOV     RP.ADDR,R0              ;                               6443
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  011644'               JMP     26$
000356  005003                6$:     CLR     R3
000360  012716  000000G               MOV     #XX18,(SP)              ;                               6447
000364  012746  000001                MOV     #1,-(SP)
000370  010600                        MOV     SP,R0                   ; SP,*
000372  104414                        TRAP    14
000374  013700  000000G               MOV     RP.ADDR,R0              ;                               6448
000400  005002                        CLR     R2
000402  156002  000014                BISB    14(R0),R2
000406  012701  177777                MOV     #-1,R1
000412  020227  000201                CMP     R2,#201
000416  001010                        BNE     7$
000420  005001                        CLR     R1
000422  012716  000000G               MOV     #EX.GDS,(SP)            ;                               6450
000426  012746  000001                MOV     #1,-(SP)
000432  010600                        MOV     SP,R0                   ; SP,*
000434  104414                        TRAP    14
000436  005726                        TST     (SP)+
000440  020227  000202        7$:     CMP     R2,#202                 ;                               6448
000444  001010                        BNE     8$
000446  005001                        CLR     R1
000450  012716  000000G               MOV     #EX.ESP,(SP)            ;                               6451
000454  012746  000001                MOV     #1,-(SP)
000460  010600                        MOV     SP,R0                   ; SP,*
000462  104414                        TRAP    14
000464  005726                        TST     (SP)+
000466  020227  000203        8$:     CMP     R2,#203                 ;                               6448
000472  001010                        BNE     9$
000474  005001                        CLR     R1
000476  012716  000000G               MOV     #EX.ELP,(SP)            ;                               6452
000502  012746  000001                MOV     #1,-(SP)
000506  010600                        MOV     SP,R0                   ; SP,*
000510  104414                        TRAP    14
000512  005726                        TST     (SP)+
000514  020227  000204        9$:     CMP     R2,#204                 ;                               6448
000520  001010                        BNE     10$
000522  005001                        CLR     R1
000524  012716  000000G               MOV     #EX.RCD,(SP)            ;                               6453
000530  012746  000001                MOV     #1,-(SP)
000534  010600                        MOV     SP,R0                   ; SP,*
000536  104414                        TRAP    14
000540  005726                        TST     (SP)+
000542  020227  000205        10$:    CMP     R2,#205                 ;                               6448
000546  001010                        BNE     11$
000550  005001                        CLR     R1
000552  012716  000000G               MOV     #EX.SDD,(SP)            ;                               6454
000556  012746  000001                MOV     #1,-(SP)
000562  010600                        MOV     SP,R0                   ; SP,*
000564  104414                        TRAP    14
000566  005726                        TST     (SP)+
```

```
000570  020227  000206          11$:    CMP     R2,#206                         ;                       6448
000574  001010                          BNE     12$
000576  005001                          CLR     R1
000600  012716  000000G               MOV     #EX.ABP,(SP)                    ;                       6455
000604  012746  000001                  MOV     #1,-(SP)
000610  010600                          MOV     SP,R0                           ; SP,*
000612  104414                          TRAP    14
000614  005726                          TST     (SP)+
000616  005701                  12$:    TST     R1                              ;                       6448
000620  001414                          BEQ     13$
000622  013700  000000G               MOV     RP.ADDR,R0                      ;                       6456
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$:    MOV     #XX15,(SP)                      ;                       6458
000656  012746  000001                  MOV     #1,-(SP)
000662  010600                          MOV     SP,R0                           ; SP,*
000664  104414                          TRAP    14
000666  013700  000000G               MOV     RP.ADDR,R0                      ;                       6459
000672  116000  000016                  MOVB    16(R0),R0
000676  042700  177740                  BIC     #177740,R0
000702  020027  000007                  CMP     R0,#7
000706  101010                          BHI     14$
000710  006300                          ASL     R0                              ;                       6460
000712  016016  010140'                 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$                             ;                       6459
000730  010016                  14$:    MOV     R0,(SP)                         ;                       6461
000732  012746  000000G               MOV     #EX.OP,-(SP)
000736  012746  000002                  MOV     #2,-(SP)
000742  010600                          MOV     SP,R0                           ; SP,*
000744  104414                          TRAP    14
000746  005726                          TST     (SP)+
000750  013700  000000G          15$:    MOV     RP.ADDR,R0                      ;                       6463
000754  126027  000014  000204          CMPB    14(R0),#204
000762  001404                          BEQ     16$
000764  126027  000014  000205          CMPB    14(R0),#205                     ;                       6464
000772  001032                          BNE     17$
000774  013700  000000G          16$:    MOV     RP.ADDR,R0                      ;                       6467
001000  016016  000020                  MOV     20(R0),(SP)
001004  012 46  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                      ;                       6468
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              ;                       6469
001054  062706  000012                  ADD     #12,SP                  ;                       6466
001060  013700  000000G          17$:    MOV     RP.ADDR,R0              ;                       6471
001064  126027  000014  000201         CMPB    14(R0),#201
001072  001062                          BNE     21$
001074  032760  000400  000022         BIT     #400,22(R0)             ;                       6474
001102  001407                          BEQ     18$
001104  012716  000000G                MOV     #DF.0,(SP)              ;                       6475
001110  012746  000001                  MOV     #1, (SP)
001114  010600                          MOV     SP,R0                   ; SP,*
001116  104414                          TRAP    14
001120  005726                          TST     (SP)+
001122  013700  000000G          18$:    MOV     RP.ADDR,R0              ;                       6476
001126  032760  001000  000022         BIT     #1000,22(R0)
001134  001407                          BEQ     19$
001136  012716  000000G                MOV     #DF.1,(SP)              ;                       6477
001142  012746  000001                  MOV     #1,-(SP)
001146  010600                          MOV     SP,R0                   ; SP,*
001150  104414                          TRAP    14
001152  005726                          TST     (SP)+
001154  013700  000000G          19$:    MOV     RP.ADDR,R0              ;                       6478
001160  032760  002000  000022         BIT     #2000,22(R0)
001166  001407                          BEQ     20$
001170  012716  000000G                MOV     #DF.2,(SP)              ;                       6479
001174  012746  000001                  MOV     #1,-(SP)
001200  010600                          MOV     SP,R0                   ; SP,*
001202  104414                          TRAP    14
001204  005726                          TST     (SP)+
001206  013700  000000G          20$:    MOV     RP.ADDR,R0              ;                       6480
001212  032760  004000  000022         BIT     #4000,22(R0)
001220  001407                          BEQ     21$
001222  012716  000000G                MOV     #DF.3,(SP)              ;                       6481
001226  012746  000001                  MOV     #1,-(SP)
001232  010600                          MOV     SP,R0                   ; SP,*
001234  104414                          TRAP    14
001236  005726                          TST     (SP)+
001240  013700  000000G          21$:    MOV     RP.ADDR,R0              ;                       6484
001244  126027  000014  000203         CMPB    14(R0),#203
001252  001062                          BNE     25$
001254  032760  000400  000022         BIT     #400,22(R0)             ;                       6487
001262  001407                          BEQ     22$
001264  012716  000000G                MOV     #DF.4,(SP)              ;                       6488
001270  012746  000001                  MOV     #1,-(SP)
001274  010600                          MOV     SP,R0                   ; SP,*
001276  104414                          TRAP    14
001300  005726                          TST     (SP)+
001302  013700  000000G          22$:    MOV     RP.ADDR,R0              ;                       6489
001306  032760  001000  000022         BIT     #1000,22(R0)
001314  001407                          BEQ     23$
001316  012716  000000G                MOV     #DF.5,(SP)              ;                       6490
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              ;                       6491
```

```
001340  032760  002000  000022          BIT     #2000,22(R0)
001346  001407                          BEQ     24$
001350  012716  000000G                 MOV     #DF.6,(SP)              ;                       6492
001354  012746  000001                  MOV     #1,-(SP)
001360  010600                          MOV     SP,R0                   ; SP,*
001362  104414                          TRAP    14
001364  005726                          TST     (SP)+
001366  013700  000000G         24$:    MOV     RP.ADDR,R0              ;                       6493
001372  032760  004000  000022          BIT     #4000,22(R0)
001400  001407                          BEQ     25$
001402  012716  000000G                 MOV     #DF.7,(SP)              ;                       6494
001406  012746  000001                  MOV     #1,-(SP)
001412  010600                          MOV     SP,R0                   ; SP,*
001414  104414                          TRAP    14
001416  005726                          TST     (SP)+
001420  013700  000000G         25$:    MOV     CUOFF,R0                ;                       6496
001424  006300                          ASL     R0
001426  063700  000000G                 ADD     CST.ADDR,R0
001432  005016                          CLR     (SP)
001434  116016  000006                  MOVB    6(R0),(SP)
001440  005046                          CLR     -(SP)
001442  116016  000006                  MOVB    6(R0),(SP)
001446  012746  000000G                 MOV     #XX23,-(SP)
001452  012746  000003                  MOV     #3,-(SP)
001456  010600                          MOV     SP,R0                   ; SP,*
001460  104415                          TRAP    15
001462  062706  000014                  ADD     #14,SP                  ;                       6446
001466  005704                  26$:    TST     R4                      ;                       6443
001470  001402                          BEQ     27$
001472  000137  013700'                 JMP     66$
001476  005003                  27$:    CLR     R3
001500  012716  000000G                 MOV     #XX19,(SP)              ;                       6500
001504  012746  000001                  MOV     #1,-(SP)
001510  010600                          MOV     SP,R0                   ; SP,*
001512  104414                          TRAP    14
001514  013700  000000G                 MOV     RP.ADDR,R0              ;                       6501
001520  005002                          CLR     R2
001522  156002  000014                  BISB    14(R0),R2
001526  012701  177777                  MOV     #-1,R1
001532  020227  000204                  CMP     R2,#204
001536  001010                          BNE     28$
001540  005001                          CLR     R1
001542  012716  000000G                 MOV     #EX.SCC,(SP)            ;                       6503
001546  012746  000001                  MOV     #1,-(SP)
001552  010600                          MOV     SP,R0                   ; SP,*
001554  104414                          TRAP    14
001556  005726                          TST     (SP)+
001560  020227  000211          28$:    CMP     R2,#211                 ;                       6501
001564  001010                          BNE     29$
001566  005001                          CLR     R1
001570  012716  000000G                 MOV     #EX.ONL,(SP)            ;                       6504
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                 ;                       6501
```

```
001612  001010                          BNE     30$
001614  005001                          CLR     R1
001616  012716  000000G                 MOV     #EX.ACC,(SP)                    ;                               6505
001622  012746  000001                  MOV     #1,-(SP)
001626  010600                          MOV     SP,R0                           ; SP,*
001630  104414                          TRAP    14
001632  005726                          TST     (SP)+
001634  020227  000241          30$:    CMP     R2,#241                         ;                               6501
001640  001010                          BNE     31$
001642  005001                          CLR     R1
001644  012716  000000G                 MOV     #EX.RD,(SP)                     ;                               6506
001650  012746  000001                  MOV     #1,-(SP)
001654  010600                          MOV     SP,R0                           ; SP,*
001656  104414                          TRAP    14
001660  005726                          TST     (SP)+
001662  020227  000242          31$:    CMP     R2,#242                         ;                               6501
001666  001010                          BNE     32$
001670  005001                          CLR     R1
001672  012716  000000G                 MOV     #EX.WRT,(SP)                    ;                               6507
001676  012746  000001                  MOV     #1,-(SP)
001702  010600                          MOV     SP,R0                           ; SP,*
001704  104414                          TRAP    14
001706  005726                          TST     (SP)+
001710  005701          32$:    TST     R1                              ;                               6501
001712  001414                          BEQ     33$
001714  013700  000000G                 MOV     RP.ADDR,R0                      ;                               6508
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$:    MOV     RP.ADDR,R0                      ;                               6510
001750  026027  000012  040000          CMP     12(R0),#40000
001756  001007                          BNE     34$
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$:    MOV     #XX15,(SP)                      ;                               6512
002002  012746  000001                  MOV     #1,-(SP)
002006  010600                          MOV     SP,R0                           ; SP,*
002010  104414                          TRAP    14
002012  013700  000000G                 MOV     ST.CODE,R0                      ;                               6513
002016  001413                          BEQ     35$
002020  020027  000013                  CMP     R0,#13                          ;                               6514
002024  101010                          BHI     35$
002026  006300                          ASL     R0                              ;                               6516
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$                             ;                               6513
002046  020027  000037          35$:    CMP     R0,#37                          ;                               6519
```

```
002052  001007                          BNE     36$
002054  013716  000026G                 MOV     ERR.COD+26,(SP)         ;                       6521
002060  012746  000001                  MOV     #1, (SP)
002064  010600                          MOV     SP,R0                   ; SP,*
002066  104414                          TRAP    14
002070  000410                          BR      37$                     ;                       6519
002072  010016                  36$:    MOV     R0,(SP)                 ;                       6523
002074  012746  000000G                 MOV     #EX.OP,-(SP)
002100  012746  000002                  MOV     #2,-(SP)
002104  010600                          MOV     SP,R0                   ; SP,*
002106  104414                          TRAP    14
002110  005726                          TST     (SP)+
002112  004737  007430'         37$:    JSR     PC,EMS.SBC              ;                       6525
002116  013700  000000G                 MOV     RP.ADDR,R0              ;                       6527
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                 ;                       6528
002144  001403                          BEQ     38$
002146  020027  000242                  CMP     R0,#242                 ;                       6529
002152  001147                          BNE     44$
002154  013700  000000G         38$:    MOV     RP.ADDR,R0              ;                       6532
002160  016016  000044                  MOV     44(R0),(SP)
002164  012746  000000G                 MOV     #XX24,-(SP)
002170  012746  000002                  MOV     #2, (SP)
002174  010600                          MOV     SP,R0                   ; SP,*
002176  104415                          TRAP    15
002200  013700  000000G                 MOV     RP.ADDR,R0              ;                       6533
002204  016016  000020                  MOV     20(R0),(SP)
002210  012746  000000G                 MOV     #XX25,-(SP)
002214  012746  000002                  MOV     #2,-(SP)
002220  010600                          MOV     SP,R0                   ; SP,*
002222  104415                          TRAP    15
002224  013700  000000G                 MOV     RP.ADDR,R0              ;                       6534
002230  016016  000024                  MOV     24(R0),(SP)
002234  016046  000026                  MOV     26(R0),-(SP)
002240  012746  000000G                 MOV     #XX26,-(SP)
002244  012746  000003                  MOV     #3,-(SP)
002250  010600                          MOV     SP,R0                   ; SP,*
002252  104415                          TRAP    15
002254  013700  000000G                 MOV     RP.ADDR,R0              ;                       6535
002260  005760  000014                  TST     14(R0)
002264  100011                          BPL     39$
002266  016016  000040                  MOV     40(R0),(SP)             ;                       6537
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$                     ;                       6535
002310  016016  000050         39$:    MOV     50(R0),(SP)             ;                       6539
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$:    MOV     #XX41,(SP)              ;                                       6540
002340  012746  000001                          MOV     #1,-(SP)
002344  010600                          MOV     SP,R0                   ; SP,*
002346  104414                          TRAP    14
002350  013700  000000G                 MOV     RP.ADDR,R0              ;                                       6541
002354  032760  040000  000014         BIT     #40000,14(R0)
002362  001407                          BEQ     41$
002364  012716  000000G                 MOV     #F.1,(SP)               ;                                       6542
002370  012746  000001                  MOV     #1, (SP)
002374  010600                          MOV     SP,R0                   ; SP,*
002376  104414                          TRAP    14
002400  005726                          TST     (SP)+
002402  013700  000000G                 41$:    MOV     RP.ADDR,R0              ;                                       6543
002406  032760  020000  000014         BIT     #20000,14(R0)
002414  001407                          BEQ     42$
002416  012716  000000G                 MOV     #F.2,(SP)               ;                                       6544
002422  012746  000001                  MOV     #1, (SP)
002426  010600                          MOV     SP,R0                   ; SP,*
002430  104414                          TRAP    14
002432  005726                          TST     (SP)+
002434  013700  000000G                 42$:    MOV     RP.ADDR,R0              ;                                       6545
002440  032760  010000  000014         BIT     #10000,14(R0)
002446  001407                          BEQ     43$
002450  012716  000000G                 MOV     #F.3,(SP)               ;                                       6546
002454  012746  000001                  MOV     #1, (SP)
002460  010600                          MOV     SP,R0                   ; SP,*
002462  104414                          TRAP    14
002464  005726                          TST     (SP)+
002466  062706  000024                  43$:    ADD     #24,SP                  ;                                       6547
002472  013700  000000G                 44$:    MOV     RP.ADDR,R0              ;                                       6548
002476  126027  000014  000204         CMPB    14(R0),#204
002504  001144                          BNE     52$
002506  012716  000000G                 MOV     #XX39,(SP)              ;                                       6551
002512  012746  000001                  MOV     #1,-(SP)
002516  010600                          MOV     SP,R0                   ; SP,*
002520  104414                          TRAP    14
002522  013700  000000G                 MOV     RP.ADDR,R0              ;                                       6552
002526  105760  000022                  TSTB    22(R0)
002532  100007                          BPL     45$
002534  012716  000000G                 MOV     #F.4,(SP)               ;                                       6553
002540  012746  000001                  MOV     #1,-(SP)
002544  010600                          MOV     SP,R0                   ; SP,*
002546  104414                          TRAP    14
002550  005726                          TST     (SP)+
002552  013700  000000G                 45$:    MOV     RP.ADDR,R0              ;                                       6554
002556  032760  000100  000022         BIT     #100,22(R0)
002564  001407                          BEQ     46$
002566  012716  000000G                 MOV     #F.5,(SP)               ;                                       6555
002572  012746  000001                  MOV     #1,-(SP)
002576  010600                          MOV     SP,R0                   ; SP,*
002600  104414                          TRAP    14
002602  005726                          TST     (SP)+
002604  013700  000000G                 46$:    MOV     RP.ADDR,R0              ;                                       6556
002610  032760  000040  000022         BIT     #40,22(R0)
002616  001407                          BEQ     47$
```

```
002620  012716  000000G                        MOV     #F.6,(SP)               ;                                       6555
002624  012746  000001                         MOV     #1,-(SP)
002630  010600                                 MOV     SP,R0                   ; SP,*
002632  104414                                 TRAP    14
002634  005726                                 TST     (SP)+
002636  013700  000000G              47$:      MOV     RP.ADDR,R0              ;                                       6558
002642  032760  000020  000022                BIT     #20,22(R0)
002650  001407                                 BEQ     48$
002652  012716  000000G                        MOV     #F.7,(SP)               ;                                       6559
002656  012746  000001                         MOV     #1,-(SP)
002662  010600                                 MOV     SP,R0                   ; SP,*
002664  104414                                 TRAP    14
002666  005726                                 TST     (SP)+
002670  013700  000000G              48$:      MOV     RP.ADDR,R0              ;                                       6560
002674  016000  000022                         MOV     22(R0),R0
002700  042700  077777                         BIC     #77777,R0
002704  020027  100000                         CMP     R0,#-100000
002710  001007                                 BNE     49$
002712  012716  000000G                        MOV     #F.8,(SP)               ;                                       6561
002716  012746  000001                         MOV     #1,-(SP)
002722  010600                                 MOV     SP,R0                   ; SP,*
002724  104414                                 TRAP    14
002726  005726                                 TST     (SP)+
002730  013700  000000G              49$:      MOV     RP.ADDR,R0              ;                                       6562
002734  032760  000002  000022                BIT     #2,22(R0)
002742  001407                                 BEQ     50$
002744  012716  000000G                        MOV     #F.9,(SP)               ;                                       6563
002750  012746  000001                         MOV     #1,-(SP)
002754  010600                                 MOV     SP,R0                   ; SP,*
002756  104414                                 TRAP    14
002760  005726                                 TST     (SP)+
002762  013700  000000G              50$:      MOV     RP.ADDR,R0              ;                                       6564
002766  032760  000001  000022                BIT     #1,22(R0)
002774  001407                                 BEQ     51$
002776  012716  000000G                        MOV     #F.10,(SP)              ;                                       6565
003002  012746  000001                         MOV     #1,-(SP)
003006  010600                                 MOV     SP,R0                   ; SP,*
003010  104414                                 TRAP    14
003012  005726                                 TST     (SP)+
003014  005726                       51$:      TST     (SP)+                   ;                                       6566
003016  013700  000000G              52$:      MOV     RP.ADDR,R0              ;                                       6567
003022  126027  000014  000211                CMPB    14(R0),#211
003030  001402                                 BEQ     53$
003032  000137  013674'                        JMP     65$
003036  012716  000000G              53$:      MOV     #XX40,(SP)              ;                                       6570
003042  012746  000001                         MOV     #1,-(SP)
003046  010600                                 MOV     SP,R0                   ; SP,*
003050  104414                                 TRAP    14
003052  013700  000000G                        MOV     RP.ADDR,R0              ;                                       6571
003056  032760  000001  000022                BIT     #1,22(R0)
003064  001407                                 BEQ     54$
003066  012716  000000G                        MOV     #F.11,(SP)              ;                                       6572
003072  012746  000001                         MOV     #1,-(SP)
003076  010600                                 MOV     SP,R0                   ; SP,*
003100  104414                                 TRAP    14
003102  005726                                 TST     (SP)+
```

```
003104  013700  000000G                 54$:    MOV     RP.ADDR,R0                      ;                                       6573
003110  032760  000002  000022                  BIT     #2,22(R0)
003116  001407                                  BEQ     55$
003120  012716  000000G                         MOV     #F.12,(SP)                      ;                                       6574
003124  012746  000001                         MOV     #1, (SP)
003130  010600                                  MOV     SP,R0                           ; SP,*
003132  104414                                  TRAP    14
003134  005726                                  TST     (SP)+
003136  013700  000000G                 55$:    MOV     RP.ADDR,R0                      ;                                       6575
003142  016000  000022                         MOV     22(R0),R0
003146  042700  077777                         BIC     #77777,R0
003152  020027  100000                         CMP     R0,#-100000
003156  001007                                  BNE     56$
003160  012716  000000G                         MOV     #F.13,(SP)                      ;                                       6576
003164  012746  000001                         MOV     #1, (SP)
003170  010600                                  MOV     SP,R0                           ; SP,*
003172  104414                                  TRAP    14
003174  005726                                  TST     (SP)+
003176  013700  000000G                 56$:    MOV     RP.ADDR,R0                      ;                                       6577
003202  032760  040000  000022                  BIT     #40000,22(R0)
003210  001407                                  BEQ     57$
003212  012716  000000G                         MOV     #F.14,(SP)                      ;                                       6578
003216  012746  000001                         MOV     #1,-(SP)
003222  010600                                  MOV     SP,R0                           ; SP,*
003224  104414                                  TRAP    14
003226  005726                                  TST     (SP)+
003230  013700  000000G                 57$:    MOV     RP.ADDR,R0                      ;                                       6579
003234  105760  000022                         TSTB    22(R0)
003240  100007                                  BPL     58$
003242  012716  000000G                         MOV     #F.15,(SP)                      ;                                       6580
003246  012746  000001                         MOV     #1,-(SP)
003252  010600                                  MOV     SP,R0                           ; SP,*
003254  104414                                  TRAP    14
003256  005726                                  TST     (SP)+
003260  013700  000000G                 58$:    MOV     RP.ADDR,R0                      ;                                       6581
003264  032760  004000  000022                  BIT     #4000,22(R0)
003272  001407                                  BEQ     59$
003274  012716  000000G                         MOV     #F.16,(SP)                      ;                                       6582
003300  012746  000001                         MOV     #1,-(SP)
003304  010600                                  MOV     SP,R0                           ; SP,*
003306  104414                                  TRAP    14
003310  005726                                  TST     (SP)+
003312  013700  000000G                 59$.    MOV     RP.ADDR,R0                      ;                                       6583
003316  032760  002000  000022                  BIT     #2000,22(R0)
003324  001407                                  BEQ     60$
003326  012716  000000G                         MOV     #F.17,(SP)                      ;                                       6584
003332  012746  000001                         MOV     #1,-(SP)
003336  010600                                  MOV     SP,R0                           ; SP,*
003340  104414                                  TRAP    14
003342  005726                                  TST     (SP)+
003344  013700  000000G                 60$:    MOV     RP.ADDR,R0                      ;                                       6585
003350  032760  000100  000022                  BIT     #100,22(R0)
003356  001407                                  BEQ     61$
003360  012716  000000G                         MOV     #F.18,(SP)                      ;                                       6586
003364  012746  000001                         MOV     #1,-(SP)
003370  010600                                  MOV     SP,R0                           ; SP,*
```

```
003372  104414                          TRAP    14
003374  005726                          TST     (SP)+
003376  013700  000000G         61$:    MOV     RP.ADDR,R0              ;                       6587
003402  032760  020000  000022          BIT     #20000,22(R0)
003410  001407                          BEQ     62$
003412  012716  000000G                 MOV     #F.19,(SP)              ;                       6588
003416  012746  000001                  MOV     #1,-(SP)
003422  010600                          MOV     SP,R0                   ; SP,*
003424  104414                          TRAP    14
003426  005726                          TST     (SP)+
003430  013700  000000G         62$:    MOV     RP.ADDR,R0              ;                       6589
003434  032760  010000  000022          BIT     #10000,22(R0)
003442  001407                          BEQ     63$
003444  012716  000000G                 MOV     #F.20,(SP)              ;                       6590
003450  012746  000001                  MOV     #1,-(SP)
003454  010600                          MOV     SP,R0                   ; SP,*
003456  104414                          TRAP    14
003460  005726                          TST     (SP)+
003462  013700  000000G         63$:    MOV     RP.ADDR,R0              ;                       6591
003466  032760  000004  000022          BIT     #4,22(R0)
003474  001407                          BEQ     64$
003476  012716  000000G                 MOV     #F.21,(SP)              ;                       6592
003502  012746  000001                  MOV     #1,-(SP)
003506  010600                          MOV     SP,R0                   ; SP,*
003510  104414                          TRAP    14
003512  005726                          TST     (SP)+
003514  005726                  64$:    TST     (SP)+                   ;                       6569
003516  062706  000006          65$:    ADD     #6,SP                   ;                       6499
003522  005703                  66$:    TST     R3                      ;                       6443
003524  001002                          BNE     67$
003526  000137  015074'                 JMP     85$
003532  013700  000000G         67$:    MOV     RP.ADDR,R0              ;                       6597
003536  005016                          CLR     (SP)
003540  116016  000003                  MOVB    3(R0),(SP)
003544  012746  000000G                 MOV     #XX38,-(SP)
003550  012746  000002                  MOV     #2,-(SP)
003554  010600                          MOV     SP,R0                   ; SP,*
003556  104414                          TRAP    14
003560  013700  000000G                 MOV     RP.ADDR,R0              ;                       6598
003564  005002                          CLR     R2
003566  156002  000014                  BISB    14(R0),R2
003572  012701  177777                  MOV     #-1,R1
003576  020227  000204                  CMP     R2,#204
003602  001010                          BNE     68$
003604  005001                          CLR     R1
003606  012716  000000G                 MOV     #EX.SCC,(SP)            ;                       6600
003612  012746  000001                  MOV     #1,-(SP)
003616  010600                          MOV     SP,R0                   ; SP,*
003620  104414                          TRAP    14
003622  005726                          TST     (SP)+
003624  020227  000211          68$:    CMP     R2,#211                 ;                       6598
003630  001010                          BNE     69$
003632  005001                          CLR     R1
003634  012716  000000G                 MOV     #EX.ONL,(SP)            ;                       6601
003640  012746  000001                  MOV     #1,-(SP)
003644  010600                          MOV     SP,R0                   ; SP,*
```

```
003646  104414                          TRAP    14
003650  005726                          TST     (SP)+
003652  020227  000220          69$:    CMP     R2,#220                 ;                       6599
003656  001010                          BNE     70$
003660  005001                          CLR     R1
003662  012716  000000G                 MOV     #EX.ACC,(SP)            ;                       6602
003666  012746  000001                  MOV     #1, (SP)
003672  010600                          MOV     SP,R0                   ; SP,*
003674  104414                          TRAP    14
003676  005726                          TST     (SP)+
003700  020227  000241          70$:    CMP     R2,#241                 ;                       6598
003704  001010                          BNE     71$
003706  005001                          CLR     R1
003710  012716  000000G                 MOV     #EX.RD,(SP)             ;                       6603
003714  012746  000001                  MOV     #1,-(SP)
003720  010600                          MOV     SP,R0                   ; SP,*
003722  104414                          TRAP    14
003724  005726                          TST     (SP)+
003726  020227  000242          71$:    CMP     R2,#242                 ;                       6598
003732  001010                          BNE     72$
003734  005001                          CLR     R1
003736  012716  000000G                 MOV     #EX.WRT,(SP)            ;                       6604
003742  012746  000001                  MOV     #1,-(SP)
003746  010600                          MOV     SP,R0                   ; SP,*
003750  104414                          TRAP    14
003752  005726                          TST     (SP)+
003754  020227  000211          72$:    CMP     R2,#211                 ;                       6598
003760  001010                          BNE     73$
003762  005001                          CLR     R1
003764  012716  000000G                 MOV     #EX.ONL,(SP)            ;                       6605
003770  012746  000001                  MOV     #1,-(SP)
003774  010600                          MOV     SP,R0                   ; SP,*
003776  104414                          TRAP    14
004000  005726                          TST     (SP)+
004002  020227  000220          73$:    CMP     R2,#220                 ;                       6598
004006  001010                          BNE     74$
004010  005001                          CLR     R1
004012  012716  000000G                 MOV     #EX.ACC,(SP)            ;                       6606
004016  012746  000001                  MOV     #1,-(SP)
004022  010600                          MOV     SP,R0                   ; SP,*
004024  104414                          TRAP    14
004026  005726                          TST     (SP)+
004030  020227  000241          74$:    CMP     R2,#241                 ;                       6598
004034  001010                          BNE     75$
004036  005001                          CLR     R1
004040  012716  000000G                 MOV     #EX.RD,(SP)             ;                       6607
004044  012746  000001                  MOV     #1,-(SP)
004050  010600                          MOV     SP,R0                   ; SP,*
004052  104414                          TRAP    14
004054  005726                          TST     (SP)+
004056  020227  000242          75$:    CMP     R2,#242                 ;                       6598
004062  001010                          BNE     76$
004064  005001                          CLR     R1
004066  012716  000000G                 MOV     #EX.WRT,(SP)            ;                       6608
004072  012746  000001                  MOV     #1,-(SP)
004076  010600                          MOV     SP,R0                   ; SP,*
```

```
004100  104414                          TRAP    14
004102  005726                          TST     (SP)+
004104  005701                  76$:    TST     R1                      ;                       6593
004106  001414                          BEQ     77$
004110  013700  000000G                 MOV     RP.ADDR,R0              ;                       6609
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              ;                       6611
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)              ;                       6612
004176  012746  000001                  MOV     #1,-(SP)
004202  010600                          MOV     SP,R0                   ; SP,*
004204  104414                          TRAP    14
004206  013700  000000G                 MOV     RP.ADDR,R0              ;                       6613
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)              ;                       6614
004242  012746  000001                  MOV     #1,-(SP)
004246  010600                          MOV     SP,R0                   ; SP,*
004250  104414                          TRAP    14
004252  013700  000000G                 MOV     RP.ADDR,R0              ;                       6615
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              ;                       6623
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                 ;                       6624
004344  001403                          BEQ     79$
004346  020027  000242                  CMP     R0,#242                 ;                       6625
```

```
004352  001072                          BNE     82$
004354  013700  000000G         79$:    MOV     RP.ADDR,R0              ;                               6628
004360  016016  000044                  MOV     44(R0),(SP)
004364  012746  000000G                 MOV     #XX24,-(SP)
004370  012746  000002                  MOV     #2,-(SP)
004374  010600                          MOV     SP,R0                   ; SP,*
004376  104415                          TRAP    15
004400  013700  000000G                 MOV     RP.ADDR,R0              ;                               6629
004404  016016  000020                  MOV     20(R0),(SP)
004410  012746  000000G                 MOV     #XX25,-(SP)
004414  012746  000002                  MOV     #2,-(SP)
004420  010600                          MOV     SP,R0                   ; SP,*
004422  104415                          TRAP    15
004424  013700  000000G                 MOV     RP.ADDR,R0              ;                               6630
004430  016016  000024                  MOV     24(R0),(SP)
004434  016046  000026                  MOV     26(R0),-(SP)
004440  012746  000000G                 MOV     #XX26,-(SP)
004444  012746  000003                  MOV     #3,-(SP)
004450  010600                          MOV     SP,R0                   ; SP,*
004452  104415                          TRAP    15
004454  013700  000000G                 MOV     RP.ADDR,R0              ;                               6631
004460  005760  000014                  TST     14(R0)
004464  100011                          BPL     80$
004466  016016  000040                  MOV     40(R0),(SP)             ;                               6633
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$                     ;                               6631
004510  016016  000050         80$:    MOV     50(R0),(SP)             ;                               6635
004514  011646                          MOV     (SP),-(SP)
004516  012746  000000G                 MOV     #XX22,-(SP)
004522  012746  000003                  MOV     #3,-(SP)
004526  010600                          MOV     SP,R0                   ; SP,*
004530  104415                          TRAP    15
004532  005726                          TST     (SP)+
004534  062706  000022          81$:    ADD     #22,SP                  ;                               6627
004540  013700  000000G         82$:    MOV     RP.ADDR,R0              ;                               6637
004544  126027  000014  000204         CMPB    14(R0),#204
004552  001404                          BEQ     83$
004554  126027  000014  000205         CMPB    14(R0),#205             ;                               6638
004562  001053                          BNE     84$
004564  013700  000000G         83$:    MOV     RP.ADDR,R0              ;                               6641
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              ;                               6642
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                ;                               6643
```

```
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              ;                       6644
004706  062706  000020                  ADD     #20,SP                  ;                       6640
004712  062706  000020          84$:    ADD     #20,SP                  ;                       6596
004716  062706  000016          85$:    ADD     #16,SP                  ;                       6420
004722  000207                          RTS     PC                      ;                       6409

; Routine Size:  1258 words,    Routine Base:  $CODE$ + 10156
; Maximum stack depth per invocation:  32 words
```

```
;        6656  !+
;        6657  GLOBAL ROUTINE EMS_DBN : NOVALUE
;        6658  !+
;        6659  !       THIS ROUTINE PRINTS THE PRESENT DBN
;        6660  !
;        6661  !       IMPLICIT INPUTS:
;        6662  !               CST_ADDR   ADDRESS OF CONTROLLER STATUS TABLE
;        6663  !
;        6664  BEGIN
;        6665  PRINTB (XX13, .CDISK);                                  ! "DISK XXX"
;        6666  PRINTB (XX23, .CST_ADDR [.CUOFF + 3, D_DBN], .CST_ADDR [.CUOFF + 3, D_DBN]);          ! "DBN: XXXXXX."
;        6667  PRINTB (XX32, .S_DUPPKT - 2);                          ! PRINTS THE BYTE COUNT
;        6668  PRINTB (XX33, .S_PATTERN);                             ! PRINTS THE PATTERN WRITTEN
;        6669  PRINTB (XX34, .(DUPPKT + .S_DUPPKT), .(DUPPKT + .S_DUPPKT));    ! PRINTS THE WORD READ
;        6670  EMS_BLK (DUPPKT +2, 256);                             ! PRINTS THE WHOLE BLOCK READ IN OCTAL
;        6671  END;                                                  ! ROUTINE EMS DBN


                                        .SBTTL  EMS.DBN ERROR MESSAGE SUBROUTINES
000000  013746  000000G         EMS.DBN::
                                MOV     CDISK,-(SP)                     ;                               6665
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                ;                               6666
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)                   ;                                       6664
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)                  ;                                       6668
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                     ;                                       6669
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)                  ;                                       6670
000160  012746  000400                  MOV     #400,-(SP)
000164  004737  000000V                 JSR     PC,EMS.BLK
000170  062706  000034                  ADD     #34,SP                          ;                                       6664
000174  000207                          RTS     PC                              ;                                       6657

; Routine Size:  63 words,      Routine Base:  $CODE$ + 15102
; Maximum stack depth per invocation:  15 words
```

```
;         6672  ROUTINE EMS_DUP : NOVALUE =
;         6673  !+
;         6674  !       THIS ROUTINE PRINTS OUT THE LISTING OF THE DUP PACKETS ONLY ASSOCIATED WITH DUP RECEIVED DATA COMMAND
;         6675  !       AND TE SEND DATA COMMAND.
;         6676  !
;         6677  !-
;         6678  begin
;         6679  OWN
;         6680          MESSAGETYPE : VECTOR [6] INITIAL (T_QUE, T_DEF, T_INF, T_TER, T_FAT, T_SPL),
;         6681          MSGNUMBERS  : VECTOR [9] INITIAL (M_TER, M_BIN, M_COD, M_DAT, M_UR, M_URP, M_UP, M_UL, M_ASC),
;         6682          RCD_ERRORS  : VECTOR [5] INITIAL (ebh_30, E_UNT, E_BLK, E_DEV, E_ZER);
;         6683                  ! TABLE OF BASIC, HARD ERROR MESSAGE ADDRESSES, INDEXED BY STATUS CODE
;         6684  !PRINTB (XX13, .CDISK);                                  ! "DISK XXX"
;         6685  PRINTB (XX29);
;         6686  if .duppkt [duptype] lss 6
;         6687  then
;         6688      PRINTB (.MESSAGETYPE [.DUPPKT [DUPTYPE]   1])
;         6689  else
;         6690      printb (ex_op,  duppkt [duptype]);
;         6691
;         6692  PRINTB (XX30);
;         6693  if .duppkt [dupmesg] lss 9
;         6694  then
;         6695      begin
;         6696      PRINTB (.MSGNUMBERS [.DUPPKT [DUPMESG]   1]);
;         6697      IF .DUPPKT [DUPMESG] EQL 3
;         6698      THEN
;         6699          BEGIN
;         6700          PRINTB (XX31);
;         6701          PRINTB (.RCD_ERRORS [.DUPPKT [DUPBF1]]);
;         6702          END;
;         6703      end
;         6704  else printb (ex_op, .duppkt [dupmesg]);
;         6705  end;

015300  000000G                         MESSAGETYPE:
                                                .WORD   T.QUE
015302  000000G                                 .WORD   T.DEF
015304  000000G                                 .WORD   T.INF
015306  000000G                                 .WORD   T.TER
015310  000000G                                 .WORD   T.FAT
015312  000000G                                 .WORD   T.SPL
015314  000000G                         MSGNUMBERS:
                                                .WORD   M.TER
015316  000000G                                 .WORD   M.BIN
015320  000000G                                 .WORD   M.COD
015322  000000G                                 .WORD   M.DAT
015324  000000G                                 .WORD   M.UR
015326  000000G                                 .WORD   M.URP
015330  000000G                                 .WORD   M.UP
015332  000000G                                 .WORD   M.UL
015334  000000G                                 .WORD   M.ASC
015336  000000G                         RCD.ERRORS:
                                                .WORD   EBH.30
015340  000000G                                 .WORD   E.UNT
```

```
015342  000000G                         .WORD   E.BLK
015344  000000G                         .WORD   E.DEV
015346  000000G                         .WORD   E.ZER


                                        .SBTTL  EMS.DUP ERROR MESSAGE SUBROUTINES
000000  012746  000000G       EMS.DUP:  MOV     #XX29,-(SP)                     ;                       6685
000004  012746  000001                  MOV     #1, (SP)
000010  010600                          MOV     SP,R0                           ; SP,*
000012  104414                          TRAP    14
000014  013700  000000G                 MOV     DUPPKT,R0                       ;                       6686
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                              ;                       6688
000046  016016  015276                  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$                              ;                       6686
000064  010016              1$:         MOV     R0,(SP)                         ;                       6690
000066  012746  000000G                 MOV     #EX.OP,-(SP)
000072  012746  000002                  MOV     #2,-(SP)
000076  010600                          MOV     SP,R0                           ; SP,*
000100  104414                          TRAP    14
000102  005726                          TST     (SP)+
000104  012716  000000G       2$:       MOV     #XX30,(SP)                      ;                       6692
000110  012746  000001                  MOV     #1, (SP)
000114  010600                          MOV     SP,R0                           ; SP,*
000116  104414                          TRAP    14
000120  013700  000000G                 MOV     DUPPKT,R0                       ;                       6693
000124  042700  170000                  BIC     #170000,R0
000130  020027  000011                  CMP     R0,#11
000134  002037                          BGE     3$
000136  006300                          ASL     R0                              ;                       6694
000140  016016  015312'                 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                       ;                       6697
000160  042700  170000                  BIC     #170000,R0
000164  020027  000003                  CMP     R0,#3
000170  001031                          BNE     4$
000172  012716  000000G                 MOV     #XX31,(SP)                      ;                       6700
000176  012746  000001                  MOV     #1,-(SP)
000202  010600                          MOV     SP,R0                           ; SP,*
000204  104414                          TRAP    14
000206  013700  000002G                 MOV     DUPPKT+2,R0                     ;                       6701
000212  006300                          ASL     R0
000214  016016  015336'                 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)+             ;                       6699
000232  000410                          BR      4$                      ;                       6693
000234  010016                  3$:     MOV     R0,(SP)                 ;                       6704
000236  012746  000000G                 MOV     #EX.OP,-(SP)
000242  012746  000002                  MOV     #2,(SP)
000246  010600                          MOV     SP,R0                   ; SP,*
000250  104414                          TRAP    14
000252  005726                          TST     (SP)+
000254  062706  000012          4$:     ADD     #12,SP                  ;                       6677
000260  000207                          RTS     PC                      ;                       6672

; Routine Size:  89 words,      Routine Base:  $CODE$ + 15350
; Maximum stack depth per invocation:  9 words
```

```
;	global routine EMS_BLK (ADDR, LENGTH) : novalue =
;
;	!+
;	!	THIS ROUTINE WILL PRINTX A BLOCK OF MEMORY WHICH IS "LENGTH" WORDS
;	!	LONG STARTING AT ADDRESS "ADDR". PRINTING IS DONE IN OCTAL, 8 WORDS
;	!	TO A LINE.
;	!-
;
;	    begin
;
;	    literal
;		MASK = %o'7' ;
;	PRINTX (CRLF);
;	    incr COUNT from 1 to .LENGTH do		! FOR EACH WORD TO PRINT
;		begin
;
;		if ((.COUNT - 1) and MASK) eql 0	! IF AT START OF A NEW LINE
;		then
;		    PRINTX (SPACE4);			! PRINTX 4 SPACES
;
;		PRINTX (EX_WRD, ..ADDR);		! PRINTX A WORD
;		ADDR = .ADDR + 2;			! ADVANCE TO NEXT ADDRESS
;
;		if (((.COUNT and MASK) eql 0) or	! IF AT THE END OF A LINE OR
;		    (.COUNT eql .LENGTH))		! WHEN DONE
;		then
;		    PRINTX (CRLF);			! PRINTX <CR><LF>
;
;		end;
;
;	    end;


					.SBTTL	EMS.BLK ERROR MESSAGE SUBROUTINES
000000	010146			EMS.BLK::
						MOV	R1,-(SP)		;		6706
000002	012746	000000G			MOV	#CRLF,-(SP)		;		6718
000006	012746	000001			MOV	#1,-(SP)
000012	010600				MOV	SP,R0			; SP,*
000014	104415				TRAP	15
000016	005001				CLR	R1			; COUNT		6719
000020	000445				BR	5$
000022	010100			1$:	MOV	R1,R0			; COUNT,*	6722
000024	005300				DEC	R0
000026	032700	000007			BIT	#7,R0
000032	001007				BNE	2$
000034	012716	000000G			MOV	#SPACE4,(SP)		;		6724
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,*	6726
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                        6727
000100  032701  000007                          BIT     #7,R1                   ; *,COUNT                       6729
000104  001403                                  BEQ     3$
000106  020166  000014                          CMP     R1,14(SP)               ; COUNT,LENGTH                  6730
000112  001007                                  BNE     4$
000114  012716  000000G                 3$:     MOV     #CRLF,(SP)              ;                               6732
000120  012746  000001                          MOV     #1, -(SP)
000124  010600                                  MOV     SP,R0                   ; SP,*
000126  104415                                  TRAP    15
000130  005726                                  TST     (SP)+
000132  022626                          4$:     CMP     (SP)+,(SP)+             ;                               6720
000134  005201                          5$:     INC     R1                      ; COUNT                         6719
000136  020166  000010                          CMP     R1,10(SP)               ; COUNT,LENGTH
000142  003727                                  BLE     1$
000144  022626                                  CMP     (SP)+,(SP)+             ;                               6714
000146  012601                                  MOV     (SP)+,R1                ;                               6706
000150  000207                                  RTS     PC

; Routine Size:  53 words,      Routine Base:  $CODE$ + 15632
; Maximum stack depth per invocation:  8 words
```

```
;       6737  !ROUTINE EMS_MAL : NOVALUE =
;       6738  !*
;       6739  !       THIS ROUTINE PRINTS ALL TABLES IN AN EFFORT TO FIND THE MISSING MESSAGE OR THE PROBLEM
;       6740  !-
;       6741  !begin
;       6742
;       6743  !PRINTB (EB_DCT,dct);                          ! PRINT   DCT TABLE CONTENTS'
;       6744  !EMS_BLK (DCT, DCT_LEN);                              ! PRINT BLOCK OF WORDS
;       6745
;       6746  !PRINTB (EB_COMM, .DCT [0, RR_BEG]);           ! PRINT "COMMAND RING" AND STARTING ADDR
;       6747  !EMS_BLK (.DCT [0, RR_BEG] - 4, COMM_LEN);            ! PRINT BLOCK OF COMMAND RING
;       6748
;       6749  !PRINTB (EBNEX1, .DCT [0, RR_POLL]);                  !PRINT ADDR OF COMMAND OF NEXT RR POLL
;       6750
;       6751  !PRINTB (EB_NEX2, ..DCT [0, RR_POLL]   8);                     !PRINT ADDR OF PACKET TO BE POLLED
;       6752
;       6753  !PRINTB (EBNEX3, .DCT [0, CR_POLL] - 8);                      !PRINT ADDR OF PACKET TO BE POLLED
;       6754  !PRINTB (EB_PKT);                             ! PRINTS "PACKETS IN MEMORY"
;       6755  !incr COUNT from 0 to PKT_CNT - 1 do           ! FOR EACH MSCP ENVELOPE
;       6756  !       begin
;       6757  !       EMS_BLK ((MSCP_PKT [.COUNT, 0,0,16,0]), PKT_LEN);       !PRINTS CONTENTS OF PACKETS
;       6758  !       end;
;       6759  !end;
;       6760

;       6761  !ROUTINE EMS_RAL : NOVALUE =
;       6762  !*
;       6763  !       THIS ROUTINE PRINTS ALL return packets IN AN EFFORT TO FIND THE MISSING packet OR THE PROBLEM
;       6764  !-
;       6765  !begin
;       6766  !PRINTB (EB_RAL);
;       6767  !incr COUNT from 0 to RP_CNT - 1 do
;       6768  !       begin
;       6769  !       EMS_BLK ((RETPKT + .COUNT * RP_LEN) , RP_LEN);                  ! PRINT BLOCK OF WORDS
;       6770  !       end;
;       6771
;       6772  !end;
;       6773
```

```
;        6774  routine EMS_LBN : novalue =
;        6775
;        6776  !+
;        6777  !       THIS ROUTINE PRINTS (EXTENDED) ONE OF TWO BLOCK NUMBERS APPEARING IN
;        6778  !       THE CURRENT RETURN PACKET. NORMALLY, THE LBN FIELD IS PRINTED; THIS
;        6779  !       FIELD WAS COPIED INTO THE RETURN PACKET FROM THE ASSOCIATED COMMAND
;        6780  !       PACKET. HOWEVER, IF THE "FLAGS" FIELD OF THE CURRENT RETURN PACKET
;        6781  !       INDICATES "BAD BLOCK REPORTED", THEN THE "FIRST BAD BLOCK" FIELD IS
;        6782  !       PRINTED.
;        6783  !
;        6784  !       IMPLICIT INPUTS:
;        6785  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;        6786  !
;        6787
;        6788      if BIT_TST (RP_ADDR [FLAGS], EF_BBR)                        ! IF BAD BLOCK REPORTED
;        6789      then
;        6790          PRINTX (XX21, .RP_ADDR [BBLK_LO], .RP_ADDR [BBLK_LO])   ! "BAD BLOCK REPORTED: XXXXXX."
;        6791      else
;        6792          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                      ;                       6788
000004  005760  000014                  TST     14(R0)
000010  100012                          BPL     1$
000012  016046  000040                  MOV     40(R0),-(SP)                    ;                       6790
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$                      ;               6788
000036  016046  000050          1$:     MOV     50(R0),-(SP)            ;               6792
000042  011646                          MOV     (SP),-(SP)
000044  012746  000000G                 MOV     #XX22,-(SP)
000050  012746  000003                  MOV     #3,-(SP)
000054  010600                          MOV     SP,R0                   ; SP,*
000056  104415                          TRAP    15
000060  062706  000010          2$:     ADD     #10,SP                  ;               6788
000064  000207                          RTS     PC                      ;               6774

; Routine Size:  27 words,      Routine Base:  $CODE$ + 16004
; Maximum stack depth per invocation:  6 words
```

```
global routine EMS_EL (index) : novalue =

!+
!       THIS ROUTINE IS CALLED FROM 'SEQUEN' AND 'DATAGM  AND PRINTS THE CONTENTS OF THE
!       ERROR-LOG PACKET
!

    begin

    local
        ELOG_ADDR : ref block [EP_LEN, word] field (EP_FIELDS),
        REASON : word,
        DISK_NUM : byte,
        ELOG_CODE : byte,
        ELOG_SUB : word;

    ELOG_ADDR = ELOG_PKT + (.index * EP_LEN * 2);                  ! ERROR LOG PACKET'S ADDRESS
    REASON = .ELOG_ADDR [EL_FORMAT];                               ! FORMAT
    DISK_NUM = .ELOG_ADDR [EL_DK_NUM];                             ! DISK NUMBER
    ELOG_CODE = .ELOG_ADDR [EL_CODE];                              ! CODE
    ELOG_SUB = .ELOG_ADDR [EL_SUBCODE];                            ! SUBCODE
    PRINTB (ELG_00);                                               ! ERROR-LOG MESSAGE RECEIVED

    if (.REASON eql FORMAT_CNTR) or
       (.REASON eql FORMAT_HOST)
    then
        PRINTB (.ELG_FMT [.REASON])                                ! PRINT BASIC REASON
    else
        PRINTB (.ELG_FMT [.REASON], .DISK_NUM);                    ! PRINT BASIC REASON WITH DISK NUMBER

    if (.ELOG_CODE gtru 0) and
       (.ELOG_CODE lequ 11)
    then
        begin
        PRINTX (ASTERISK);
        PRINTX (.ERR_COD [.ELOG_CODE - 1]);                        ! CODE
        end
    else

        if .ELOG_CODE eql ST_DIA                                   ! MESSAGE FROM INTERNAL DIAGNOSTICS
        then
            begin
            PRINTX (ASTERISK);
            PRINTX (.ERR_COD [12]);
            end;

    if (.ELOG_CODE eql ST_MFE) and
       (.ELOG_SUB lequ 10)
    then
        begin
        PRINTX (CRLF);
        PRINTX (ASTERISK);
        PRINTX (.TBL_MFE [.ELOG_SUB]);                             ! MEDIA FORMAT ERROR
        end;

    if (.ELOG_CODE eql ST_DAT) and
```

```
;       6849        (.ELOG_SUB lequ 15)
;       6850    then
;       6851        begin
;       6852        PRINTX (CRLF);
;       6853        PRINTX (ASTERISK);
;       6854        PRINTX (.TBL_DAT [.ELOG_SUB]);                  ! DATA ERROR
;       6855        end;
;       6856
;       6857    if (.ELOG_CODE eql ST_HST) and
;       6858       (.ELOG_SUB lequ 4)
;       6859    then
;       6860        begin
;       6861        PRINTX (CRLF);
;       6862        PRINTX (ASTERISK);
;       6863        PRINTX (.TBL_HST [.ELOG_SUB]);                  ! HOST ACCESS ERROR
;       6864        end;
;       6865
;       6866    if (.ELOG_CODE eql ST_CNT) and
;       6867       (.ELOG_SUB lequ 3)
;       6868    then
;       6869        begin
;       6870        PRINTX (CRLF);
;       6871        PRINTX (ASTERISK);
;       6872        PRINTX (.TBL_CNT [.ELOG_SUB]);                  ! CONTROLLER ERROR
;       6873        end;
;       6874
;       6875    if (.ELOG_CODE eql ST_DRV) and
;       6876       (.ELOG_SUB lequ 8)
;       6877    then
;       6878        begin
;       6879        PRINTX (CRLF);
;       6880        PRINTX (ASTERISK);
;       6881        PRINTX (.TBL_DRV [.ELOG_SUB]);                  ! DRIVE ERROR
;       6882        end;
;       6883
;       6884    if .REASON eql FORMAT_XFER                          ! IF DISK XFER INVOLVED
;       6885    then
;       6886
;       6887        if .ELOG_ADDR [EL_BLOCK_TYPE] eql TYPE_LBN      ! PRINT LBN OR RBN
;       6888        then
;       6889            PRINTX (XX22, .ELOG_ADDR [EL_BLOCK], .ELOG_ADDR [EL_BLOCK])
;       6890        else
;       6891            PRINTX (EX_RBN, .ELOG_ADDR [EL_BLOCK], .ELOG_ADDR [EL_BLOCK]);
;       6892
;       6893    PRINTX (CRLF);
;       6894    EMS_BLK ((.ELOG_ADDR + 2), ((.ELOG_ADDR [EL_MSGLEN] + 1) / 2) + 2); ! PRINTX CONTENTS OF PACKET
;       6895    ELOG_ADDR [EL_CONTENTS] = EMPTY;                    ! DECLARE SAVE AREA FREE
;       6896
;       6897    end;


                                    .SBTTL  EMS.EL ERROR MESSAGE SUBROUTINES
000000  004137  000000G    EMS.EL::JSR     R1,$SAVE5                       ;                       6793
000004  005746                     TST     -(SP)
000006  016646  000020             MOV     20(SP), (SP)                    ; INDEX,*               6809
000012  012746  000102             MOV     #102,-(SP)
```

```
000016  004737  000000G                 JSR     PC,BL$MUL
000022  062700  000000G                 ADD     #ELOG.PKT,R0
000026  010003                         MOV     R0,R3           ; *,ELOG.ADDR
000030  005004                         CLR     R4              ; REASON                        6810
000032  156304  000016                 BISB    16(R3),R4       ; *(ELOG.ADDR),REASON
000036  116366  000012  000004         MOVB    12(R3),4(SP)    ; *(ELOG.ADDR),DISK.NUM         6811
000044  116300  000020                 MOVB    20(R3),R0       ; *(ELOG.ADDR),*                6812
000050  042700  177740                 BIC     #177740,R0
000054  105002                         CLRB    R2              ; ELOG.CODE
000056  050002                         BIS     R0,R2           ; *,ELOG.CODE
000060  016301  000020                 MOV     20(R3),R1       ; *(ELOG.ADDR),ELOG.SUB         6813
000064  006201                         ASR     R1              ; ELOG.SUB
000066  006201                         ASR     R1              ; ELOG.SUB
000070  006201                         ASR     R1              ; ELOG.SUB
000072  006201                         ASR     R1              ; ELOG.SUB
000074  006201                         ASR     R1              ; ELOG.SUB
000076  042701  174000                 BIC     #174000,R1      ; *,ELOG.SUB
000102  012716  000000G                MOV     #ELG.00,(SP)    ;                               6814
000106  012746  000001                 MOV     #1,-(SP)
000112  010600                         MOV     SP,R0           ; SP,*
000114  104414                         TRAP    14
000116  010405                         MOV     R4,R5           ; REASON,*                      6819
000120  006305                         ASL     R5
000122  005704                         TST     R4              ; REASON                        6816
000124  001403                         BEQ     1$
000126  020427  000001                 CMP     R4,#1           ; REASON,*                      6817
000132  001007                         BNE     2$
000134  016516  000000G      1$:        MOV     ELG.FMT(R5),(SP)  ;                             6819
000140  012746  000001                 MOV     #1,-(SP)
000144  010600                         MOV     SP,R0           ; SP,*
000146  104414                         TRAP    14
000150  000412                         BR      3$              ;                               6816
000152  005016              2$:        CLR     (SP)            ;                               6821
000154  116616  000006                 MOVB    6(SP),(SP)      ; DISK.NUM,*
000160  016546  000000G                MOV     ELG.FMT(R5),-(SP)
000164  012746  000002                 MOV     #2,-(SP)
000170  010600                         MOV     SP,R0           ; SP,*
000172  104414                         TRAP    14
000174  005726                         TST     (SP)+
000176  105702              3$:        TSTB    R2              ; ELOG.CODE                     6823
000200  001423                         BEQ     4$
000202  120227  000013                 CMPB    R2,#13          ; ELOG.CODE,*                   6824
000206  101020                         BHI     4$
000210  012716  000000G                MOV     #ASTERISK,(SP)  ;                               6827
000214  012746  000001                 MOV     #1,-(SP)
000220  010600                         MOV     SP,R0           ; SP,*
000222  104415                         TRAP    15
000224  005000                         CLR     R0              ;                               6828
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$              ;                               6826
000250  120227  000037      4$:        CMPB    R2,#37          ; ELOG.CODE,*                   6832
```

```
000254  001015                          BNE     6$
000256  012716  000000G                 MOV     #ASTERISK,(SP)          ;                               6835
000262  012746  000001                  MOV     #1,-(SP)
000266  010600                          MOV     SP,R0                   ; SP,*
000270  104415                          TRAP    15
000272  013716  000030G                 MOV     ERR.COD+30,(SP)         ;                               6836
000276  012746  000001                  MOV     #1, (SP)
000302  010600                          MOV     SP,R0                   ; SP,*
000304  104415                          TRAP    15
000306  022626                  5$:     CMP     (SP)+,(SP)+             ;                               6834
000310  120227  000005          6$:     CMPB    R2,#5                   ; ELOG.CODE,*                   6835
000314  001031                          BNE     7$
000316  020127  000012                  CMP     R1,#12                  ; ELOG.SUB,*                    6840
000322  101026                          BHI     7$
000324  012716  000000G                 MOV     #CRLF,(SP)              ;                               6843
000330  012746  000001                  MOV     #1,-(SP)
000334  010600                          MOV     SP,R0                   ; SP,*
000336  104415                          TRAP    15
000340  012716  000000G                 MOV     #ASTERISK,(SP)          ;                               6844
000344  012746  000001                  MOV     #1,-(SP)
000350  010600                          MOV     SP,R0                   ; SP,*
000352  104415                          TRAP    15
000354  010100                          MOV     R1,R0                   ; ELOG.SUB,*                    6845
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                   ;                               6842
000400  120227  000010          7$:     CMPB    R2,#10                  ; ELOG.CODE,*                   6848
000404  001031                          BNE     8$
000406  020127  000017                  CMP     R1,#17                  ; ELOG.SUB,*                    6849
000412  101026                          BHI     8$
000414  012716  000000G                 MOV     #CRLF,(SP)              ;                               6852
000420  012746  000001                  MOV     #1,-(SP)
000424  010600                          MOV     SP,R0                   ; SP,*
000426  104415                          TRAP    15
000430  012716  000000G                 MOV     #ASTERISK,(SP)          ;                               6853
000434  012746  000001                  MOV     #1,-(SP)
000440  010600                          MOV     SP,R0                   ; SP,*
000442  104415                          TRAP    15
000444  010100                          MOV     R1,R0                   ; ELOG.SUB,*                    6854
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                   ;                               6851
000470  120227  000011          8$:     CMPB    R2,#11                  ; ELOG.CODE,*                   6857
000474  001031                          BNE     9$
000476  020127  000004                  CMP     R1,#4                   ; ELOG.SUB,*                    6858
000502  101026                          BHI     9$
000504  012716  000000G                 MOV     #CRLF,(SP)              ;                               6861
000510  012746  000001                  MOV     #1,-(SP)
000514  010600                          MOV     SP,R0                   ; SP,*
000516  104415                          TRAP    15
```

```
000520  012716  000000G                        MOV     #ASTERISK,(SP)                  ;                               6862
000524  012746  000001                         MOV     #1,-(SP)
000530  010600                                 MOV     SP,R0                           ; SP,*
000532  104415                                 TRAP    15
000534  010100                                 MOV     R1,R0                           ; ELOG.SUB,*                    6863
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                           ;                               6860
000560  120227  000012            9$:          CMPB    R2,#12                          ; ELOG.CODE,*                   6866
000564  001031                                 BNE     10$
000566  020127  000003                         CMP     R1,#3                           ; ELOG.SUB,*                    6867
000572  101026                                 BHI     10$
000574  012716  000000G                        MOV     #CRLF,(SP)                      ;                               6870
000600  012746  000001                         MOV     #1,-(SP)
000604  010600                                 MOV     SP,R0                           ; SP,*
000606  104415                                 TRAP    15
000610  012716  000000G                        MOV     #ASTERISK,(SP)                  ;                               6871
000614  012746  000001                         MOV     #1,-(SP)
000620  010600                                 MOV     SP,R0                           ; SP,*
000622  104415                                 TRAP    15
000624  010100                                 MOV     R1,R0                           ; ELOG.SUB,*                    6872
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                           ;                               6869
000650  120227  000013            10$:         CMPB    R2,#13                          ; ELOG.CODE,*                   6875
000654  001030                                 BNE     11$
000656  020127  000010                         CMP     R1,#10                          ; ELOG.SUB,*                    6876
000662  101025                                 BHI     11$
000664  012716  000000G                        MOV     #CRLF,(SP)                      ;                               6879
000670  012746  000001                         MOV     #1,-(SP)
000674  010600                                 MOV     SP,R0                           ; SP,*
000676  104415                                 TRAP    15
000700  012716  000000G                        MOV     #ASTERISK,(SP)                  ;                               6880
000704  012746  000001                         MOV     #1,-(SP)
000710  010600                                 MOV     SP,R0                           ; SP,*
000712  104415                                 TRAP    15
000714  006301                                 ASL     R1                              ;                               6881
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                           ;                               6878
000736  020427  000002            11$:         CMP     R4,#2                           ; REASON,*                      6884
000742  001031                                 BNE     14$
000744  032763  170000  000060                 BIT     #170000,60(R3)                  ; *,*(ELOG.ADDR)                6887
000752  001012                                 BNE     12$
000754  016316  000056                         MOV     56(R3),(SP)                     ; *(ELOG.ADDR),*                6889
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$                     ;                               6887
001000  016316  000056          12$:    MOV     56(R3),(SP)             ; *(ELOG.ADDR),*                6891
001004  011646                          MOV     (SP), (SP)
001006  012746  000000G                 MOV     #EX.RBN, (SP)
001012  012746  000003                  MOV     #3, (SP)
001016  010600                          MOV     SP,R0                   ; SP,*
001020  104415                          TRAP    15
001022  062706  000006          13$:    ADD     #6,SP                   ;                               6887
001026  012716  000000G         14$:    MOV     #CRLF,(SP)              ;                               6893
001032  012746  000001                  MOV     #1,-(SP)
001036  010600                          MOV     SP,R0                   ; SP,*
001040  104415                          TRAP    15
001042  012716  000002                  MOV     #2,(SP)                 ;                               6894
001046  060316                          ADD     R3,(SP)                 ; ELOG.ADDR,*
001050  016346  000002                  MOV     2(R3),-(SP)             ; *(ELOG.ADDR),*
001054  005216                          INC     (SP)
001056  012746  000002                  MOV     #2,-(SP)
001062  004737  000000G                 JSR     PC,BL$DIV
001066  010066  000002                  MOV     R0,2(SP)
001072  062766  000002  000002          ADD     #2,2(SP)
001100  005726                          TST     (SP)+
001102  004737  015632'                 JSR     PC,EMS.BLK
001106  105063  000001                  CLRB    1(R3)                   ; *(ELOG.ADDR)                  6895
001112  062706  000016                  ADD     #16,SP                  ;                               6793
001116  000207                          RTS     PC

; Routine Size:  296 words,    Routine Base:  $CODE$ + 16072
; Maximum stack depth per invocation:  16 words
```

```
;        6898  global routine EMS_CMP (ADDR) : novalue
;        6899
;        6900  !+
;        6901  ! THIS ROUTINE IS CALLED FROM  HOST WRT CHK' AND PRINTS RELEVENT DATA ON A HOST
;        6902  ! COMPARE ERROR
;        6903  !-
;        6904
;        6905      begin
;        6906
;        6907      local
;        6908          ORIG_ADDR : ref block [RP_LEN, word] field (RP_FIELDS);
;        6909
;        6910      ORIG_ADDR = .ADDR;                                          ! ADDRESS OF THE WRITE RETPKT
;        6911      PRINTB (XX13, .CDISK);                                      ! "DISK XXX"
;        6912      PRINTB (DASH);                                              !
;        6913      PRINTB (.ERR_COD [12]);                                     ! " - HOST COMPARE ERROR"
;        6914      PRINTX (EX_LBW, .ORIG_ADDR [LBN_LO], .ORIG_ADDR [LBN_LO]);  ! LBN (WRITE)
;        6915      PRINTX (EX_LBR, .RP_ADDR [LBN_LO], .RP_ADDR [LBN_LO]);      ! LBN (READE)
;        6916      PRINTX (EX_CBW, .ORIG_ADDR [CBCNT_LO]);                     ! BYTE COUNT (WRITE)
;        6917      PRINTX (XX25, .ORIG_ADDR [BCNT_LO]);                        ! BYTE COUNT XMITTED (WRITE)
;        6918      PRINTX (EX_CBR, .RP_ADDR [CBCNT_LO]);                       ! BYTE COUNT (READ);
;        6919      PRINTX (XX25, .RP_ADDR [BCNT_LO]);                          ! BYTE COUNT XMITTED (READ)
;        6920      PRINTX (EX_BDW, .ORIG_ADDR [BUFF_1], .ORIG_ADDR [BUFF_0]);  ! BUFFER ADDRESS (WRITE)
;        6921      PRINTX (EX_BDR, .RP_ADDR [BUFF_1], .RP_ADDR [BUFF_0]);      ! BUFFER ADDRESS (READ)
;        6922      end;

                        .SBTTL  EMS.CMP ERROR MESSAGE SUBROUTINES
000000  010146          EMS.CMP::
                        MOV     R1,-(SP)                        ;                               6898
000002  016601  000004  MOV     4(SP),R1                        ; ADDR,ORIG.ADDR                6910
000006  013746  000000G MOV     CDISK,-(SP)                     ;                               6911
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)                      ;                               6912
000032  012746  000001  MOV     #1,-(SP)
000036  010600          MOV     SP,R0                           ; SP,*
000040  104414          TRAP    14
000042  013716  000030G MOV     ERR.COD+30,(SP)                 ;                               6913
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),*                6914
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                      ;                               6915
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),*                6916
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),*                6917
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              ;                               6918
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              ;                               6919
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),*                6920
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.BDR,R0               ;                               6921
000266  016C16  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                  ;                               6905
000316  012601                  MOV     (SP)+,R1                ;                               6898
000320  000207                  RTS     PC

; Routine Size:  105 words,    Routine Base:  $CODE$ + 17212
; Maximum stack depth per invocation:  28 words
```

```
;         6923  BGNMSG (EMS_01);


                                        .SBTTL  EMS.01 ERROR MESSAGE SUBROUTINES
000000  004737  000000V              EMS.01::JSR     PC,M$EMS.01                     ;                       6923
000004  104423                              TRAP    23
000006  000207                              RTS     PC

; Routine Size:  4 words,      Routine Base:  $CODE$ + 17534
; Maximum stack depth per invocation:  2 words


;         6924  PRINTB (EBS_01, MAX_UNITS);                 ! "MORE THAN XX UNITS SPECIFIED"
;         6925  ENDMSG;


                                        .SBTTL  M$EMS.01 ERROR MESSAGE SUBROUTINES
000000  012746  000004               M$EMS.01:
                                        MOV     #4,-(SP)                        ;                       6924
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                           ;                       6923
000024  000207                          RTS     PC

; Routine Size:  11 words,     Routine Base:  $CODE$ + 17544
; Maximum stack depth per invocation:  5 words


;         6926
;         6927  BGNMSG (EMS_10);


                                        .SBTTL  EMS.10 ERROR MESSAGE SUBROUTINES
000000  004737  000000V              EMS.10::JSR     PC,M$EMS.10                     ;                       6927
000004  104423                              TRAP    23
000006  000207                              RTS     PC

; Routine Size:  4 words,      Routine Base:  $CODE$ + 17572
; Maximum stack depth per invocation:  2 words


;         6928  PRINTB (EBD_10, .RDRX_ADDR + .OF_RC);          ! "NO RESPONSE AT ADDRESS XXXXXX"
;         6929  ENDMSG;


                                        .SBTTL  M$EMS.10 ERROR MESSAGE SUBROUTINES
000000  013746  000000G              M$EMS.10:
                                        MOV     RDRX.ADDR,-(SP)                 ;                       6928
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                           ;                       6927
```

```
000050  000207                                  RTS     PC

; Routine Size:  13 words,      Routine Base:  $CODE$ + 17602
; Maximum stack depth per invocation:  5 words


;         6930
;         6931  BGNMSG (EMS_12);


                                                .SBTTL  EMS.12 ERROR MESSAGE SUBROUTINES
000000  004737  000000V                 EMS.12::JSR     PC,M$EMS.12                     ;                                       6931
000004  104423                                  TRAP    23
000006  000207                                  RTS     PC

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


;         6932  PRINTB (EBD_12, .RDRX_ADDR);                       ! 'INCORRECT BR LEVEL GIVEN FOR DEVICE XXXXXX"
;         6933  ENDMSG;


                                                .SBTTL  M$EMS.12 ERROR MESSAGE SUBROUTINES
000000  013746  000000G               M$EMS.12:
                                                MOV     RDRX.ADDR,-(SP)                 ;                                       6932
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                           ;                                       6931
000024  000207                                  RTS     PC

; Routine Size:  11 words,      Routine Base:  $CODE$ + 17644
; Maximum stack depth per invocation:  5 words


;         6934
;         6935  BGNMSG (EMS_13);


                                                .SBTTL  EMS.13 ERROR MESSAGE SUBROUTINES
000000  004737  000000V                 EMS.13::JSR     PC,M$EMS.13                     .                                       6935
000004  104423                                  TRAP    23
000006  000207                                  RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 17672
; Maximum stack depth per invocation:  2 words


;         6936  PRINTB (EBD_13, .STEP);                           ! "STEP X READ ERROR"
;         6937  EMS_SA ();                                        ! PRINTX SA CONTENTS
;         6938  ENDMSG;


                                                .SBTTL  M$EMS.13 ERROR MESSAGE SUBROUTINES
```

```
000000  013746  000000G                    M$EMS.13:
                                                    MOV     STEP, (SP)                              ;                               6936
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  007122'                            JSR     PC,EMS.SA                               ;                               6937
000024  062706  000006                             ADD     #6,SP                                   ;                               6935
000030  000207                                     RTS     PC

; Routine Size:  13 words,      Routine Base:  $CODE$ + 17702
; Maximum stack depth per invocation:  5 words


;         6939
;         6940  BGNMSG (EMS_14);


                                                    .SBTTL  EMS.14 ERROR MESSAGE SUBROUTINES
000000  004737  000000V                 EMS.14::JSR     PC,M$EMS.14                             ;                               6940
000004  104423                                     TRAP    23
000006  000207                                     RTS     PC

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


;         6941  PRINTB (EBD_14, .IRDRX_ADDR);                   ! "BAD SA CODE FROM DEVICE XXXXXX"
;         6942  EMS_SA ();                                      ! PRINTX SA REGISTER CONTENTS
;         6943  ENDMSG;


                                                    .SBTTL  M$EMS.14 ERROR MESSAGE SUBROUTINES
000000  013746  000000G                    M$EMS.14:
                                                    MOV     IRDRX.ADDR,-(SP)                        ;                               6941
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  007122'                            JSR     PC,EMS.SA                               ;                               6942
000024  062706  000006                             ADD     #6,SP                                   ;                               6940
000030  000207                                     RTS     PC

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


;         6944
;         6945  BGNMSG (EMS_18);


                                                    .SBTTL  EMS.18 ERROR MESSAGE SUBROUTINES
000000  004737  000000V                 EMS.18::JSR     PC,M$EMS.18                             ;                               6945
000004  104423                                     TRAP    23
000006  000207                                     RTS     PC
```

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

;         6946  PRINTB (EBD_18, .CDISK);                    ! 'DISK XXX WENT OFFLINE"
;         6947  EMSCMD ();                                  ! PRINTx RELEVANT RETPKT FIELDS
;         6948  ENDMSG;

                                        .SBTTL  M$EMS.18 ERROR MESSAGE SUBROUTINES
000000  013746  000000G           M$EMS.18:
                                        MOV     CDISK,-(SP)                 ;                   6946
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  010156                  JSR     PC,EMSCMD                   ;                   6947
000024  062706  000006                  ADD     #6,SP                       ;                   6945
000030  000207                          RTS     PC

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

;         6949
;         6950  BGNMSG (EMS_21);
                                        .SBTTL  EMS.21 ERROR MESSAGE SUBROUTINES
000000  004737  000000V         EMS.21::JSR     PC,M$EMS.21                 ;                   6950
000004  104423                          TRAP    23
000006  000207                          RTS     PC

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

;         6951  EMSCMD ();                                  ! CONTENTS OF RETURN PACKET
;         6952  !EMS_RAL ();                                ! all return packets
;         6953  !EMS_MAL ();                                ! all message packets
;         6954  ENDMSG;

                                        .SBTTL  M$EMS.21 ERROR MESSAGE SUBROUTINES
000000  004737  010156'          M$EMS.21:
                                        JSR     PC,EMSCMD                   ;                   6951
000004  000207                          RTS     PC                          ;                   6950

; Routine Size:  3 words,      Routine Base:  $CODE$ + 20050
; Maximum stack depth per invocation:  1 word

;         6955
;         6956  BGNMSG (EMS_22)
```

```
                                        .SBTTL  EMS.22 ERROR MESSAGE SUBROUTINES
000000  004737  000000V        EMS.22::JSR     PC,M$EMS.22                     ;                       6956
000004  104423                          TRAP    23
000006  000207                          RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 20056
; Maximum stack depth per invocation:  2 words


;         6957  EMS_DBN ();                                 ! contents of dup buffer
;         6958  ENDMSG;


                                        .SBTTL  M$EMS.22 ERROR MESSAGE SUBROUTINES
000000  004737  015102'        M$EMS.22:
                                        JSR     PC,EMS.DBN                      ;                       6957
000004  000207                          RTS     PC                              ;                       6956

; Routine Size:  3 words,       Routine Base:  $CODE$ + 20066
; Maximum stack depth per invocation:  1 word


;         6959
;         6960  BGNMSG (EMS_30);

                                        .SBTTL  EMS.30 ERROR MESSAGE SUBROUTINES
000000  004737  000000V        EMS.30::JSR     PC,M$EMS.30                     ;                       6960
000004  104423                          TRAP    23
000006  000207                          RTS     PC

; Routine Size:  4 words,       Routine Base:  $CODE$ + 20074
; Maximum stack depth per invocation:  2 words


;         6961  EMSCMD ();                                  ! contents of return packet
;         6962  ENDMSG;


                                        .SBTTL  M$EMS.30 ERROR MESSAGE SUBROUTINES
000000  004737  010156'        M$EMS.30:
                                        JSR     PC,EMSCMD                       ;                       6961
000004  000207                          RTS     PC                              ;                       6960

; Routine Size:  3 words,       Routine Base:  $CODE$ + 20104
; Maximum stack depth per invocation:  1 word


;         6963
;         6964  end
;         6965
;         6966  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
;       $CODE$                          4133        RO ,  I  ,  LCL,  REL,  CON
;       $PLIT$                            12        RO ,  D  ,  LCL,  REL,  CON



;                              LIBRARY STATISTICS
;
;                                          -------  Symbols  -------      Blocks
;      File                                Total    Loaded    Percent      Read
;
;  SPIDER$USERS:[DOUCETTE.FALCON]CNRQAA.L16;7
;                                           457       351         76         82



;                                 COMMAND QUALIFIERS

;      BLISS /PDP11 CNRQAA.BL1/LIST=CNRQAA.LI1/OBJECT=CNRQAA.OB1/SOURCE=PAGE:56

; Size:          3953 code + 6736 data words
; Run Time:         03:45.9
; Elapsed Time:     11:11.3
; Memory Used:   824 pages
; Compilation Complete
```

```
;          0001  module NRQAM3 (
;          0002
;          0003  %title 'RD/RX EXERCISER
;          0004                  ident = 'V01.0 ,
;          0005                  addressing_mode (absolute),
;          0006                  environment (noeis)
;          0007                  ) =
;          0008
;          0009  !                       REVISION HISTORY
;          0010  !                       --------
;          0011  !
;          0012  !       The following changes were made to CZRQAB in producing CNRQAA for the
;          0013  !       FALCON-PLUS project (SBC 11/21+).  Release date December 19, 1983.
;          0014  !       Changes made by James S. Doucette.  All changes are marked by
;          0015  !       "!JSD REV A".
;          0016  !
;          0017  !       1. Set the ODT BREAK vector (location 140) to the starting address
;          0018  !             of FALCON's ODT ROM (170000-octal).
;          0019  !
;          0020  !       2. Lower the general operating priority of the program from level 7
;          0021  !             to level 6 to allow the BREAK key to interrupt and invoke
;          0022  !             ODT.
;          0023  !
;          0024  !       3. Due to space limitations, removed all references to DBM's (debug
;          0025  !             messages).
;          0026  !
;          0027  !       4. Changed the default IP address from 172150 to 176150.
;          0028  !
;          0029  !------ ------- -    -------- - ----   - - ------------------ -  --------
;          0030
;          0031  begin
;          0032
;          0033  %sbttl 'DECLARATIONS'
;          0034
;          0035  library 'CNRQAA.L16';                          ! RDRX EXERCISER GLOBAL LIBRARY
;          0036
;          0037  require 'BLSMAC.REQ';                          ! DIAGNOSTIC SUPERVISOR LIBRARY
;          1526
;          1527  EQUALS;
;          1528
;          1529  forward routine                               ! ROUTINES APPEAR IN THIS ORDER
;          1530      INIT_TEST : novalue,                      !    INDENTATION IMPLIES CALLED SUBROUTINE
;          1531          DRIVER_INIT : novalue,
;          1532              CTLR_INIT : novalue,
;          1533                  INI_CTLR_DAT : novalue,
;          1534                  REG_EXIST,
;          1535                  VEC_BR_TEST,
;          1536                  INT_GEN,
;          1537                  HARD_INIT,
;          1538                  INI_RRING : novalue,
;          1539                  SET_CTLR_CHAR,
;          1540              UNIT_INIT : novalue,
;          1541                  DR_ERR : novalue,
;          1542          ACCESS : novalue,
;          1543      MULTI_DRIVE : novalue,
;          1544          MD_INIT : novalue,
```

```
;          1545                INIT IO_BUFF : novalue,
;          1546            FATAL_ERROR : novalue,
;          1547            QIO_OK,
;          1548                QIO_OUT,
;          1549                QIO_GEN : novalue,
;          1550                    GET_RANDOM : novalue,
;          1551                    QIO_UNIT : novalue,
;          1552                    QIO_FUNC : novalue,
;          1553                      DUP : novalue,
;          1554                            DUPWRTDBN : novalue,
;          1555                            DUPREDDBN : novalue,
;          1556                                DUPCOMMAND : novalue,
;          1557                    QIO_LBN : novalue,
;          1558                    QIO_SIZE : novalue,
;          1559                    FILL_BUFF : novalue,
;          1560                PROC_RETPKT : novalue,
;          1561                    DIO_RETPKT : novalue,
;          1562                        DUP_COMPARE : novalue,
;          1563                    IO_RETPKT : novalue,
;          1564                        FSET_UPAR : novalue,
;          1565                        HARD_ERROR : novalue,
;          1566                        UPD_IO_TALLY : novalue,
;          1567                            OVF_CHK : novalue,
;          1568                        HOST_WRT_CHK,
;          1569                        SWEEP : novalue,
;          1570                            RPS_REM,
;          1571                    DR_RETPKT : novalue,
;          1572        AZINTO : L$ISR novalue,
;          1573            AZINT : novalue,
;          1574    !               FATAL_ERROR : novalue,
;          1575                    POLL_CRING : novalue,
;          1576                    POLL_RRING : novalue,
;          1577                        DUP_RSP : novalue,
;          1578                        MSCP_RSP : novalue,
;          1579                            SEQUEN : novalue,
;          1580                                SOFT_ERROR : novalue,
;          1581                            DATAGM : novalue;
;          1582    !                           SOFT_ERROR : novalue;
;          1583
;          1584    external
;          1585        CST : blockvector [MAX_CTLR, CST_LEN, word] field (CST_FIELDS),
;          1586                            ! RUN-TIME CONTROLLER STATUS TABLES
;          1587        CST_ADDR : ref block [CST_LEN, word] field (CST_FIELDS),
;          1588                            ! CONTROLLER STATUS TABLE ADDRESS OF "CURRENT" CONTROLLER
;          1589        DCT : blockvector [MAX_CTLR, DCT_LEN, word] field (DCT_FIELDS),
;          1590                            ! DRIVER CONTROLLER TABLES
;          1591        DCT_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS),
;          1592                            ! ADDRESS OF "CURRENT" DRIVER CONTROLLER TABLE
;          1593        RDRX_ADDR : ref rdrx field (RC_REG),
;          1594                            ! DEVICE ADDRESS OF "CURRENT" CONTROLLER
;          1595        IRDRX_ADDR : ref rdrx field (RC_REG),
;          1596                            ! DEVICE ADDRESS OF INTERRUPTING CONTROLLER
;          1597        DUPPKT : BLOCK [257, WORD] field (DP_FIELDS),
;          1598                            ! BUFFER CONTAINING DUP INFORMATION FROM RECEIVE AND SEND COMMANDS
;          1599        TALLY : vector [MAX_UNITS * TALLY_LEN, word] field (T_FIELDS),
;          1600                            ! STATISTICS TABLES
```

```
;  1601      T_ADDR : ref block [TALLY_LEN, word] field (T_FIELDS),
;  1602                          ! ADDRESS OF STATISTICS TABLE (TALLY) FOR CURRENT UNIT
;  1603      C_ERR_TBL : blockvector [MAX_CTLR, C_ERR_LEN, word] field (C_ERR FIELDS),
;  1604                          ! STATISTICS TABLE FOR CONTROLLER ERRORS
;  1605      MSCP_PKT : blockvector [PKT_CNT, PKT_LEN, word] field (PKT_FIELDS),
;  1606                          ! MSCP PACKET POOL
;  1607      IPKT_ADDR : ref block [PKT_LEN, word] field (PKT_FIELDS),
;  1608                          ! ADDRESS OF AN MSCP PACKET (INTERRUPT PROCESSING)
;  1609      PKT_USE : vector [PKT_CNT, byte, signed],
;  1610                          ! MSCP PACKET POOL ALLOCATION TABLE
;  1611      RETPKT : blockvector [RP_CNT, RP_LEN, word] field (RP_FIELDS),
;  1612                          ! RETURN PACKET POOL
;  1613      RP_USE : vector [RP_CNT, byte, signed],
;  1614                          ! RETURN PACKET POOL ALLOCATION TABLE
;  1615      RP_INDX : word,     ! CURRENT RETURN PACKET INDEX
;  1616      RP_ADDR : ref block [RP_LEN, word] field (RP_FIELDS),
;  1617                          ! CURRENT RETURN PACKET ADDRESS
;  1618      ELOG_PKT : blockvector [EP_CNT, EP_LEN, word] field (EP_FIELDS),
;  1619                          ! ERROR-LOG PACKET SAVE AREA
;  1620      BUFF_ADDR : vector [MAX_BUF_CNT],   ! TABLE OF I/O BUFFER DESCRIPTORS
;  1621      BUFF_OWN : vector [MAX_BUF_CNT, byte, signed],      ! I/O BUFFER OWNERSHIP (CONTROLLER NUMBER)
;  1622      IODQ : vector [IODQ_LEN, byte],
;  1623                                          ! I/O DONE QUEUE - CIRCULAR QUEUE OF RETPKT INDECES
;  1624      IODQ_IN : word,                     ! I/O DONE QUEUE IN POINTER
;  1625      IODQ_OUT : word,                    ! I/O DONE QUEUE OUT POINTER
;  1626      ENTRY_REASON : byte,                ! CURRENT OPERATOR COMMAND
;  1627      EOP_FLAG : byte,                    ! END-OF-PASS FLAG
;  1628      DUP_FLAGS : WORD,                   ! DUP FLAGS
;  1629      CCTLR : word,                       ! NUMBER OF "CURRENT" CONTROLLER
;  1630      CDISK : word,                       ! CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;  1631      CUOFF : word,                       ! CURRENT UNIT CST OFFSET
;  1632      CTLR_CNT : word,                    ! TOTAL NUMBER OF CONFIGURED CONTROLLERS
;  1633      DUR : vector [MAX_UNITS, byte],     ! DROP UNIT REASON
;  1634      QIO : vector [MAX_CTLR, byte],      ! NUMBER OF OUTSTANDING QIOS PER CONTROLLER
;  1635      FREE_MEM_ADDR,                      ! START OF FREE MEMORY
;  1636      BYTS_PER_QIO : word,                ! SIZE (BYTES) OF AN I/O BUFFER
;  1637      ST_CODE : word,                     ! CURRENT STATUS CODE
;  1638      SB_CODE : word,                     ! CURRENT SUB-CODE
;  1639      STEP : word,                        ! CURRENT STEP IN HARD_INIT
;  1640      OF_RC : signed word,                ! OFFSET (0 OR 2) TO READ IP OR SA
;  1641      SA_REG : word,                      ! STORAGE FOR SA REGISTER READS AND WRITES
;  1642      CMD_TIME : word,                    ! COMMAND TIMEOUT VALUE (IN SECONDS)
;  1643      NEX : word,                         ! NON-EXISTENT MEMORY TRAP INDICATOR
;  1644      CRN_LOW : word,                           ! COMMAND REF NUMBER OF LAST COMMAND SENT
;  1645      CRN_HIGH : word,                          ! COMMAND REF NUMBER OF LAST COMMAND SENT
;  1646      P_INDEX : signed word,       ! CURRENT message PACKET INDEX
;  1647      S_DUPPKT : WORD,             ! DBN BYTE COUNTER
;  1648      S_PATTERN : WORD,            ! THE PATTERN WRITTEN TO DBN'S
;  1649      CREDIT_BAL : word,                  ! CREDIT BALANCE
;  1650      NEXT_PKT USE : byte,                ! POINTER TO NEXT ENTRY IN PKT_USE TABLE
;  1651  !    DBM12,                             !JSD REV A - REMOVED
;  1652  !    DBM18,                             !JSD REV A - REMOVED
;  1653  !    DBM19,                             !JSD REV A - REMOVED
;  1654  !    DBM20,                             !JSD REV A - REMOVED
;  1655  !    DBM21,                             !JSD REV A - REMOVED
;  1656      'DBM22,
```

```
;          1657  !    DBM23.                    !JSD REV A - REMOVED
;          1658  !    DBM25.                    !JSD REV A - REMOVED
;          1659  !    DBM26.                    !JSD REV A   REMOVED
;          1660  !    DBM29.                    !JSD REV A   REMOVED
;          1661  !    DBM108.                   !JSD REV A - REMOVED
;          1662  !    DBM109.                   !JSD REV A   REMOVED
;          1663  !    DBM110.                   !JSD REV A - REMOVED
;          1664  !    DBM112.                   !JSD REV A - REMOVED
;          1665       EM_0,
;          1666       EM_1,
;          1667       EM_2,
;          1668       EM_3,
;          1669       EM_4,
;          1670       EM_5,
;          1671       EM_6,
;          1672       EM_7,
;          1673       EM_8,
;          1674       EM_9,
;          1675       EM_10,
;          1676       EM_12,
;          1677       EM_13,
;          1678       MSG_02,
;          1679       MSG_03,
;          1680       EGS_02,
;          1681       EGD_10,
;          1682       EGD_11,
;          1683       EGD_12,
;          1684       EGD_13,
;          1685       EGD_14,
;          1686       EGD_15,
;          1687       EGD_16,
;          1688       EGD_17,
;          1689       EGD_18,
;          1690       EGD_20,
;          1691       EGD_21,
;          1692       EGD_22,
;          1693       EGD_23,
;          1694       EGH_30,
;          1695       CRLF,
;          1696       SWP_ERROR : word,                       ! HARD ERROR LIMIT FOR DROPPING UNIT
;          1697       SWP_XFER : word,                        ! TRANSFER LIMIT FOR DROPPING UNIT
;          1698       SWP_FLAGS : word,                       ! FLAGS (SEE DOCUMENTATION)
;          1699       dupround : word,                        ! ratio constant for dup exerciser
;          1700       SWP_RAT : word,                         ! RD51 OPERATION RATIO
;          1701       SWP_DPAT : word,                        ! DATA PATTERN NUMBER
;          1702       SWP_UCNT : word,                        ! USER DATA PATTERN COUNT
;          1703       SWP_UDPAT : vector [MAX_UDP_CNT, word], ! USER DATA PATTERN
;          1704       L$LUN,
;          1705       L$UNIT;
;          1706
;          1707  psect
;          1708       own = $GGG$(read, nowrite, execute, local, concatenate);
;          1709
;          1710  own
;          1711       COMM_AREA : blockvector [MAX_CTLR, COMM_LEN, word] field (COM_FIELDS),
;          1712                                ! COMMUNICATIONS AREA BETWEEN HOST AND AZTEC CONTROLLERS
```

```
;       1713      BST : vector [MAX_UNITS, word, signed],
;       1714                    ! BLOCK SEQUENCE TABLE FOR SEQUENTIAL LBN (VS. RANDOM SEEK) MODE
;       1715      DPST : vector [MAX UNITS, byte],              ! DATA PATTERN SEQUENCE TABLE
;       1716      MAX_LBN : vector [MAX_UNITS, word],           ! LARGEST LBN ALLOWED
;       1717      STORAGE : vector [MAX_UNITS, word],           ! DUMMY STORAGE
;       1718      ICOM_ADDR : ref block [COMM_LEN, word] field (COM_FIELDS),
;       1719                    ! ADDRESS OF INTERRUPTING CONTROLLER'S COMMUNICATION AREA
;       1720      ICST_ADDR : ref block [CST_LEN, word] field (CST_FIELDS),
;       1721                    ! ADDRESS OF INTERRUPTING CONTROLLER'S CST
;       1722      IDCT_ADDR : ref block [DCT_LEN, word] field (DCT_FIELDS),
;       1723                    ! ADDRESS OF INTERRUPTING CONTROLLER'S DCT
;       1724      INT_ADDR : vector [MAX_CTLR] initial (AZINTO %(, AZINT1, AZINT2, AZINT3)%),
;       1725                    ! INTERRUPT SERVICE ROUTINE ADDRESS TABLE
;       1726      RDM_CNT : word initial (RDM_LEN),             ! NUMBER OF RANDOM NUMBERS  \ KEEP
;       1727      RANDOM : vector [RDM_LEN, word],              ! RANDOM NUMBER TABLE       / TOGETHER
;       1728      ICTLR : word,                                 ! INTERRUPTING CONTROLLING NUMBER
;       1729      MX1 : signed word,                            ! MSCP PKT INDEX FOR FIRST QIO
;       1730      MX2 : signed word,                            ! MSCP PKT INDEX FOR SECOND QIO
;       1731      MAD1 : ref block [PKT_LEN, word] field (PKT_FIELDS),
;       1732                    ! ADDRESS OF MSCP PACKET FOR FIRST QIO
;       1733      MAD2 : ref block [PKT_LEN, word] field (PKT_FIELDS),
;       1734                    ! ADDRESS OF MSCP PACKET FOR SECOND QIO
;       1735      LAST_PKT : blockvector [MAX_CTLR, LAST_PKT_LEN, word] field (LAST_PKT_FIELDS),
;       1736                    ! SAVE AREA FOR INFO ABOUT LAST RESPONSE PACKET
;       1737      PAT02 : vector [2] initial (1,                                ! PATTERN 2
;       1738              %o'000000'),
;       1739      PAT03 : vector [2] initial (1,                                ! PATTERN 3
;       1740              %o'177777'),
;       1741      PAT04 : vector [2] initial (1,                                ! PATTERN 4
;       1742              %o'105613'),
;       1743      PAT05 : vector [2] initial (1,                                ! PATTERN 5
;       1744              %o'031463'),
;       1745      PAT06 : vector [2] initial (1,                                ! PATTERN 6
;       1746              %o'030221'),
;       1747      PAT07 : vector [17] initial (16,                              ! PATTERN 7
;       1748              %o'000001', %o'000003', %o'000007', %o'000017',
;       1749              %o'000037', %o'000077', %o'000177', %o'000377',
;       1750              %o'000777', %o'001777', %o'003777', %o'007777',
;       1751              %o'017777', %o'037777', %o'077777', %o'177777'),
;       1752      PAT08 : vector [17] initial (16,                              ! PATTERN 8
;       1753              %o'177776', %o'177774', %o'177770', %o'177760',
;       1754              %o'177740', %o'177700', %o'177600', %o'177400',
;       1755              %o'177000', %o'176000', %o'174000', %o'170000',
;       1756              %o'160000', %o'140000', %o'100000', %o'000000'),
;       1757      PAT09 : vector [17] initial (16,                              ! PATTERN 9
;       1758              rep 3 of (%o'000000'), rep 3 of (%o'177777'),
;       1759              rep 2 of (%o'000000'), rep 2 of (%o'177777'),
;       1760              %o'000000', %o'177777', %o'000000', %o'177777',
;       1761              %o'000000', %o'177777'),
;       1762      PAT10 : vector [2] initial (1,                                ! PATTERN 10
;       1763              %o'133331'),
;       1764      PAT11 : vector [17] initial (16,                              ! PATTERN 11
;       1765              rep 3 of (%o'052525'), rep 3 of (%o'125252'),
;       1766              rep 2 of (%o'052525'), rep 2 of (%o'125252'),
;       1767              %o'052525', %o'125252', %o'052525', %o'125252',
;       1768              %o'052525', %o'125252'),
```

M2


```
;       1769      PAT12 : vector [21] initial (20,                                ! PATTERN 12
;       1770                rep 3 of (%o'026455'), rep 3 of (%o'151322'),
;       1771                rep 2 of (%o'026455'), rep 2 of (%o'151322'),
;       1772                rep 2 of (%o'026455'),
;       1773                %o'151322', %o'026455', %o'151322', %o'026455',
;       1774                %o'151322', %o'026455', %o'151322', %o'026455'),
;       1775      PAT13 : vector [2] initial (1,                                  ! PATTERN 13
;       1776                %o'066666'),
;       1777      PAT14 : vector [17] initial (16,                                ! PATTERN 14
;       1778                %o'000001', %o'000002', %o 000004', %o'000010',
;       1779                %o'000020', %o'000040 ', %o'000100', %o'000200',
;       1780                %o'000400', %o'001000', %o 002000', %o'004000',
;       1781                %o'010000', %o'020000', %o'040000', %o'100000'),
;       1782      PAT15 : vector [17] initial (16,                                ! PATTERN 15
;       1783                %o'177776', %o'177775', %o'177773', %o'177767',
;       1784                %o'177757', %o'177737', %o'177677', %o'177577',
;       1785                %o'177377', %o'176777', %o'175777', %o'173777',
;       1786                %o'167777', %o'157777', %o'137777', %o'077777'),
;       1787      PAT16 : vector [17] initial (16,                                ! PATTERN 16
;       1788                rep 3 of (%o'133331'), rep 3 of (%o'155554'),
;       1789                rep 2 of (%o'133331'), rep 2 of (%o'155554'),
;       1790                %o'133331', %o'155554', %o'133331', %o'155554',
;       1791                %o'133331', %o'155554'),
;       1792      PAT17 : vector [22] initial (21,                                ! PATTERN 17
;       1793                %o'000000', rep 2 of (%o'106466'),
;       1794                rep 3 of (%o'071311'), rep 4 of (%o'106466'),
;       1795                rep 5 of (%o'071311'), rep 6 of (%o'106466')),
;       1796      PAT18 : vector [22] initial (21,                                ! PATTERN 18
;       1797                %o'106466', %o'000000', %o'071311',
;       1798                rep 3 of (%o'106466'), rep 4 of (%o'071311'),
;       1799                rep 5 of (%o'106466'), rep 6 of (%o'071311')),
;       1800      PAT19 : vector [22] initial (21,                                ! PATTERN 19
;       1801                %o'000000', rep 2 of (%o'134631'),
;       1802                rep 3 of (%o'043146'), rep 4 of (%o'134631'),
;       1803                rep 5 of (%o'043146'), rep 6 of (%o'134631')),
;       1804      PAT20 : vector [22] initial (21,                                ! PATTERN 20
;       1805                %o'134631', %o'000000', %o'043146',
;       1806                rep 3 of (%o'134631'), rep 4 of (%o'043146'),
;       1807                rep 5 of (%o'134631'), rep 6 of (%o'043146')),
;       1808      PAT21 : vector [2] initial (1,                                  ! PATTERN 21
;       1809                %o'000000'),                                          !    (LBN)
;       1810      DPA_TBL : vector [DP_CNT] initial                               ! DATA PATTERN ADDRESS TABLE
;       1811                  (RDM_CNT, PAT02, PAT03, PAT04, PAT05,
;       1812                  PAT06, PAT07, PAT08, PAT09, PAT10, PAT11,
;       1813                  PAT12, PAT13, PAT14, PAT15, PAT16, PAT17,
;       1814                  PAT18, PAT19, PAT20, PAT21),
;       1815      TRK_SGN : vector [MAX_UNITS, word, signed] initial (rep MAX_UNITS of (1)),          ! CURRENT TRACK DIRECTION
;       1816      BST_CNT : word initial (0),                                     ! CURRENT SEQUENTIAL BLOCK COUNT
;       1817      BST_DEV : word initial (0),                                     ! CURRENT SEQUENTIAL BLOCK DEVICE
;       1818      CURRENT_VECTOR : word,                                          ! CURRENT DEVICE'S VECTOR ADDRESS
;       1819      BRLEVEL : word,                                                 ! CURRENT DEVICE'S BR LEVEL
;       1820      COMPARE_DATA : byte;                                            ! FLAGGED CLEARED TO BYPASS HOST COMPARES
;       1821
;       1822  external routine
;       1823      NEX_TRAP : L%ISR novalue,
;       1824      SET_CPAR : novalue,
```

N2


```
;       1825      SET_UPAR : novalue,
;       1826      OUT_IODQ,
;       1827      IN_IODQ : novalue,
.       1828      GET_PKT,
;       1829      PUT_PKT : novalue,
.       1830      GET_RETPKT,
;       1831      PUT_RETPKT : novalue,
.       1832      GET_IO_BUFF : novalue,
;       1833      PUT_IO_BUFF : novalue,
;       1834      PUTA_BUFF : novalue,
.       1835      SEND,
;       1836      WAIT : novalue,
;       1837      DROP_CTLR : novalue,
;       1838      DRV_CTLERR : novalue,
.       1839      EMSCMD : novalue,
;       1840      EMS_22 : novalue,
;       1841      EMS_EL : novalue,
;       1842      EMS_CMP : novalue,
.       1843      EMS_10 : novalue,
;       1844      EMS_12 : novalue,
;       1845      EMS_13 : novalue,
;       1846      EMS_14 : novalue,
;       1847      EMS_18 : novalue,
.       1848      EMS_21 : novalue,
.       1849      EMS_30 : novalue;
```

```
;          1850  %sbttl  TEST SECTION
;          1851
;          1852  !+
;          1853  !       THIS SECTION CONTAINS THE TOP LEVEL TEST CODE FOR THE RDRX EXERCISER.
;          1854  !       THE EXERCISER CONSISTS OF ONE TEST WHICH IS SUBDIVIDED INTO A NUMBER OF
;          1855  !       SUBTESTS. ALL SUBTESTS ARE DECLARED WITHIN THIS BLOCK.
;          1856  !-
;          1857
;          1858  BGNTST;
;          1859
;          1860  EOP_FLAG = TRUE;                                  ! ASSUME NO UNIT AVAILABLE
;          1861  COMPARE_DATA = TRUE;                              ! ALLOW HOST COMPARES IF ASKED FOR
;          1862  INIT_TEST ();                                     ! INITIALIZE TEST ENVIRONMENT
;          1863
;          1864  incr CTLR from 0 to (MAX_CTLR - 1) do            ! FOR EVERY CONTROLLER
;          1865
;          1866      if (.CST [.CTLR, STATE] eql ONLINE) and       ! IF CONTROLLER ONLINE
;          1867         (.DCT [.CTLR, STAT] eql ONLINE) and
;          1868         (.CST [.CTLR, U_CNT] gequ 0)
;          1869      then
;          1870
;          1871          incr OFFSET from (0 * OF_UN) to (3 * UNIT_SIZE - 4) by UNIT_SIZE do
;          1872
;          1873              if .CST [.CTLR, .OFFSET, D STAT] eql ONLINE ! IF AT LEAST ONE UNIT ALIVE
;          1874              then
;          1875                  begin
;          1876                  EOP_FLAG = FALSE;                 ! NOT END OF PASS
;          1877                  exitloop;
;          1878                  end;
;          1879
;          1880  if not .EOP_FLAG
;          1881  then
;          1882      MULTI_DRIVE ();                               ! RUN MULTI-DRIVE TEST
;          1883
;          1884  ENDTST;

                                        .TITLE  NRQAM3 RD/Rx EXERCISER
                                        .IDENT  /V01.0/
                                        .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   AZINTO
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  12525                           .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                  PA120:  .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, 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, MSG.02
                                .GLOBL  MSG.03, EGS.02, EGD.10, EGD.11
                                .GLOBL  EGD.12, EGD.13, EGD.14, EGD.15
                                .GLOBL  EGD.16, EGD.17, EGD.18, EGD.20
                                .GLOBL  EGD.21, EGD.22, EGD.23, EGH.30
                                .GLOBL  CRLF, SWP.ERROR, SWP.XFER, SWP.FLAGS
                                .GLOBL  DUPROUND, SWP.RAT, SWP.DPAT, SWP.UCNT
                                .GLOBL  SWP.UDPAT, L$LUN, L$UNIT, NEX.TRAP
                                .GLOBL  SET.CPAR, SET.UPAR, OUT.IODQ, IN.IODQ
                                .GLOBL  GET.PKT, 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                       ;                               1849
000004  112737  000001  000000G       MOVB    #1,EOP.FLAG                     ;                               1860
000012  112737  000001  001240'       MOVB    #1,COMPARE.DATA                 ;                               1861
000020  004737  000000V               JSR     PC,INIT.TEST                    ;                               1862
000024  005003                        CLR     R3                              ; CTLR                          1864
000026  010346                1$:     MOV     R3,-(SP)                        ; CTLR,*                        1866
000030  012746  000056                MOV     #56.,-(SP)
000034  004737  000000G               JSR     PC,BL$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,*                        1867
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,*                        1873
000076  012746  000027                        MOV     #27, (SP)
000102  004737  000000G                        JSR     PC,BL$MUL
000106  012702  000003                        MOV     #3,R2                   ; *,OFFSET                      1871
000112  010001                          2$:   MOV     R0,R1                   ;                               1873
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                ;                               1876
000134  000405                                BR      4$                      ;                               1875
000136  062702  000005                  3$:   ADD     #5,R2                   ; *,OFFSET                      1871
000142  020227  000023                        CMP     R2,#23                  ; OFFSET,*
000146  003761                                BLE     2$
000150  022626                          4$:   CMP     (SP)+,(SP)+
000152  005203                          5$:   INC     R3                      ; CTLR                          1864
000154  000243                                .WORD CLV!CLC

000156  003723                                BLE     1$
000160  132737  000001  000000G               BITB    #1,EOP.FLAG             ;                               1880
000166  001002                                BNE     6$
000170  004737  000000V                        JSR     PC,MULTI.DRIVE          ;                               1882
000174  000207                          6$:   RTS     PC                      ;                               1849

; 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                  ;                               1882
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
```

```
;          1885  %sbttl 'INITIALIZATION TEST ROUTINES'
;          1886
;          1887  routine INIT_TEST : novalue =
;          1888
;          1889  !+
;          1890  !       THE INITIALIZATION TEST IS DESIGNED TO VERIFY THE EXISTENCE OF THE
;          1891  !       DEVICES AS CONFIGURED BY THE OPERATOR DURING THE HW DIALOG, AND TO
;          1892  !       BRING EACH DEVICE ONLINE IN PREPARATION FOR EITHER THE MULTI-DRIVE TEST
;          1893  !       OR THE DM EXERCISER.
;          1894  !
;          1895  !       BASICALLY, THE DEVICES ARE BROUGHT ONLINE VIA "DRIVER_INIT", WHICH IS
;          1896  !       INVOKED IMMEDIATELY. ANY DEVICES WHICH FAIL DURING THIS PHASE WILL BE
;          1897  !       MARKED OFFLINE IN THEIR DCT AND CST. FOR THOSE DEVICES WHICH SURVIVE
;          1898  !       THE INITIALIZATION, THIS ROUTINE WILL ATTEMPT 1 OR 2 ACCESS COMMANDS TO
;          1899  !       EACH DISK VIA ROUTINE "ACCESS". THE INITIALIZATION TEST IS DEEMED A
;          1900  !       SUCCESS IF A BLOCK ON THE INNER TRACK OF EACH DISK CAN BE ACCESSED.
;          1901  !-
;          1902
;          1903      begin
;          1904      DRIVER_INIT ();                                 ! INIT DRIVER DATA AND DEVICES
;          1905
;          1906      incr CTLR from 0 to (MAX_CTLR - 1) do           ! FOR EACH CONTROLLER
;          1907          begin
;          1908          SET_CPAR (.CTLR);                           ! SET UP COMMONLY-USED CONTROLLER-RELATED DATA ITEMS
;          1909
;          1910          if .CST_ADDR [STATE] eql ONLINE             ! IF CONTROLLER IS STILL ALIVE
;          1911          then
;          1912
;          1913              incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do     ! FOR EACH DISK
;          1914
;          1915                  if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and
;          1916                     (.CST_ADDR [.OFFSET, D_STAT] eql ONLINE) and
;          1917                     (not .CST_ADDR [.OFFSET, D_FATAL])
;          1918                  then
;          1919                      begin
;          1920                      SET_UPAR (.OFFSET);             ! SET UP UNIT-RELATED DATA ITEMS
;          1921                      IF SWP_DINT NEQ (.DUP_FLAGS AND SWP_DINT)   ! IF DUP CAUSED INIT THEN SKIP THIS SECTION
;          1922                          THEN ACCESS ();                         ! TRY ACCESS TO INNER TRACK
;          1923                      end;                            ! IF UNIT IS PRESENT AND ONLINE
;          1924
;          1925          end;                                        ! CONTROLLER LOOP
;          1926
;          1927      end;                                            ! ROUTINE INIT_TEST


                                    .SBTTL  INIT.TEST INITIALIZATION TEST ROUTINES
000000  004137  000000G       INIT.TEST:
                                    JSR     R1,$SAVE2                       ;                                       1887
000004  004737  000000V               JSR     PC,DRIVER.INIT                  ;                                       1904
000010  005002                        CLR     R2                              ; CTLR                                  1906
000012  010246                1$:     MOV     R2,-(SP)                        ; CTLR,*                                1908
000014  004737  000000G               JSR     PC,SET.CPAR
000020  013700  000000G               MOV     CST.ADDR,R0                     ;                                       1910
000024  005760  000002                TST     2(R0)
000030  100035                        BPL     4$
000032  012701  000003                MOV     #3,R1                           ; *,OFFSET                              1913
```

```
000036  010100                          2$:     MOV     R1,R0                   ; OFFSET,*                      1915
000040  006300                                  ASL     R0
000042  063700  000000G                         ADD     CST.ADDR,R0
000046  032710  040000                          BIT     #40000,(R0)
000052  001417                                  BEQ     3$
000054  032710  020000                          BIT     #20000,(R0)             ;                               1916
000060  001414                                  BEQ     3$
000062  032710  010000                          BIT     #10000,(R0)             ;                               1917
000066  001011                                  BNE     3$
000070  010116                                  MOV     R1,(SP)                 ; OFFSET,*                      1920
000072  004737  000000G                         JSR     PC,SET.UPAR
000076  032737  000002  000000G                 BIT     #2,DUP.FLAGS            ;                               1921
000104  001002                                  BNE     3$
000106  004737  000000V                         JSR     PC,ACCESS               ;                               1922
000112  062701  000005                  3$:     ADD     #5,R1                   ; *,OFFSET                      1913
000116  020127  000022                          CMP     R1,#22                  ; OFFSET,*
000122  003745                                  BLE     2$
000124  005726                          4$:     TST     (SP)+                   ;                               1907
000126  005202                                  INC     R2                      ; CTLR                          1906
000130  000243                                  .WORD CLV!CLC

000132  003727                                  BLE     1$
000134  000207                                  RTS     PC                      ;                               1887

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

```
;          1928  routine DRIVER_INIT : novalue =
;          1929
;          1930  !+
;          1931  !       THIS ROUTINE IS EQUIVALENT IN FUNCTION TO THE INITIALIZATION ENTRY
;          1932  !       POINT OF A STANDARD DEVICE DRIVER. ITS RESPONSIBILITY IS TO INITIALIZE
;          1933  !       DRIVER DATA, AND TO BRING EACH RDRX CONTROLLER AND UNIT (DISK)
;          1934  !       ONLINE.
;          1935  !-
;          1936
;          1937      begin
;          1938
;          1939      local
;          1940          PKT_ADDR;
;          1941
;          1942      PKT_ADDR = MSCP_PKT + 10;                   ! ADDR (TEXT + 0) OF FIRST MSCP PACKET
;          1943      NEXT_PKT_USE = 0;                           ! NEXT PACKET TO ALLOCATE
;          1944
;          1945      incr COUNT from 0 to (PKT_CNT - 1) do       ! FOR EACH MSCP PACKET
;          1946          begin
;          1947          PKT_USE [.COUNT] = -1;                  ! MARK PACKET FREE
;          1948          MSCP_PKT [.COUNT, PKT_LO] = .PKT_ADDR;  ! LOAD PKT ADDR INTO BUFFER DESCRIPTOR
;          1949          MSCP_PKT [.COUNT, PKT_HI] = 0;
;          1950          MSCP_PKT [.COUNT, CONNID] = CID_MSCP;   ! SET CONNECTION ID TO MSCP ID
;          1951          PKT_ADDR = .PKT_ADDR + (PKT_LEN  *  2); ! ADVANCE ADDR TO NEXT PACKET
;          1952          end;
;          1953
;          1954      incr CTLR from 0 to (MAX_CTLR - 1) do       ! FOR EACH CONTROLLER
;          1955
;          1956          if .CST [.CTLR, IP_ADDR] neqa 0         ! IF CONTROLLER IS PRESENT
;          1957          then
;          1958              begin
;          1959              SET_CPAR (.CTLR);                   ! SET UP CURRENT CONTROLLER PARAMETERS
;          1960              CURRENT_VECTOR = .CST_ADDR [VEC_ADDR];      ! SET CURRENT CONTROLLER'S VECTOR ADDRESS
;          1961              BRLEVEL = .CST_ADDR [BR_LEV] ^ 5;   ! SET CURRENT CONTROLLER'S BR LEVEL
;          1962              CTLR_INIT ();                       ! INIT DEVICE AND CTLR DATA
;          1963
;          1964              if .DCT_ADDR [STAT] eql ONLINE      ! IF CONTROLLER IS STILL ALIVE
;          1965              then
;          1966
;          1967                  incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do ! FOR EACH UNIT (DISK)
;          1968
;          1969                      if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and    ! IF UNIT EXISTS
;          1970                         (not .CST_ADDR [.OFFSET, D_FATAL])
;          1971                      then
;          1972                          begin
;          1973                          SET_UPAR (.OFFSET);     ! SET UP UNIT-RELATED DATA ITEMS
;          1974                          UNIT_INIT ();           ! BRING UNIT ONLINE
;          1975                          end;                    ! IF UNIT EXISTS
;          1976
;          1977              end;                                ! IF CONTROLLER IS PRESENT
;          1978
;          1979      end;                                        ! ROUTINE DRIVER_INIT


                                    .SBTTL  DRIVER.INIT INITIALIZATION TEST ROUTINES
000000  004137  000000G            DRIVER.INIT:
```

```
                                        JSR     R1,$SAVE2                       ;                                       1928
000004  012702  000012G                 MOV     #MSCP.PKT+12,R2                 ; .,PKT.ADDR                            1942
000010  105037  000000G                 CLRB    NEXT.PKT.USE                    ;                                       1943
000014  005001                          CLR     R1                              ; COUNT                                 1945
000016  112761  000377  000000G  1$:    MOVB    #377,PKT.USE(R1)                ; .,@(COUNT)                            1947
000024  010146                          MOV     R1,-(SP)                        ; COUNT,.                               1948
000026  012746  000104                  MOV     #104,-(SP)
000032  004737  000000G                 JSR     PC,BL$MUL
000036  010260  000000G                 MOV     R2,MSCP.PKT(R0)                 ; PKT.ADDR,.
000042  005060  000002G                 CLR     MSCP.PKT+2(R0)                  ;                                       1949
000046  105060  000011G                 CLRB    MSCP.PKT+11(R0)                 ;                                       1950
000052  062702  000104                  ADD     #104,R2                         ; .,PKT.ADDR                            1951
000056  022626                          CMP     (SP)+,(SP)+                     ;                                       1946
000060  005201                          INC     R1                              ; COUNT                                 1945
000062  020127  000013                  CMP     R1,#13                          ; COUNT,.
000066  003753                          BLE     1$
000070  005002                          CLR     R2                              ; CTLR                                  1954
000072  010246                  2$:     MOV     R2,-(SP)                        ; CTLR,.                                1956
000074  012746  000056                  MOV     #56,-(SP)
000100  004737  000000G                 JSR     PC,BL$MUL
000104  022626                          CMP     (SP)+,(SP)+
000106  005760  000000G                 TST     CST(R0)
000112  001460                          BEQ     6$
000114  010246                          MOV     R2,-(SP)                        ; CTLR,.                                1959
000116  004737  000000G                 JSR     PC,SET.CPAR
000122  013700  000000G                 MOV     CST.ADDR,R0                     ;                                       1960
000126  016037  000002  001234'         MOV     2(R0),CURRENT.VECTOR
000134  042737  177000  001234'         BIC     #177000,CURRENT.VECTOR
000142  005016                          CLR     (SP)                            ;                                       1961
000144  116016  000004                  MOVB    4(R0),(SP)
000150  012746  000005                  MOV     #5,-(SP)
000154  004737  000000G                 JSR     PC,BL$SHF
000160  010037  001236'                 MOV     R0,BRLEVEL
000164  004737  000000V                 JSR     PC,CTLR.INIT                    ;                                       1962
000170  005777  000000G                 TST     @DCT.ADDR                       ;                                       1964
000174  100026                          BPL     5$
000176  012701  000003                  MOV     #3,R1                           ; .,OFFSET                              1967
000202  010100                  3$:     MOV     R1,R0                           ; OFFSET,.                              1969
000204  006300                          ASL     R0
000206  063700  000000G                 ADD     CST.ADDR,R0
000212  032710  040000                  BIT     #40000,(R0)
000216  001410                          BEQ     4$
000220  032710  010000                  BIT     #10000,(R0)                     ;                                       1970
000224  001005                          BNE     4$
000226  010116                          MOV     R1,(SP)                         ; OFFSET,.                              1973
000230  004737  000000G                 JSR     PC,SET.UPAR
000234  004737  000000V                 JSR     PC,UNIT.INIT                    ;                                       1974
000240  062701  000005          4$:     ADD     #5,R1                           ; .,OFFSET                              1967
000244  020127  000022                  CMP     R1,#22                          ; OFFSET,.
000250  003754                          BLE     3$
000252  022626                  5$:     CMP     (SP)+,(SP)+                     ;                                       1958
000254  005202                  6$:     INC     R2                              ; CTLR                                  1954
000256  000243                          .WORD CLV!CLC
000260  003704                          BLE     2$
000262  000207                          RTS     PC                              ;                                       1928
; Routine Size:  90 words,      Routine Base:  $CODE$ + 0350
; Maximum stack depth per invocation:  6 words
```

```
;        1980  routine CTLR_INIT : novalue =
;        1981
;        1982  !+
;        1983  !       THIS "DRIVER" ROUTINE IS CALLED FROM DRIVER_INIT FOR EACH CONTROLLER
;        1984  !       CONFIGURED FOR TESTING. ITS GENERAL PURPOSE IS TO BRING THE RDRX ONLINE
;        1985  !       TO THE HOST. SPECIFICALLY, IT IS WRITTEN TO:
;        1986  !
;        1987  !       1. INITIALIZE DRIVER CONTROLLER DATA, INCLUDING THE DCT,
;        1988  !       2. SET UP THE DEVICE'S INTERRUPT VECTOR ADDRESS,
;        1989  !       3. PERFORM A REGISTER EXISTENCE TEST TO VERIFY THE DEVICE'S PRESENCE,
;        1990  !       4. PERFORM A VECTOR AND BR LEVEL TEST TO VERIFY THE DEVICE S VECTOR
;        1991  !            ADDRESS AND INTERRUPT REQUEST LEVEL.
;        1992  !       5. DO A HARD INITIALIZATION (FOUR STEPS) ON THE DEVICE.
;        1993  !
;        1994  !       IF ANY OF THESE INITIAL TESTS FAIL, THEN ALL UNITS ASSOCIATED WITH THE
;        1995  !       DEVICE ARE DROPPED.
;        1996  !
;        1997
;        1998      begin
;        1999
;        2000      local
;        2001          RESULT : byte;
;        2002
;        2003      INI_CTLR_DAT ();                          ! INITIALIZE CONTROLLER DATA
;        2004  !    SETVEC (.CURRENT_VECTOR, .INT_ADDR [.CCTLR], PRIO7);       ! SET DEVICE S ASSUMED VECTOR ADDRESS !JSD REV A
;        2005      SETVEC (.CURRENT_VECTOR, .INT_ADDR [.CCTLR], PRIO6);       ! SET DEVICE S ASSUMED VECTOR ADDRESS !JSD REV A
;        2006      DCT_ADDR [IG_INT] = TRUE;                 ! SET "IGNORE INTERRUPT" BIT
;        2007      L#LUN = .CST_ADDR [OF_UN, D_UNIT];        ! GET FIRST UNIT NUMBER OF CONTROLLER
;        2008                                                ! (USED BY DRS FOR DEVICE-FATAL CTLR ERRORS)
;        2009  IF SWP_DINT NEQ (.DUP_FLAGS AND SWP_DINT)     ! IF DUP CAUSED INIT THEN SKIP THIS SECTION
;        2010  THEN
;        2011      if REG_EXIST () eql FAILURE               ! REGISTER EXISTENCE TEST
;        2012      then
;        2013          begin
;        2014          DROP_CTLR (.CCTLR, DU_INIT);          ! DROP ALL CONTROLLER S UNITS
;        2015          return;
;        2016          end;
;        2017
;        2018  IF SWP_DINT NEQ (.DUP_FLAGS AND SWP_DINT)     ! IF DUP CAUSED INIT THEN SKIP THIS SECTION
;        2019  THEN
;        2020      if VEC_BR_TEST () eql FAILURE             ! VECTOR ADDR AND BR LEVEL TEST
;        2021      then
;        2022          begin
;        2023          DROP_CTLR (.CCTLR, DU_INIT);          ! DROP ALL CONTROLLER'S UNITS
;        2024          return;
;        2025          end;
;        2026
;        2027      RESULT = HARD_INIT ();                    ! ATTEMPT HARD DEVICE INIT
;        2028      DCT_ADDR [IG_INT] = FALSE;                ! CLAER "IGNORE INTERRUPT" BIT
;        2029
;        2030      if .RESULT eql SUCCESS                    ! IF HARD INIT WAS SUCCESSFUL
;        2031      then
;        2032          begin
;        2033          INI_RRING ();                         ! INITIALIZE RESPONSE RING
;        2034          WRT_RDRX (RCSA, RC ALL, SA_GO);       ! SET "GO' BIT (START CTLR POLLING)
;        2035
```

```
;       2036            if SET_CTLR_CHAR () eql SUCCESS         ! SET CONTROLLER CHARACTERISTICS
;       2037            then
;       2038                begin
;       2039                DCT_ADDR [STAT] = ONLINE;           ! MARK CONTROLLER ONLINE IN "DRIVER
;       2040                CST_ADDR [STATE] = ONLINE;          ! MARK CONTROLLER ONLINE IN "PROGRAM
;       2041                end;
;       2042            end
;       2043
;       2044        else                                        ! HARD INIT FAILED
;       2045            begin
;       2046            DROP_CTLR (.CCTLR, DU_INIT);            ! DROP ALL CONTROLLER'S UNITS
;       2047            end;
;       2048
;       2049        end;                                        ! ROUTINE CTLR_INIT

                                        .SBTTL  CTLR.INIT INITIALIZATION TEST ROUTINES
000000  010146                    CTLR.INIT:
                                        MOV     R1,-(SP)                ;                       1980
000002  004737  000000V                 JSR     PC,INI.CTLR.DAT         ;                       2003
000006  012746  000300                  MOV     #300,-(SP)              ;                       2005
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        ;                       2006
000044  013700  000000G                 MOV     CST.ADDR,R0             ;                       2007
000050  016001  000006                  MOV     6(R0),R1
000054  000301                          SWAB    R1
000056  042701  177760                  BIC     #177760,R1
000062  010137  000000G                 MOV     R1,L$LUN
000066  032737  000002  000000G         BIT     #2,DUP.FLAGS            ;                       2009
000074  001025                          BNE     2$
000076  004737  000000V                 JSR     PC,REG.EXIST            ;                       2011
000102  005700                          TST     R0
000104  001410                          BEQ     1$                      ;                       2014
000106  032737  000002  000000G         BIT     #2,DUP.FLAGS            ;                       2018
000114  001015                          BNE     2$
000116  004737  000000V                 JSR     PC,VEC.BR.TEST          ;                       2020
000122  005700                          TST     R0
000124  001011                          BNE     2$
000126  013716  000000G         1$:     MOV     CCTLR,(SP)              ;                       2023
000132  012746  000002                  MOV     #2,-(SP)
000136  004737  000000G                 JSR     PC,DROP.CTLR
000142  062706  000012                  ADD     #12,SP                  ;                       2020
000146  000450                          BR      5$                      ;                       2022
000150  004737  000000V         2$:     JSR     PC,HARD.INIT            ;                       2027
000154  110001                          MOVB    R0,R1                   ; ^,RESULT
000156  042777  040000  000000G         BIC     #40000,@DCT.ADDR        ;                       2028
000164  120127  000001                  CMPB    R1,#1                   ; RESULT,^              2030
000170  001026                          BNE     3$
000172  004737  000000V                 JSR     PC,INI.RRING            ;                       2033
000176  012701  000001                  MOV     #1,R1                   ; ^,RC.REG              2034
000202  013700  000000G                 MOV     RDRX.ADDR,R0
```

```
000206  010160  000002                          MOV     R1,2(R0)                        ; RC.REG,*
000212  004737  000000V                         JSR     PC,SET.CTLR.CHAR                ;                                       2036
000216  020027  000001                          CMP     R0,#1
000222  001020                                  BNE     4$
000224  052777  100000  000000G                 BIS     #100000,@DCT.ADDR               ;                                       2039
000232  013700  000000G                         MOV     CST.ADDR,R0                     ;                                       2040
000236  052760  100000  000002                  BIS     #100000,2(R0)
000244  000407                                  BR      4$                              ;                                       2030
000246  013716  000000G                 3$:     MOV     CCTLR,(SP)                      ;                                       2046
000252  012746  000002                          MOV     #2,-(SP)
000256  004737  000000G                         JSR     PC,DROP.CTLR
000262  005726                                  TST     (SP)+                           ;                                       2045
000264  062706  000010                  4$:     ADD     #10,SP                          ;                                       1998
000270  012601                          5$:     MOV     (SP)+,R1                        ;                                       1980
000272  000207                                  RTS     PC

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

```
;        2050  routine INI_CTLR_DAT : novalue =
;        2051
;        2052  !+
;        2053  !       THIS ROUTINE IS RESPONSIBLE FOR INITIALIZING ALL CONTROLLER RELATED
;        2054  !       DATA IN THE "DRIVER" PORTION OF THE EXERCISER. THIS INCLUDES THE
;        2055  !       CONTROLLER'S DCT AND OUTSTANDING COMMAND LIST.
;        2056  !
;        2057  !       IMPLICIT INPUTS:
;        2058  !               CCTLR   CURRENT CONTROLLER NUMBER
;        2059  !               DCT_ADDR   ADDRESS OF CURENT CONTROLLER'S DCT
;        2060  !-
;        2061
;        2062      begin
;        2063      DCT_ADDR [WORD0] = 0;                                          ! CLEAR FIRST DCT WORD
;        2064      DCT_ADDR [RR_BEG] = COMM_AREA + 8 + (.CCTLR * COMM_LEN * 2);   ! START OF RESPONSE RING
;        2065      DCT_ADDR [RR_END] = .DCT_ADDR [RR_BEG] + ((RRING_LEN - 1) * 4);! LAST SLOT IN RESPONSE RING
;        2066      DCT_ADDR [CR_BEG] = .DCT_ADDR [RR_END] + 4;                    ! START OF COMMAND RING
;        2067      DCT_ADDR [CR_END] = .DCT_ADDR [CR_BEG] + ((CRING_LEN - 1) * 4);! LAST SLOT IN COMMAND RING
;        2068      DCT_ADDR [RR_POLL] = .DCT_ADDR [RR_BEG];                       ! FIRST RRING SLOT TO POLL
;        2069      DCT_ADDR [CR_POLL] = DCT_ADDR [CR_NEXT] = .DCT_ADDR [CR_BEG];  ! CRING POLL AND NEXT COMMAND POINTERS
;        2070      end;

                                        .SBTTL  INI.CTLR.DAT INITIALIZATION TEST ROUTINES
000000  004137  000000G         INI.CTLR.DAT:
                                        JSR     R1,$SAVE2                       ;                       2050
000004  013701  000000G                 MOV     DCT.ADDR,R1                     ;                       2063
000010  005011                          CLR     (R1)
000012  012702  000004                  MOV     #4,R2                           ;                       2064
000016  060102                          ADD     R1,R2
000020  013746  000000G                 MOV     CCTLR,-(SP)
000024  012746  000050                  MOV     #50,-(SP)
000030  004737  000000G                 JSR     PC,BL$MUL
000034  062700  000010'                 ADD     #COMM.AREA+10,R0
000040  010012                          MOV     R0,(R2)
000042  010061  000006                  MOV     R0,6(R1)                        ;                       2065
000046  062761  000014  000006         ADD     #14,6(R1)
000054  012700  000010                  MOV     #10,R0                          ;                       2066
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)                     ;                       2067
000076  062761  000014  000012         ADD     #14,12(R1)
000104  011261  000014                  MOV     (R2),14(R1)                     ;                       2068
000110  011061  000020                  MOV     (R0),20(R1)                     ;                       2069
000114  011061  000016                  MOV     (R0),16(R1)
000120  022626                          CMP     (SP)+,(SP)+                     ;                       2062
000122  000207                          RTS     PC                              ;                       2050

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

```
;          2071  routine REG_EXIST =
;          2072
;          2073  !+
;          2074  !       THIS IS THE REGISTER EXISTENCE (OR "PROBE") TEST DESIGNED TO VERIFY
;          2075  !       THE PRESENCE OF AN RDRX DEVICE. THIS OBJECTIVE IS ACCOMPLISHED BY
;          2076  !       SETTING UP THE NON-EXISTENT MEMORY (NEX) TRAP VECTOR (LOCATION 4) AND
;          2077  !       ATTEMPTING TO READ WHAT IS ASSUMED TO BE THE DEVICE'S SA AND IP
;          2078  !       REGISTERS. IF THE NEX TRAP HANDLER IS INVOKED DUE TO AN ABSENT DEVICE,
;          2079  !       THEN THE GLOBAL DATUM "NEX" WILL BE SET TO "TRUE". THIS DATUM
;          2080  !       DETERMINES THE SUCCESS / FAILURE VALUE OF THIS ROUTINE.
;          2081  !-
;          2082
;          2083      begin
;          2084
;          2085      local
;          2086          TEMP : word,                          ! TEMP FOR READING SA AND IP
;          2087          DUMMY : word;                         ! AS THE NAME IMPLIES
;          2088
;          2089      if .ENTRY_REASON eql NEW_PASS
;          2090      then
;          2091          return SUCCESS;                       ! SKIP TEST FOR NEXT PASS
;          2092
;          2093      OF_RC = 2;                                ! SET UP TO READ SA FIRST
;          2094
;          2095      do
;          2096          begin
;          2097          NEX = FALSE;                          ! SET TO "TRAP NOT RECEIVED"
;          2098          SETVEC (4, NEX_TRAP, PRIO7);          ! SET LOCATION 4 TRAP VECTOR ADDRESS
;          2099          TEMP = .(.RDRX_ADDR + .OF_RC);        ! READ REGISTER (THEN TRAP OR CONTINUE)
;          2100          DUMMY = 0;                            ! DUMMY INSTRUCTION TO COVER TRAP RETURN BUG
;          2101                                                !       (TRAP RETURNS TO NEXT INSTRUCTION)
;          2102          CLRVEC (4);                           ! CLEAR LOCATION 4 TRAP VECTOR ADDRESS
;          2103
;          2104          if .NEX                               ! IF NEX TRAP OCCURRED
;          2105          then
;          2106              begin
;          2107              C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;          2108              ERRDF (10, EGD_10, EMS_10);       ! REGISTER EXISTENCE TEST FAILED
;          2109              SETPRI (PRI00);                   ! LOWER PRIORITY
;          2110              return FAILURE;
;          2111              end
;          2112          else
;          2113              OF_RC = .OF_RC - 2;               ! SET UP FOR IP REG OR QUIT
;          2114
;          2115          end
;          2116      until .OF_RC leq 0;
;          2117
;          2118      return SUCCESS;
;          2119      end;

                                        .SBTTL  REG.EXIST INITIALIZATION TEST ROUTINES
000000  004137  000000G          REG.EXIST:
                                        JSR     R1,$SAVE2                       ;                       2071
000004  123727  000000G 000005          CMPB    ENTRY.REASON,#5                 ;                       2089
000012  001461                          BEQ     3$                              ;                       2091
```

```
000014  012737  000002  000000G          MOV     #2,OF.RC                ;                       2093
000022  005037  000000G          1$:     CLR     NEX                     ;                       2097
000026  012746  000340                   MOV     #340,-(SP)              ;                       2098
000032  012746  000000G                  MOV     #NEX.TRAP,-(SP)
000036  012746  000004                   MOV     #4,-(SP)
000042  012746  000003                   MOV     #3,-(SP)
000046  104437                           TRAP    37
000050  013700  000000G                  MOV     RDRX.ADDR,R0            ;                       2099
000054  063700  000000G                  ADD     OF.RC,R0
000060  011001                           MOV     (R0),R1                 ; *.TEMP
000062  005002                           CLR     R2                      ; DUMMY                 2100
000064  012700  000004                   MOV     #4,R0                   ;                       2102
000070  104436                           TRAP    36
000072  032737  000001  000000G          BIT     #1,NEX                  ;                       2104
000100  001416                           BEQ     2$
000102  013700  000000G                  MOV     CCTLR,R0                ;                       2107
000106  006300                           ASL     R0
000110  105260  000000G                  INCB    C.ERR.TBL(R0)
000114  104455                           TRAP    55                      ;                       2108
000116  000012                           .WORD   12
000120  000000G                          .WORD   EGD.10
000122  000000G                          .WORD   EMS.10
000124  005000                           CLR     R0                      ;                       2109
000126  104441                           TRAP    41
000130  062706  000010                   ADD     #10,SP                  ;                       2104
000134  000413                           BR      4$                      ;                       2106
000136  162737  000002  000000G  2$:     SUB     #2,OF.RC                ;                       2113
000144  062706  000010                   ADD     #10,SP                  ;                       2096
000150  005737  000000G                  TST     OF.RC                   ;                       2116
000154  002322                           BGE     1$
000156  012700  000001          3$:     MOV     #1,R0                   ;                       2083
000162  000207                           RTS     PC
000164  005000                  4$:      CLR     R0                      ;                       2071
000166  000207                           RTS     PC

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

```
;       2120  routine VEC_BR_TEST =
;       2121
;       2122  !+
;       2123  !       THIS ROUTINE ATTEMPTS TO VERIFY (A) THAT THE RDRX VECTOR ADDRESS GIVEN
;       2124  !       BY THE USER DURING THE HW DIALOG IS VALID, AND (B) THAT THE
;       2125  !       USER-SPECIFIED BUS REQUEST LEVEL FOR THE DEVICE IS CORRECT. THE FIRST
;       2126  !       OBJECTIVE IS ACCOMPLISHED BY SETTING THE CPU PRIORITY TO 0 AND FORCING
;       2127  !       AN RDRX INTERRUPT. IF THE USER SPECIFIED AND INCORRECT VECTOR ADDRESS,
;       2128  !       THEN THE RESULT MAY BE UNPREDICTABLE. FOR THIS REASON, THE MESSAGE
;       2129  !       "FUNCTIONAL TEST STARTED" IS PRINTED BEFORE THE TEST, AND
;       2130  !       "EXERCISER STARTED" IS PRINTED AT ITS SUCCESSFUL CONCLUSION. IF
;       2131  !       EITHER "FUNCTIONAL TEST ..." OR "EXERCISER ..." DOES NOT APPEAR, THEN
;       2132  !       PROGRAM CONTROL IS ASSUMED LOST AND A FATAL TRAP IS LIKELY TO OCCUR. AT
;       2133  !       THIS POINT, THE EXERCISER MUST BE STARTED AGAIN.
;       2134  !
;       2135  !       IF THIS TEST SUCCEEDS, THEN THE BR LEVEL TEST IS RUN BY SETTING THE
;       2136  !       PROCESSOR PRIORITY TO THE ASSUMED INTERRUPT PRIORITY GIVEN BY THE
;       2137  !       USER. A FORCED INTERRUPT SHOULD NOT OCCURR. THEN, BY LOWERING THE
;       2138  !       PRIORITY BY ONE, THE DELAYED INTERRUPT SHOULD OCCUR.
;       2139  !-
;       2140
;       2141      begin
;       2142
;       2143      if .ENTRY_REASON eql NEW_PASS
;       2144      then
;       2145          begin
;       2146          SETPRI (PRI00);                         ! LOWER PRIORITY
;       2147          return SUCCESS;                         ! SKIP TEST IF NEXT PASS
;       2148          end;
;       2149
;       2150      PRINTF (MSG_02);                            ! "FUNCTIONAL TEST STARTED"
;       2151
;       2152      if INT_GEN () eql FALSE                     ! FORCE AN INTERRUPT
;       2153      then
;       2154          begin                                   ! IF INTERRUPT DID NOT OCCUR
;       2155          C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;       2156          ERRDF (11, EGD_11, 0);                  ! VECTOR TEST FAILED
;       2157          return FAILURE;
;       2158          end
;       2159      else
;       2160          begin                                   ! INTERRUPT DID OCCUR
;       2161          PRINTF (MSG_03);                        ! "EXERCISER STARTED"
;       2162          SETPRI (.BRLEVEL);                      ! SET PRIORITY TO ASSUMED BR LEVEL
;       2163
;       2164          if INT_GEN () eql FALSE                 ! FORCE AN INTERRUPT (SHOULD NOT OCCUR)
;       2165          then
;       2166              begin                               ! IF INTERRUPT DID NOT OCCUR
;       2167              SETPRI (.BRLEVEL - %o'40');         ! LOWER PRIORITY BY 1
;       2168              DELAY (1);                          ! WAIT
;       2169
;       2170              if .DCT_ADDR [SA_SAVE] neq 0        ! IF INTERRUPT DID OCCUR (SA_SAVE WOULD BE NON-ZERO)
;       2171              then
;       2172                  begin
;       2173                  SETPRI (PRI00);                 ! RESTORE PROCESSOR PRIORITY TO 0
;       2174                  return SUCCESS;                 ! ONLY SUCCESSFUL EXIT POINT
;       2175                  end;
```

```
;         2176
;         2177                end;
;         2178
;         2179            end;
;         2180
;         2181        SETPRI (PRIOO);                          ! COME HERE ONLY FOR BR TEST FAILURE
;         2182        C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C ERR_TBL [.CCTLR, C ERR_HRD] + 1;
;         2183        ERRDF (12, EGD_12, EMS_12);
;         2184        return FAILURE;
;         2185        end;

                                    .GLOBL  L$DLY

                                    .SBTTL  VEC.BR.TEST INITIALIZATION TEST ROUTINES
000000  010146                VEC.BR.TEST:
                                    MOV     R1, (SP)                        ;                       2120
000002  005746                      TST     -(SP)
000004  123727  000000G 000005     CMPB    ENTRY.REASON,#5                 ;                       2143
000012  001003                      BNE     1$
000014  005000                      CLR     R0                              ;                       2146
000016  104441                      TRAP    41
000020  000473                      BR      7$                              ;                       2145
000022  012746  000000G       1$:   MOV     #MSG.02,-(SP)                   ;                       2150
000026  012746  000001              MOV     #1,-(SP)
000032  010600                      MOV     SP,R0                           ; SP,*
000034  104417                      TRAP    17
000036  004737  000000V             JSR     PC,INT.GEN                      ;                       2152
000042  005700                      TST     R0
000044  001012                      BNE     2$
000046  013700  000000G             MOV     CCTLR,R0                        ;                       2155
000052  006300                      ASL     R0
000054  105260  000000G             INCB    C.ERR.TBL(R0)
000060  104455                      TRAP    55                              ;                       2156
000062  000013                      .WORD   13
000064  000000G                     .WORD   EGD.11
000066  000000                      .WORD   0
000070  000466                      BR      9$                              ;                       2152
000072  012716  000000G       2$:   MOV     #MSG.03,(SP)                    ;                       2161
000076  012746  000001              MOV     #1,-(SP)
000102  010600                      MOV     SP,R0                           ; SP,*
000104  104417                      TRAP    17
000106  013700  001236'             MOV     BRLEVEL,R0                      ;                       2162
000112  104441                      TRAP    41
000114  004737  000000V             JSR     PC,INT.GEN                      ;                       2164
000120  005700                      TST     R0
000122  001035                      BNE     8$
000124  013700  001236'             MOV     BRLEVEL,R0                      ;                       2167
000130  162700  000040              SUB     #40,R0
000134  104441                      TRAP    41
000136  012701  000001              MOV     #1,R1                           ; *,$$TMP2              2168
000142  001411                3$:   BEQ     6$
000144  013700  000000G             MOV     L$DLY,R0                        ; *,$$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$:     MOV     DCT.ADDR,R0             ;                       2170
000172  005760  000002                  TST     2(R0)
000176  001407                          BEQ     8$
000200  005000                          CLR     R0                      ;                       2173
000202  104441                          TRAP    41
000204  062706  000006                  ADD     #6,SP                   ;                       2170
000210  012700  000001          7$:     MOV     #1,R0                   ;                       2172
000214  000416                          BR      10$
000216  005726                  8$:     TST     (SP)+                   ;                       2160
000220  005000                          CLR     R0                      ;                       2181
000222  104441                          TRAP    41
000224  013700  000000G                 MOV     CCTLR,R0                ;                       2182
000230  006300                          ASL     R0
000232  105260  000000G                 INCB    C.ERR.TBL(R0)
000236  104455                          TRAP    55                      ;                       2183
000240  000014                          .WORD   14
000242  000000G                         .WORD   EGD.12
000244  000000G                         .WORD   EMS.12
000246  022626                  9$:     CMP     (SP)+,(SP)+             ;                       2120
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
```

```
;          2186  routine INT_GEN =
;          2187
;          2188  !+
;          2189  !       THIS ROUTINE BEGINS AN RDRX INITIALIZATION SEQUENCE, BUT ONLY
;          2190  !       COMPLETES THROUGH THE STEP 1 WRITE. ITS PURPOSE IS TO CREATE AN RDRX
;          2191  !       INTERRUPT (AT THE COMPLETEION OF STEP 1) IN ORDER TO HELP VERIFY THE
;          2192  !       THE USER-SPECIFIED VECTOR ADDRESS AND BUS REQUEST INTERRUPT LEVEL.
;          2193  !       A VALUE OF "TRUE" IS RETURNED TO THE CALLER IF AN INTERRUPT OCCURS,
;          2194  !       AND "FALSE" OTHERWISE. THE INTERRUPT IS VERIFIED BY A NON ZERO VALUE
;          2195  !       IN THE  SA SAVE" WORD IN THE DEVICE'S DCT.
;          2196  !-
;          2197
;          2198      begin
;          2199
;          2200      local
;          2201          SA : word;                              ! STORAGE FOR STEP 1 READ AND WRITE
;          2202
;          2203      DCT_ADDR [SA_SAVE] = 0;                     ! ZERO OUT SA SAVE WORD IN DCT
;          2204      WRT_RDRX (RCIP, RC_ALL, ALL_ONES);          ! WRITE IP TO START INIT SEQUENCE
;          2205      DELAY (10);                                 ! WAIT
;          2206      SA = .RDRX_ADDR [RCSA, RC_ALL];             ! STEP 1 READ
;          2207      SA = (WR_RING ^ 8) or (.CURRENT_VECTOR ^ -2) or SA_INT;     ! STEP 1 WRITE VALUE
;          2208      WRT_RDRX (RCSA, RC_ALL, .SA);               ! STEP 1 WRITE
;          2209
;          2210      incr COUNT from 1 to 1600 do
;          2211          begin
;          2212          DELAY (5);                              ! TOTAL DELAY COUNT OF 8,000
;          2213          BREAK;
;          2214
;          2215          if .DCT_ADDR [SA_SAVE] neq 0            ! IF SA WAS CHANGED
;          2216          then
;          2217              return TRUE;                        ! INTERRUPT OCCURED
;          2218          end;
;          2219
;          2220      return FALSE;                               ! IF INTERRUPT DID NOT OCCUR
;          2221      end;

                                        .SBTTL  INT.GEN INITIALIZATION TEST ROUTINES
000000  004137  000000G       INT.GEN: JSR     R1,$SAVE2                      ;                       2186
000004  024646                         CMP     -(SP),-(SP)
000006  013700  000000G                MOV     DCT.ADDR,R0                    ;                       2203
000012  005060  000002                 CLR     2(R0)
000016  012700  177777                 MOV     #-1,R0                         ; *,RC.REG              2204
000022  010077  000000G                MOV     R0,@RDRX.ADDR                  ; RC.REG,*
000026  012701  000012                 MOV     #12,R1                         ; *,$$TMP2              2205
000032  001411                 1$:     BEQ     4$
000034  013700  000000G                MOV     L$DLY,R0                       ; *,$$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                   ;                       2206
000062  016016  000002                 MOV     2(R0),(SP)                     ; *,RC.REG
```

```
000066  013701  001234'                        MOV     CURRENT.VECTOR,R1               ;                                       2207
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                             2208
000106  010160  000002                          MOV     R1,2(R0)                        ; RC.REG,*
000112  012702  003100                          MOV     #3100,R2                        ; *,COUNT                               2210
000116  012701  000005          5$:     MOV     #5,R1                           ; *,$$TMP2                              2212
000122  001411                  6$:     BEQ     9$
000124  013700  000000G                         MOV     L$DLY,R0                        ; *,$$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$:     TRAP    22
000150  013700  000000C                         MOV     DCT.ADDR,R0                     ;                                       2215
000154  005760  000002                          TST     2(R0)
000160  001403                          BEQ     10$
000162  012700  000001                          MOV     #1,R0                           ;                                       2217
000166  000403                          BR      11$
000170  005302                  10$:    DEC     R2                              ; COUNT                                 2210
000172  001351                          BNE     5$
000174  005000                          CLR     R0                              ;                                       2198
000176  022626                  11$:    CMP     (SP)+,(SP)+                     ;                                       2186
000200  000207                          RTS     PC

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

```
;          2222  routine HARD_INIT =
;          2223
;          2224  !+
;          2225  !       THIS ROUTINE PERFORMS THE FOUR READ / WRITE STEPS REQUIRED TO
;          2226  !       INITIALIZE AN RDRX DEVICE. IF NO READ ERRORS ARE DETECTED IN ANY OF
;          2227  !       THE FOUR STEPS, THEN A SUCCESS VALUE IS RETURNED TO THE CALLER.
;          2228  !       OTHERWISE, ADDITIONAL ATTEMPTS MAY BE MADE TO INITIALIZE THE DEVICE.
;          2229  !       IF ALL ATTEMPTS FAIL, A FAILURE INDICATION IS RETURNED.
;          2230  !-
;          2231
;          2232      begin
;          2233
;          2234      local
;          2235          IE_VEC : word;                              ! IE-BIT-AND VECTOR ADDRESS/4 BYTE
;          2236                                                      !   (USED IN STEP 1 WRITE AND STEP 3 READ)
;          2237
;          2238      IE_VEC = .CURRENT_VECTOR + -2;                  ! GET VECTOR ADDR/4 (IE = 0)
;          2239
;          2240      incr ATTEMPTS from 1 to INI_ATT do
;          2241          begin
;          2242
;          2243          label
;          2244              STEP_1_READ,
;          2245              STEP_2_READ,
;          2246              STEP_3_READ,
;          2247              STEP_4_READ;
;          2248
;          2249          WRT_RDRX (RCIP, RC_ALL, ALL_ONES);          ! WRITE IP TO START INIT SEQUENCE
;          2250  !
;          2251  !       STEP 1 READ
;          2252  !
;          2253          STEP = 1;
;          2254          STEP_1_READ:
;          2255              begin
;          2256
;          2257              incr COUNT from 1 to 100 do
;          2258                  begin
;          2259                  DELAY (5);                          ! TOTAL DELAY COUNT OF 500 FOR STEP 1
;          2260                  BREAK;
;          2261                  SA_REG = .RDRX_ADDR [RCSA, RC_ALL];     ! READ SA
;          2262
;          2263                  if (.SA_REG and S1_MASK) eql SA_S1      ! IF STEP 1 READ IS O.K.
;          2264                  then
;          2265                      leave STEP_1_READ;
;          2266
;          2267                  end;
;          2268
;          2269              exitloop;
;          2270              end;
;          2271
;          2272  !
;          2273  !       STEP 1 WRITE
;          2274  !
;          2275          SA_REG = (WR_RING + 8) or .IE_VEC;          ! STEP 1 WRITE VALUE
;          2276          WRT_RDRX (RCSA, RC ALL, .SA_REG);           ! STEP 1 WRITE
;          2277  !
```

```
;          2278  !       STEP 2 READ
;          2279  !
;          2280          STEP = .STEP + 1;
;          2281          STEP_2_READ:
;          2282              begin
;          2283
;          2284               incr COUNT from 1 to 2000 do
;          2285                  begin
;          2286                  DELAY (5);                                  ! TOTAL DELAY COUNT OF 10,000 FOR STEP 2
;          2287                  BREAK;
;          2288                  SA_REG = .RDRX_ADDR [RCSA, RC ALL];         ! READ SA
;          2289
;          2290                  if (.SA_REG and S2_MASK) eql (SA S2 or WR_RING) ! IF STEP 2 READ IS O.K.
;          2291                  then
;          2292                      leave STEP_2 READ;
;          2293
;          2294                  end;
;          2295
;          2296              exitloop;
;          2297              end;
;          2298
;          2299  !
;          2300  !       STEP 2 WRITE
;          2301  !
;          2302          WRT_RDRX (RCSA, RC_ALL, .DCT_ADDR [RR_BEG]);    ! RINGBASE-LO, PI = 0
;          2303  !
;          2304  !       STEP 3 READ
;          2305  !
;          2306          STEP = .STEP + 1;
;          2307          STEP_3_READ:
;          2308              begin
;          2309
;          2310              incr COUNT from 1 to 2000 do
;          2311                  begin
;          2312                  DELAY (5);                                  ! TOTAL DELAY COUNT OF 10,000 FOR STEP 3 READ
;          2313                  BREAK;
;          2314                  SA_REG = .RDRX_ADDR [RCSA, RC_ALL];         ! READ SA
;          2315
;          2316                  if (.SA_REG and S3_MASK) eql (SA_S3 or .IE_VEC) ! IF STEP 3 READ IS O.K.
;          2317                  then
;          2318                      leave STEP_3_READ;
;          2319
;          2320                  end;
;          2321
;          2322              exitloop;
;          2323              end;
;          2324
;          2325  !
;          2326  !       STEP 3 WRITE
;          2327  !
;          2328          WRT_RDRX (RCSA, RC_ALL, 0);                     ! PP, RINGBASE-HI = 0
;          2329  !
;          2330  !       STEP 4 READ
;          2331  !
;          2332          STEP = .STEP + 1;
;          2333          STEP_4_READ:
```

```
;                       begin
;
;                       incr COUNT from 1 to 2000 do
;                           begin
;                           DELAY (5);                              ! TOTAL DELAY COUNT OF 10,000 FOR STEP 4 READ
;                           BREAK;
;                           SA_REG = .RDRX ADDR [RCSA, RC ALL];     ! READ SA
;
;                           if (.SA_REG and S4 MASK  eql SA S4      ! IF STEP 4 READ IS O.K.
;                           then
;                               leave STEP 4 READ;
;
;                           end;
;
;                       exitloop;
;                       end;
;           !
;           !       STEP 4 WRITE
;           !
;                   CREDIT_BAL = 1;                                 ! START WITH A CREDIT BALANCE = 1
;                   WRT_RDRX (RCSA, RC ALL, 0);                     ! BURST, LF, GO = 0
;                   return SUCCESS;                                 ! SUCCESS EXIT POINT
;
;                   end;                                            ! TRY AGAIN OR GIVE UP
;
;               CREDIT_BAL = 0;                         ! NO CREDIT BALANCE
;               C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR TBL [.CCTLR, C_ERR_HRD] + 1;
;               ERRDF (13, EGD_13, EMS_13);             ! INIT SEQUENCE FAILED
;               return FAILURE;
;               end;                                    ! ROUTINE HARD_INIT


                                        .SBTTL  HARD.INIT INITIALIZATION TEST ROUTINES
000000  004137  000000G                 HARD.INIT:
                                              JSR     R1,$SAVE5                 ;                       2222
000004  162706  000012                        SUB     #12,SP
000010  013704  001234'                       MOV     CURRENT.VECTOR,R4         ; *,IE.VEC              2238
000014  006204                                ASR     R4                        ; IE.VEC
000016  006204                                ASR     R4                        ; IE.VEC
000020  012705  000002                        MOV     #2,R5                     ; *,ATTEMPTS            2240
000024  012700  177777                        MOV     #-1,R0                    ; *,RC.REG              2249
000030  010077  000000G                       MOV     R0,@RDRX.ADDR             ; RC.REG,*
000034  012737  000001  000000G               MOV     #1,STEP                   ;                       2253
000042  012702  000144                        MOV     #144,R2                   ; *,COUNT               2257
000046  012701  000005          1$:           MOV     #5,R1                     ; *,$$TMP2              2259
000052  001411                  2$:           BEQ     5$
000054  013700  000000G                       MOV     L$DLY,R0                  ; *,$$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
000100  013700  000000G                       MOV     RDRX.ADDR,R0              ;                       2261
000104  016066  000002  000006                MOV     2(R0),6(SP)               ; *,RC.REG
```

```
000116  016637  000006  000000G          MOV     6(SP),SA.REG            ; RC.REG,*
000120  016600  000006                   MOV     6(SP),R0                ; SA.REG,*                      2263
000124  042700  001777                   BIC     #1777,R0
000130  020027  004000                   CMP     R0,#4000
000134  001403                           BEQ     6$                      ;                               2265
000136  005302                           DEC     R2                      ; COUNT                         2267
000140  001342                           BNE     1$
000142  000576                           BR      24$                     ;                               2241
000144  010437  000000G          6$:     MOV     R4,SA.REG               ; IE.VEC,*                      2275
000150  052737  111000  000000G          BIS     #111000,SA.REG
000156  013701  000000G                   MOV     SA.REG,R1               ; *,RC.REG                      2276
000162  013700  000000G                   MOV     RDRX.ADDR,R0
000166  010160  000002                   MOV     R1,2(R0)                ; RC.REG,*
000172  005237  000000G                   INC     STEP                    ;                               2280
000176  012702  003720                   MOV     #3720,R2                ; *,COUNT                       2284
000202  012701  000005          7$:     MOV     #5,R1                   ; *,$$TMP2                      2286
000206  001411                  8$:     BEQ     11$
000210  013700  000000G                   MOV     L$DLY,R0                ; *,$$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                      ; $$TMP2
000230  000766                           BR      8$
000232  104422                  11$:     TRAP    22
000234  013700  000000G                   MOV     RDRX.ADDR,R0            ;                               2288
000240  016066  000002  000004           MOV     2(R0),4(SP)             ; *,RC.REG
000246  016637  000004  000000G          MOV     4(SP),SA.REG            ; RC.REG,*
000254  016600  000004                   MOV     4(SP),R0                ; SA.REG,*                      2290
000260  042700  003400                   BIC     #3400,R0
000264  020027  010222                   CMP     R0,#10222
000270  001403                           BEQ     12$                     ;                               2292
000272  005302                           DEC     R2                      ; COUNT                         2284
000274  001342                           BNE     7$
000276  000534                           BR      26$                     ;                               2241
000300  013700  000000G          12$:    MOV     DCT.ADDR,R0             ;                               2302
000304  016001  000004                   MOV     4(R0),R1                ; *,RC.REG
000310  013700  000000G                   MOV     RDRX.ADDR,R0
000314  010160  000002                   MOV     R1,2(R0)                ; RC.REG,*
000320  005237  000000G                   INC     STEP                    ;                               2306
000324  010403                           MOV     R4,R3                   ; IE.VEC,*                      2316
000326  052703  020000                   BIS     #20000,R3
000332  012702  003720                   MOV     #3720,R2                ; *,COUNT                       2310
000336  012701  000005          13$:    MOV     #5,R1                   ; *,$$TMP2                      2312
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
000370  013700  000000G                   MOV     RDRX.ADDR,R0            ;                               2314
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,*                      2316
000414  042700  003400                  BIC     #3400,R0
000420  020003                          CMP     R0,R3
000422  001403                          BEQ     18$                     ;                               2318
000424  005302                          DEC     R2                      ; COUNT                         2310
000426  001343                          BNE     13$
000430  000457                          BR      26$                     ;                               2241
000432  013700  000000G       18$:     MOV     RDRX.ADDR,R0            ;                               2328
000436  005060  000002                  CLR     2(R0)
000442  005237  000000G                 INC     STEP                    ;                               2332
000446  012703  003720                  MOV     #3720,R3                ; *.COUNT                       2336
000452  012701  000005        19$:     MOV     #5,R1                   ; *.$$TMP2                      2338
000456  001411                20$:     BEQ     23$
000460  013700  000000G                 MOV     L$DLY,R0                ; *.$$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$:     TRAP    22
000504  013700  000000G               MOV     RDRX.ADDR,R0            ;                               2340
000510  016016  000002                  MOV     2(R0),(SP)              ; *,RC.REG
000514  011637  000000G                 MOV     (SP),SA.REG             ; RC.REG,*
000520  011600                          MOV     (SP),R0                 ; SA.REG,*                      2342
000522  042700  003777                  BIC     #3777,R0
000526  020027  040000                  CMP     R0,#40000
000532  001403                          BEQ     25$                     ;                               2344
000534  005303                          DEC     R3                      ; COUNT                         2336
000536  001345                          BNE     19$
000540  000413                24$:     BR      26$                     ;                               2241
000542  012737  000001  000000G 25$:     MOV     #1,CREDIT.BAL           ;                               2353
000550  005001                          CLR     R1                      ; RC.REG                        2354
000552  013700  000000G                 MOV     RDRx.ADDR,R0
000556  005060  000002                  CLR     2(R0)
000562  012700  000001                  MOV     #1,R0                   ;                               2241
000566  000414                          BR      27$
000570  005037  000000G       26$:     CLR     CREDIT.BAL              ;                               2359
000574  013700  000000G                 MOV     CCTLR,R0                ;                               2360
000600  006300                          ASL     R0
000602  105260  000000G                 INCB    C.ERR.TBL(R0)
000606  104455                          TRAP    55                      ;                               2361
000610  000015                          .WORD   15
000612  000000G                         .WORD   EGD.13
000614  000000G                         .WORD   EMS.13
000616  005000                          CLR     R0                      ;                               2232
000620  062706  000012        27$:     ADD     #12,SP                  ;                               2222
000624  000207                          RTS     PC

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

```
;          2364  routine INI_RRING : novalue =
;          2365
;          2366  !+
;          2367  !       THIS ROUTINE IS RESPONSIBLE FOR ALLOCATING ENOUGH MSCP PACKETS TO
;          2368  !       FILL AN RDRX RESPONSE RING. THE BUFFER DESCRIPTOR OF EACH PACKET
;          2369  !       (LOCATED IN FRONT OF THE PACKET ITSELF) IS LOADED INTO SUCCESSIVE
;          2370  !       RRING SLOTS. NOTE THAT THE BUFFER DESCRIPTORS HAVE BEEN INITIALIZED
;          2371  !       WITH THE FLAG AND OWNERSHIP BITS SET TO "1", MAKING EACH SLOT
;          2372  !       CONTROLLER-OWNED.
;          2373  !
;          2374  !       IMPLICIT INPUTS:
;          2375  !               CCTLR - CURRENT CONTROLLER NUMBER
;          2376  !               DCT_ADDR - ADDRESS OF CURRENT CONTROLLER'S DCT
;          2377  !-
;          2378
;          2379      begin
;          2380
;          2381      local
;          2382          index : word,
;          2383          RRING_ADDR;
;          2384
;          2385      RRING_ADDR = .DCT_ADDR [RR_BEG];                    ! FIRST RESPONSE RING SLOT
;          2386
;          2387      incr COUNT from 1 to RRING_LEN do
;          2388          begin
;          2389          index = GET_PKT (.CCTLR);                       ! GET AN MSCP PACKET
;          2390          .RRING_ADDR = .MSCP_PKT [.index, PKT_LO];       ! LOAD LO-ORDER BUFF DESC INTO SLOT
;          2391          RRING_ADDR = .RRING_ADDR + 2;                   ! ADVANCE TO SECOND WORD
;          2392          .RRING_ADDR = .MSCP_PKT [.index, PKT_HI];       ! LOAD HI-ORDER BUFF DESC INTO SLOT
;          2393          PKT_USE [.index] = .CCTLR;                      ! PACKET IN USE
;          2394          .RRING_ADDR = ..RRING_ADDR or ED_OWN or ED_FLAG;! GIVE OWNERSHIP TO CONTRLLER
;          2395          RRING_ADDR = .RRING_ADDR + 2;                   ! ADVANCE TO NEXT SLOT
;          2396          end;
;          2397
;          2398      end;
```

```
                                        .SBTTL  INI.RRING INITIALIZATION TEST ROUTINES
000000  004137  000000G           INI.RRING:
                                        JSR     R1,$SAVE4                       ;                                       2364
000004  013700  000000G                 MOV     DCT.ADDR,R0                     ;                                       2385
000010  016004  000004                  MOV     4(R0),R4                        ; *,RRING.ADDR
000014  013703  000000G                 MOV     CCTLR,R3                        ;                                       2389
000020  012702  000004                  MOV     #4,R2                           ; *,COUNT                               2387
000024  010346                  1$:     MOV     R3,-(SP)                        ;                                       2389
000026  004737  000000G                 JSR     PC,GET.PKT
000032  010001                          MOV     R0,R1                           ; *,INDEX
000034  010116                          MOV     R1,(SP)                         ; INDEX,*                               2390
000036  012746  000104                  MOV     #104,-(SP)
000042  004737  000000G                 JSR     PC,BL$MUL
000046  016024  000000G                 MOV     MSCP.PKT(R0),(R4)+              ; *,RRING.ADDR
000052  016014  000002G                 MOV     MSCP.PKT+2(R0),(R4)             ; *,RRING.ADDR                          2392
000056  013703  000000G                 MOV     CCTLR,R3                        ;                                       2393
000062  110361  000000G                 MOVB    R3,PKT.USE(R1)                  ; *,*(INDEX)
000066  052724  140000                  BIS     #140000,(R4)+                   ; *,RRING.ADDR                          2394
000072  022626                          CMP     (SP)+,(SP)+                     ;                                       2388
000074  005302                          DEC     R2                              ; COUNT                                 2387
000076  001352                          BNE     1$
000100  000207                          RTS     PC                              ;                                       2364

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

```
2399  routine SET_CTLR_CHAR =
2400
2401  !+
2402  !       THIS ROUTINE IS CALLED BY CTLR_INIT AFTER THE RDRX HAS BEEN HARD
2403  !       INITIALIZED. ITS PURPOSE IS TO FORMAT AND SEND THE "SET CONTROLLER
2404  !       CHARACTERISTICS" COMMAND, AND TO VALIDATE THE RESPONSE (END MESSAGE).
2405  !
2406  !       IMPLICIT INPUTS:
2407  !               CCTLR   CURRENT CONTROLLER NUMBER
2408  !
2409
2410      begin
2411
2412  !+
2413  !       MISCELLANEOUS INITIALIZATON
2414  !
2415
2416  QIO [.CCTLR] = 0;                               ! INITIALIZE NO. OF OUTSTANDING QIOS
2417
2418  incr COUNT from 0 to (RP_CNT - 1) do            ! INITIALIZE RETURN PACKET POOL
2419      RP_USE [.COUNT] = -1;
2420
2421  IODQ_IN = IODQ_OUT = 0;                         ! INITIALIZE I/O DONE QUEUE POINTERS
2422
2423      P_INDEX = GET_PKT (.CCTLR);                 ! GET AN MSCP PACKET
2424      MSCP_PKT [.P_INDEX, MSGLEN] = SZ_SCC;       ! PACKET SIZE
2425      MSCP_PKT [.P_INDEX, OPCODE] = OP_SCC;       ! OPCODE = SET CTLR CHAR
2426      MSCP_PKT [.P_INDEX, C_FLAGS] = CF_MASK;     ! CONTROLLER FLAGS
2427      MSCP_PKT [.P_INDEX, CMD_TYPE] = IMM_CMD;    ! IMMEDIATE COMMAND
2428
2429      if SEND (.P_INDEX) eql FAILURE              ! ATTEMPT SEND
2430      then
2431          begin                                   ! IF SEND WAS UNSUCCESSFUL
2432          C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
2433          ERRDF (20, EGD_20, 0);                  ! FATAL ERROR
2434          PUT_PKT (.P_INDEX);                     ! RETURN PACKET TO POOL
2435          DROP_CTLR (.CCTLR, DU_CFATAL);          ! DROP CONTROLLER
2436          return FAILURE;
2437          end
2438      else
2439          begin                                   ! IF SEND WAS SUCCESSFUL
2440
2441          do
2442              begin
2443              WAIT ();                            ! WAIT FOR RETPKT RESPONSE
2444              RP_INDX = OUT_IODQ ();              ! GET INDEX OF RETPKT
2445              RP_ADDR = RETPKT + (.RP_INDX * RP_LEN * 2); ! CALCULATE RETPKT ADDRESS
2446
2447              if .RP_ADDR [MESTYP] neq MT_SEQ     ! RETURN ALL RETPKTS NOT SENT BY CONTROLLER
2448              then
2449                  PUT_RETPKT (.RP_INDX);
2450
2451              end
2452          until (.RP_ADDR [CONID] eql CID_DRIVER) or
2453                ((.RP_ADDR [MESTYP] eql MT_SEQ) and
2454                 ((.RP_ADDR [ENDCOD] and OP_END) eql OP_END));
```

```
;          2455
;          2456          if .RP_ADDR [CONID] eql CID_DRIVER      ! IF RETPKT IS FROM "DRIVER"
;          2457          then
;          2458              begin
;          2459  !           PRINTF (DBM23);                     ! "ERROR IN SET_CTLR_CHAR"      !JSD REV A
;          2460              PUT_RETPKT (.RP_INDX);              ! RELEASE RETURN PACKET
;          2461              DR_ERR ();                          ! DROP CONTROLLER
;          2462              return FAILURE;
;          2463              end
;          2464          else
;          2465              begin                               ! ELSE   RETPKT IS FROM DISK MSCP
;          2466
;          2467              if (.RP_ADDR [ENDCOD] neq (OP_SCC or OP_END)) or    ! IF WRONG ENDCODE
;          2468                 ((.RP_ADDR [C_FLGS] and CF_MASK) neq CF_MASK)    ! OR FLAGS IN ERROR
;          2469              then
;          2470                  begin
;          2471                  C_ERR_TBL [.CCTLR, C_ERR_HRD] = .C_ERR_TBL [.CCTLR, C_ERR_HRD] + 1;
;          2472                  ERRDF (21, EGD_21, EMS_21);     ! FATAL ERROR
;          2473                  DROP_CTLR (.CCTLR, DU_CFATAL);  ! DROP CONTROLLER
;          2474                  PUT_RETPKT (.RP_INDX);          ! RELEASE RETURN PACKET
;          2475                  return FAILURE;
;          2476                  end
;          2477              else
;          2478                  begin                           ! RETPKT HAS CORRECT ENDCODE
;          2479                  CMD_TIME = .RP_ADDR [C_TIME] * 2;
;          2480
;          2481  !               if (.SWP_FLAGS and SWF_TRC) eql SWF_TRC         !JSD REV A
;          2482  !               then                                            !JSD REV A
;          2483  !                   PRINTF (DBM25, .RP_ADDR [C_TIME]);          !JSD REV A
;          2484
;          2485                  end;                            ! RETPKT HAS CORRECT ENDCODE
;          2486
;          2487              end;                                ! IF RETPKT WAS SENT BY DISK MSCP
;          2488
;          2489          PUT_RETPKT (.RP_INDX);
;          2490          return SUCCESS;                         ! IF SEND WAS SUCCESSFUL
;          2491          end;
;          2492
;          2493      end;                                        ! ROUTINE SET-CTLR_CHAR


                                    .SBTTL  SET.CTLR.CHAR INITIALIZATION TEST ROUTINES
000000  010146                SET.CTLR.CHAR:
                                    MOV     R1,-(SP)                        ;                       2399
000002  013701  000000G             MOV     CCTLR,R1                        ;                       2416
000006  105061  000000G             CLRB    QIO(R1)
000012  005000                      CLR     R0                              ; COUNT                 2418
000014  112760  000377  000000G 1$: MOVB    #377,RP.USE(R0)                 ; #.#(COUNT)            2419
000022  005200                      INC     R0                              ; COUNT                 2418
000024  020027  000003              CMP     R0,#3                           ; COUNT,#
000030  003771                      BLE     1$
000032  005037  000000G             CLR     IODQ.OUT                        ;                       2421
000036  005037  000000G             CLR     IODQ.IN
000042  010146                      MOV     R1,-(SP)                        ;                       2423
000044  004737  000000G             JSR     PC,GET.PKT
000050  010037  000000G             MOV     R0,P.INDEX
```

```
000054  010016                          MOV     R0,(SP)                 ; P.INDEX,*                     2424
000056  012746  000104                  MOV     #104, (SP)
000062  004737  000000G                 JSR     PC,BL$MUL
000066  012760  000040  000006G         MOV     #40,MSCP.PKT+6(R0)
000074  112760  000004  000022G         MOVB    #4,MSCP.PKT+22(R0)      ;                       2425
000102  012760  000120  000030G         MOV     #120,MSCP.PKT+30(R0)    ;                       2426
000110  005060  000004G                 CLR     MSCP.PKT+4(R0)          ;                       2427
000114  013716  000000G                 MOV     P.INDEX,(SP)            ;                       2429
000120  004737  000000G                 JSR     PC,SEND
000124  005700                          TST     R0
000126  001026                          BNE     2$
000130  013700  000000G                 MOV     CCTLR,R0                ;                       2432
000134  006300                          ASL     R0
000136  105260  000000G                 INCB    C.ERR.TBL(R0)
000142  104455                          TRAP    55                      ;                       2433
000144  000024                          .WORD   24
000146  000000G                          WORD   EGD.20
000150  000000                          .WORD   0
000152  013716  000000G                 MOV     P.INDEX,(SP)            ;                       2434
000156  004737  000000G                 JSR     PC,PUT.PKT
000162  013716  000000G                 MOV     CCTLR,(SP)              ;                       2435
000166  012746  000006                  MOV     #6,-(SP)
000172  004737  000000G                 JSR     PC,DROP.CTLR
000176  005726                          TST     (SP)+                   ;                       2431
000200  005000                          CLR     R0                      ;                       2410
000202  000536                          BR      9$
000204  004737  000000G         2$:     JSR     PC,WAIT                 ;                       2443
000210  004737  000000G                 JSR     PC,OUT.IODQ             ;                       2444
000214  010037  000000G                 MOV     R0,RP.INDX
000220  010016                          MOV     R0,(SP)                 ; RP.INDX,*             2445
000222  012746  000060                  MOV     #60,-(SP)
000226  004737  000000G                 JSR     PC,BL$MUL
000232  062700  000000G                 ADD     #RETPKT,R0
000236  010037  000000G                 MOV     R0,RP.ADDR
000242  132760  000360  000002         BITB    #360,2(R0)              ;                       2447
000250  001404                          BEQ     3$
000252  013716  000000G                 MOV     RP.INDX,(SP)            ;                       2449
000256  004737  000000G                 JSR     PC,PUT.RETPKT
000262  005726                  3$:     TST     (SP)+                   ;                       2442
000264  013701  000000G                 MOV     RP.ADDR,R1              ;                       2452
000270  005000                          CLR     R0
000272  126127  000003  000003         CMPB    3(R1),#3
000300  001002                          BNE     4$
000302  005200                          INC     R0
000304  000407                          BR      5$
000306  132761  000360  000002  4$:     BITB    #360,2(R1)              ;                       2453
000314  001333                          BNE     2$
000316  105761  000014                  TSTB    14(R1)                  ;                       2454
000322  100330                          BPL     2$
000324  006000                  5$:     ROR     R0                      ;                       2456
000326  103010                          BCC     6$
000330  013716  000000G                 MOV     RP.INDX,(SP)            ;                       2460
000334  004737  000000G                 JSR     PC,PUT.RETPKT
000340  004737  000000V                 JSR     PC,DR.ERR               ;                       2461
000344  022626                          CMP     (SP)+,(SP)+             ;                       2456
000346  000456                          BR      10$                     ;                       2458
```

```
000350  126127  000014  000204          6$:     CMPB    14(R1),#0204                    ;                       2467
000356  001007                                  BNE     7$
000360  016100  000022                          MOV     22(R1),R0                       ;                       2468
000364  042700  177657                          BIC     #177657,R0
000370  020027  000120                          CMP     R0,#0120
000374  001426                                  BEQ     8$
000376  013700  000000G               7$:     MOV     CCTLR,R0                        ;                       2471
000402  006300                                  ASL     R0
000404  105260  000000G                         INCB    C.ERR.TBL(R0)
000410  104455                                  .TRAP   55                              ;                       2472
000412  000025                                  .WORD   25
000414  000000G                                 .WORD   EGD.21
000416  000000G                                 .WORD   EMS.21
000420  013716  000000G                         MOV     CCTLR,(SP)                      ;                       2473
000424  012746  000006                          MOV     #6, (SP)
000430  004737  000000G                         JSR     PC,DROP.CTLR
000434  013716  000000G                         MOV     RP.INDX,(SP)                    ;                       2474
000440  004737  000000G                         JSR     PC,PUT.RETPKT
000444  062706  000006                          ADD     #6,SP                           ;                       2467
000450  000415                                  BR      10$                             ;                       2470
000452  016137  000024  000000G       8$:     MOV     24(R1),CMD.TIME                 ;                       2479
000460  006337  000000G                         ASL     CMD.TIME
000464  013716  000000G                         MOV     RP.INDX,(SP)                    ;                       2489
000470  004737  000000G                         JSR     PC,PUT.RETPKT
000474  012700  000001                          MOV     #1,R0                           ;                       2410
000500  022626                        9$:     CMP     (SP)+,(SP)+                     ;                       2399
000502  000401                                  BR      11$                             ;                       2410
000504  005000                       10$:     CLR     R0                              ;                       2399
000506  012601                       11$:     MOV     (SP)+,R1
000510  000207                                RTS     PC

; Routine Size:  165 words,    Routine Base:  $CODE$ + 3056
; Maximum stack depth per invocation:  5 words
```

```
;          2494  routine UNIT_INIT : novalue =
;          2495
;          2496  !+
;          2497  !       THIS ROUTINE IS CALLED FROM DRIVER_INIT FOR EACH CONFIGURED UNIT
;          2498  !       (DISK) WHICH IS ATTACHED TO A CONTROLLER THAT SURVIVED
;          2499  !       INITIALIZATION. ITS PURPOSE IS TO FORMAT AND SEND AN "ONLINE"
;          2500  !       MESSAGE, AND TO VERIFY THE RESPONSE.
;          2501  !
;          2502  !       IMPLICIT INPUTS:
;          2503  !               CCTLR   CURRENT CONTROLLER NUMBER
;          2504  !               CDISK   CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;          2505  !               L$LUN   CURRENT (DRS) UNIT NUMBER
;          2506  !               CST ADDR   ADDRESS OF CURRENT CONTROLLER'S CST
;          2507  !
;          2508
;          2509      begin
;          2510
;          2511      P_INDEX = GET_PKT (.CCTLR);                 ! GET AN MSCP PACKET
;          2512      MSCP_PKT [.P_INDEX, MSGLEN] = SZ_ONL;       ! PACKET SIZE
;          2513      MSCP_PKT [.P_INDEX, DK_NUM] = .CDISK;       ! SET DISK ADDRESS (RD/RX DISK NUMBER)
;          2514      MSCP_PKT [.P_INDEX, OPCODE] = OP_ONL;       ! OPCODE FOR "ONLINE"
;          2515      MSCP_PKT [.P_INDEX, DDPAR] = BIT00;         ! SHOW ALL ECC ERRORS IN ERROR LOG MESSAGES
;          2516      MSCP_PKT [.P_INDEX, CMD_TYPE] = SEQ_CMD;    ! SEQUENTIAL COMMAND
;          2517
;          2518      if SEND (.P_INDEX) eql FAILURE              ! ATTEMPT TO SEND; IF CTLR IS OFFLINE
;          2519      then
;          2520          begin
;          2521          T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;          2522          CST_ADDR [.CUOFF, D_FATAL] = TRUE;      ! FATAL ERROR
;          2523          ERRDF (22, EGD_22, 0);                  !
;          2524          DUR [.L$LUN] = DU_ONLINE;               ! SETUP REASON TO DROP UNIT
;          2525          DODU (.L$LUN);                          ! DROP UNIT
;          2526          PUT_PKT (.P_INDEX);                     ! RETURN PACKET TO POOL
;          2527          end
;          2528      else
;          2529          begin                                   ! OTHERWISE (SEND WAS SUCCESSFUL)
;          2530
;          2531          do
;          2532              begin
;          2533              WAIT ();                            ! WAIT FOR RETPKT RESPONSE
;          2534              RP_INDX = OUT_IODQ ();              ! GET INDEX OF RETPKT
;          2535              RP_ADDR = RETPKT + (.RP_INDX * RP_LEN * 2); ! CALCULATE RETPKT ADDRESS
;          2536
;          2537              if .RP_ADDR [MESTYP] neq MT_SEQ     ! RETURN ALL RETPKTS NOT SENT BY CONTROLLER
;          2538              then
;          2539                  PUT_RETPKT (.RP_INDX);
;          2540
;          2541              end
;          2542          until (.RP_ADDR [CONID] eql CID_DRIVER) or
;          2543                ((.RP_ADDR [MESTYP] eql MT_SEQ) and
;          2544                 ((.RP_ADDR [ENDCOD] and OP_END) eql OP_END));
;          2545
;          2546          if .RP_ADDR [CONID] eql CID_DRIVER      ! IF RETPKT IS FROM "DRIVER'
;          2547          then
;          2548              begin
;          2549  !           PRINTF (DBM26);                     ! "ERROR IN UNIT_INIT"  !JSD REV A
```

```
;          2550                DR_ERR ();                              ! DROP CONTROLLER
;          2551                end
;          2552            else
;          2553
;          2554                if .RP_ADDR [ENDCOD] neq (OP_ONL or OP_END) ! IF RETPKT IS FROM DISK MSCP
;          2555                then
;          2556                    begin
;          2557                    T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;          2558                    CST_ADDR [.CUOFF, D_FATAL] = TRUE;
;          2559                    ERRDF (23, EGD_23, EMS_21);         ! FATAL ERROR
;          2560                    DUR [.L$LUN] = DU_ONLINE;           ! SETUP REASON TO DROP UNIT
;          2561                    DODU (.L$LUN);                      ! DROP UNIT
;          2562                    end
;          2563                else
;          2564                    begin                               ! RETPKT HAS GOOD ENDCODE
;          2565                    ST_CODE = .RP_ADDR [STSCOD];        ! GET STATUS CODE
;          2566                    SB_CODE = .RP_ADDR [SUBCOD];        ! GET SUB-CODE
;          2567
;          2568                    if .ST_CODE neq ST_SUC              ! IF STATUS CODE IS NOT SUCCESSFUL
;          2569                    then
;          2570                        begin
;          2571                        T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;          2572                        CST_ADDR [.CUOFF, D_FATAL] = TRUE;
;          2573                        ERRDF (15, EGD_15, EMS_30); ! ONLINE FAILED
;          2574                        DUR [.L$LUN] = DU_ONLINE;   ! SET UP REASON FOR DROPPING UNIT
;          2575                        DODU (.L$LUN);              ! DROP UNIT
;          2576                        end
;          2577                    else
;          2578
;          2579                        if ((.RP_ADDR [U_FLGS] and UF_WPH) eql UF_WPH) and  ! STATUS CODE IS O.K.
;          2580                           (.CST_ADDR [.CUOFF, D_PROT] eql UNPROTECTED)
;          2581                        then
;          2582                            begin
;          2583                            T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;          2584                            CST_ADDR [.CUOFF, D_FATAL] = TRUE;
;          2585                            ERRDF (16, EGD_16, EMS_30);             ! WRITE-PROTECT CONFLICT
;          2586                            DUR [.L$LUN] = DU_PROTECT;      ! SET REASON TO DROP UNIT
;          2587                            DODU (.L$LUN);                  ! DROP UNIT
;          2588                            end
;          2589                        else
;          2590                            begin                                       ! WRITE PROTECT SWITCH IS O.K.
;          2591                            MAX_LBN [.L$LUN] = .RP_ADDR [USIZ_LO] - 1;  ! LARGEST LBN
;          2592                            CST_ADDR [.CUOFF, D_STA:] = ONLINE;
;          2593                            CST [.CCTLR, U_CNT] = .CST [.CCTLR, U_CNT] + 1;
;          2594                            end;
;          2595
;          2596                        end;                                ! IF RETPKT HAS CORRECT ENDCODE
;          2597
;          2598            PUT_RETPKT (.RP_INDX);
;          2599            end;                                            ! IF SEND WAS SUCCESSFUL
;          2600
;          2601        end;                                                ! ROUTINE UNIT_INIT


                                        .SBTTL  UNIT.INIT INITIALIZATION TEST ROUTINES
000000  004137  000000G                UNIT.INIT:
```

```
                                        JSR     R1,$SAVE3                       ;                               2494
000004  013746  000000G                 MOV     CCTLR,-(SP)                     ;                               2511
000010  004737  000000G                 JSR     PC,GET.PKT
000014  010037  000000G                 MOV     R0,P.INDEX
000020  010016                          MOV     R0,(SP)                         ; P.INDEX,*                     2512
000022  012746  000104                  MOV     #104,-(SP)
000026  004737  000000G                 JSR     PC,BL$MUL
000032  012760  000044  000006G         MOV     #44,MSCP.PKT+6(R0)
000040  013760  000000G  000016G         MOV     CDISK,MSCP.PKT+16(R0)           ;                               2513
000046  112760  000011  000022G         MOVB    #11,MSCP.PKT+22(R0)             ;                               2514
000054  012760  000001  000046G         MOV     #1,MSCP.PKT+46(R0)              ;                               2515
000062  012760  000001  000004G         MOV     #1,MSCP.PKT+4(R0)               ;                               2516
000070  013716  000000G                 MOV     P.INDEX,(SP)                    ;                               2518
000074  004737  000000G                 JSR     PC,SEND
000100  005700                          TST     R0
000102  001033                          BNE     1$
000104  013700  000000G                 MOV     T.ADDR,R0                       ;                               2521
000110  105260  000063                  INCB    63(R0)
000114  013700  000000G                 MOV     CUOFF,R0                        ;                               2522
000120  006300                          ASL     R0
000122  063700  000000G                 ADD     CST.ADDR,R0
000126  052710  010000                  BIS     #10000,(R0)
000132  104455                          TRAP    55                              ;                               2523
000134  000026                          .WORD   26
000136  000000G                         .WORD   EGD.22
000140  000000                          .WORD   0
000142  013700  000000G                 MOV     L$LUN,R0                        ;                               2524
000146  112760  000007  000000G         MOVB    #7,DUR(R0)
000154  104451                          TRAP    51                              ;                               2525
000156  013716  000000G                 MOV     P.INDEX,(SP)                    ;                               2526
000162  004737  000000G                 JSR     PC,PUT.PKT
000166  000137  004454'                 JMP     10$                             ;                               2518
000172  004737  000000G         1$:     JSR     PC,WAIT                         ;                               2533
000176  004737  000000G                 JSR     PC,OUT.IODQ                     ;                               2534
000202  010037  000000G                 MOV     R0,RP.INDX
000206  010016                          MOV     R0,(SP)                         ; RP.INDX,*                     2535
000210  012746  000060                  MOV     #60,-(SP)
000214  004737  000000G                 JSR     PC,BL$MUL
000220  062700  000000G                 ADD     #RETPKT,R0
000224  010037  000000G                 MOV     R0,RP.ADDR
000230  132760  000360  000002         BITB    #360,2(R0)                      ;                               2537
000236  001404                          BEQ     2$
000240  013716  000000G                 MOV     RP.INDX,(SP)                    ;                               2539
000244  004737  000000G                 JSR     PC,PUT.RETPKT
000250  005726                  2$:     TST     (SP)+                           ;                               2532
000252  013701  000000G                 MOV     RP.ADDR,R1                      ;                               2542
000256  005000                          CLR     R0
000260  126127  000003  000003         CMPB    3(R1),#3
000266  001002                          BNE     3$
000270  005200                          INC     R0
000272  000407                          BR      4$
000274  132761  000360  000002         BITB    #360,2(R1)                      ;                               2543
000302  001333                  3$:     BNE     1$
000304  105761  000014                  TSTB    14(R1)                          ;                               2544
000310  100330                          BPL     1$
000312  006000                  4$:     ROR     R0                              ;                               2546
```

```
000314  103003                                  BCC     5$
000316  004737  000000V                         JSR     PC,DR.ERR               ;                       2550
000322  000554                                  BR      9$                      ;                       2546
000324  013703  000000G                 5$:     MOV     CUOFF,R3                ;                       2558
000330  006303                                  ASL     R3
000332  063703  000000G                         ADD     CST.ADDR,R3
000336  013702  000000G                         MOV     L$LUN,R2                ;                       2560
000342  126127  000014  000211                 CMPB    14(R1),#211             ;                       2554
000350  001420                                  BEQ     6$
000352  013700  000000G                         MOV     T.ADDR,R0               ;                       255
000356  105260  000062                          INCB    62(R0)
000362  052713  010000                          BIS     #10000,(R3)             ;                       2558
000366  104455                                  TRAP    55                      ;                       2559
000370  000027                                  .WORD   27
000372  000000G                                 .WORD   EGD.23
000374  000000G                                 .WORD   EMS.21
000376  112762  000007  000000G                 MOVB    #7,DUR(R2)              ;                       2560
000404  010200                                  MOV     R2,R0                   ;                       2561
000406  104451                                  TRAP    51
000410  000521                                  BR      9$                      ;                       2554
000412  116137  000016  000000G         6$:     MOVB    16(R1),ST.CODE          ;                       2565
000420  042737  177740  000000G                 BIC     #177740,ST.CODE
000426  016100  000016                          MOV     16(R1),R0               ;                       2566
000432  006200                                  ASR     R0
000434  006200                                  ASR     R0
000436  006200                                  ASR     R0
000440  006200                                  ASR     R0
000442  006200                                  ASR     R0
000444  042700  174000                          BIC     #174000,R0
000450  010037  000000G                         MOV     R0,SB.CODE
000454  005737  000000G                         TST     ST.CODE                 ;                       2568
000460  001420                                  BEQ     7$
000462  013700  000000G                         MOV     T.ADDR,R0               ;                       2571
000466  105260  000062                          INCB    62(R0)
000472  052713  010000                          BIS     #10000,(R3)             ;                       2572
000476  104455                                  TRAP    55                      ;                       2573
000500  000017                                  .WORD   17
000502  000000G                                 .WORD   EGD.15
000504  000000G                                 .WORD   EMS.30
000506  112762  000007  000000G                 MOVB    #7,DUR(R2)              ;                       2574
000514  010200                                  MOV     R2,R0                   ;                       2575
000516  104451                                  TRAP    51
000520  000455                                  BR      9$                      ;                       2568
000522  032761  020000  000022         7$:     BIT     #20000,22(R1)           ;                       2579
000530  001427                                  BEQ     8$
000532  013700  000000G                         MOV     CUOFF,R0                ;                       2580
000536  006300                                  ASL     R0
000540  063700  000000G                         ADD     CST.ADDR,R0
000544  005710                                  TST     (R0)
000546  100020                                  BPL     8$
000550  013700  000000G                         MOV     T.ADDR,R0               ;                       2583
000554  105260  000062                          INCB    62(R0)
000560  052713  010000                          BIS     #10000,(R3)             ;                       2584
000564  104455                                  TRAP    55                      ;                       2585
000566  000020                                  .WORD   20
000570  000000G                                 .WORD   EGD.16
```

```
000000G                         .WEAK   $$MS.50
000004  112762  000011  000000G       MOVB    #11,DUB(R2)             ;2586
000010  010200                  98:     MOV     R2,R0                   ;2587
000012  104451                          TRAP    51
000014  000422                          BR      98                      ;2589
000016  010200                  98:     MOV     R2,R0                   ;2591
000020  006300                          ASL     R0
000022  016160  000044  000064          MOV     44(R1),MAX.LBN(R0)
000030  005360  000064                  DEC     MAX.LBN(R0)
000034  052715  020000                  BIS     #20000,(R5)             ;2592
000040  013716  000000G                 MOV     CCTLR,(SP)              ;2593
000044  012746  000056                  MOV     #56,-(SP)
000050  004737  000000G                 JSR     PC,$$LSMUL
000054  105260  000005G                 INCB    CST+5(R0)
000060  005726                          TST     (SP)+                   ;2590
000062  013716  000000G         98:     MOV     RP.INDX,(SP)            ;2598
000066  004737  000000G                 JSR     PC,PUT.RETPKT
000072  022626                  108:    CMP     (SP)+,(SP)+             ;2509
000074  000207                          RTS     PC                      ;2494

; Routine Size:  220 words,    Routine Base:  $CODE$ + 3570
; Maximum stack depth per invocation:  8 words
```

```
;  2602  routine DR_ERR : novalue =
;  2603
;  2604  !+
;  2605  !       THIS ROUTINE IS DESIGNED TO PROCESS RETURN PACKETS THAT ORIGINATE AT
;  2606  !       THE "DRIVER" RATHER THAN THE DEVICE. DRIVER ORIGINATED PACKETS INDICATE
;  2607  !       EITHER A FATAL DEVICE ERROR OR A COMMAND TIMEOUT. SINCE THIS ROUTINE IS
;  2608  !       ONLY CALLED DURING THE INITIALIZATION TEST, IT TREATS A COMMAND TIMEOUT
;  2609  !       AS AN INITIALIZATION ERROR.
;  2610  !
;  2611  !       IMPLICIT INPUTS:
;  2612  !               RP_ADDR   ADDRESS OF A RETPKT THAT ORIGINATED AT THE "DRIVER"
;  2613  !                       (I.E., CONNECTION ID = CID DRIVER)
;  2614  !-
;  2615
;  2616      begin
;  2617
;  2618      local
;  2619          REASON : word initial (DU_TIME);        ! ASSUME COMMAND TIMEOUT
;  2620
;  2621      if .RP_ADDR [MESTYP] eql MT_FATAL           ! IF FATAL DEVICE ERROR
;  2622      then
;  2623          REASON = DU_DFATAL;                     ! CHANGE REASON TO FATAL ERROR
;  2624
;  2625      DROP_CTLR (.CCTLR, .REASON);                ! DROP ALL UNITS ON CONTROLLER
;  2626      end;

                                        .SBTTL  DR.ERR INITIALIZATION TEST ROUTINES
000000  010146                  DR.ERR: MOV     R1,-(SP)                ;                       2602
000002  012701  000012                  MOV     #12,R1                  ; *,REASON              2616
000006  013700  000000G                 MOV     RP.ADDR,R0              ;                       2621
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              2623
000034  013746  000000G           1$:    MOV     CCTLR,-(SP)             ;                       2625
000040  010146                          MOV     R1,-(SP)                ; REASON,*
000042  004737  000000G                 JSR     PC,DROP.CTLR
000046  022626                          CMP     (SP)+,(SP)+             ;                       2616
000050  012601                          MOV     (SP)+,R1                ;                       2602
000052  000207                          RTS     PC

; Routine Size:  22 words,      Routine Base:  $CODE$ + 4460
; Maximum stack depth per invocation:  4 words
```

```
;          2627  routine ACCESS : novalue =
;          2628
;          2629  !+
;          2630  !       THIS ROUTINE IS CALLED BY INIT_TEST TO VERIFY THAT THE CURRENT DISK
;          2631  !       CAN BE ACCESSED. THIS OBJECTIVE IS ACCOMPLISHED BY FORMATTING AND
;          2632  !       SENDING ONE OR TWO MSCP ACCESS COMMANDS TO THE DISK, AND CHECKING
;          2633  !       THE STATUS FIELD OF THE RESPONSE MESSAGE(S).
;          2634  !
;          2635  !       IMPLICIT INPUTS:
;          2636  !               CCTLR   CURRENT CONTROLLER NUMBER
;          2637  !               CDISK   CURRENT DISK ADDRESS (RD/RX DISK NUMBER)
;          2638  !               L$LUN - CURRENT (DRS) UNIT NUMBER
;          2639  !-
;          2640
;          2641      begin
;          2642
;          2643  local
;          2644          RESULT : word initial (FAILURE),        ! GUILTY UNTIL PROVEN INNOCENT
;          2645          LBN : word,
;          2646          PASS : word initial (1);                ! LOOP PASS COUNT
;          2647
;          2648      ST_CODE = SB_CODE = 0;                      ! STATUS CODE AND SUB CODE
;          2649      LBN = (((.MAX_LBN [.L$LUN] + 1) +  1) and %o'77777') - 1;
;          2650                                                  ! START WITH LAST LBN ON TOP SURFACE: [(x+1)/2]  1
;          2651
;          2652      do
;          2653          begin                                   ! LOOP STARTS HERE
;          2654          P_INDEX = GET_PKT (.CCTLR);             ! GET AN MSCP PACKET
;          2655          MSCP_PKT [.P_INDEX, DK_NUM] = .CDISK;   ! SET DISK ADDR (RD/RX DISK NUMBER)
;          2656          MSCP_PKT [.P_INDEX, OPCODE] = OP_ACC;   ! ACCESS OPCODE
;          2657          MSCP_PKT [.P_INDEX, BC_LO] = 512;       ! BYTE COUNT (1 BLOCK)
;          2658          MSCP_PKT [.P_INDEX, LBN_L] = .LBN;      ! LOGICAL BLOCK NUMBER
;          2659          MSCP_PKT [.P_INDEX, CMD_TYPE] = NON_SEQ_CMD;    ! NON-SEQUENTIAL COMMAND
;          2660
;          2661          if SEND (.P_INDEX) eql FAILURE          ! ATTEMPT TO SEND; IF CTLR NOT ONLINE
;          2662          then
;          2663              begin
;          2664              PUT_PKT (.P_INDEX);                 ! RETURN PACKET TO POOL
;          2665              PASS = 2;                           ! NO MORE TRIES
;          2666              end
;          2667          else
;          2668              begin                               ! IF SEND WAS SUCCESSFUL
;          2669
;          2670              do
;          2671                  begin
;          2672                  WAIT ();                        ! WAIT FOR RESPONSE
;          2673                  RP_INDX = OUT_IDDQ ();          ! GET RETPKT (RESPONSE) INDEX
;          2674                  RP_ADDR = RETPKT + (.RP_INDX * RP_LEN * 2);     ! CALCULATE RETPKT ADDRESS
;          2675
;          2676                  if .RP_ADDR [MESTYP] neq MT_SEQ ! RETURN ALL RETPKTS NOT SENT BY CONTROLLER
;          2677                  then
;          2678                      PUT_RETPKT (.RP_INDX);
;          2679
;          2680                  end
;          2681              until (.RP_ADDR [CONID] eql CID_DRIVER) or
;          2682                    ((.RP_ADDR [MESTYP] eql MT_SEQ) and
```

```
;          2683                 ((.RP_ADDR [ENDCOD] and OP_END) eql OP_END));
;          2684
;          2685             if .RP_ADDR [CONID] eql CID_DRIVER  ! IF RETPKT CAME FROM "DRIVER"
;          2686             then
;          2687                 PASS = 2                        ! NO MORE TRIES
;          2688             else
;          2689
;          2690                  f .RP_ADDR [ENDCOD] neq (OP_ACC or OP_END)
;          2691                 then
;          2692                     begin
;          2693 !                   PRINTF (DBM29);             ! 'RETPKT HAS BAD ENDCODE'      !USD REV A
;          2694                     EMSCMD ();
;          2695                     end
;          2696                 else
;          2697                     begin                       ! RETPKT HAS CORRECT ENDCODE
;          2698                     ST_CODE = .RP_ADDR [STSCOD];! GET STATUS CODE FROM PACKET
;          2699                     SB_CODE = .RP_ADDR [SUBCOD];! GET SUB-CODE FROM PACKET
;          2700
;          2701                     if .ST_CODE eql ST_SUC      ! IF STATUS CODE INDICATES SUCCESS
;          2702                     then
;          2703                         begin
;          2704                         RESULT = SUCCESS;
;          2705                         PASS = 2;               ! NO NEED TO TRY AGAIN
;          2706                         end;
;          2707
;          2708                     end;                        ! IF RETPKT HAS CORRECT ENDCODE
;          2709
;          2710             PUT_RETPKT (.RP_INDX);
;          2711             end;                                ! IF SEND WAS SUCCESSFUL
;          2712
;          2713             LBN = .LBN + 1;                     ! ADVANCE TO FIRST LBN OF BOTTOM SURFACE
;          2714             PASS = .PASS + 1;                   ! SECOND PASS
;          2715             end                                 ! END OF PASS LOOP
;          2716         until .PASS gequ 3;
;          2717
;          2718         if .RESULT eql FAILURE
;          2719         then
;          2720             begin
;          2721             T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;          2722             CST_ADDR [.CUOFF, D_FATAL] = TRUE;  ! FATAL ERROR
;          2723             ERRDF (17, EGD_17, EMS_30);                 ! ACCESS FAILED
;          2724             DUR [.L$LUN] = DU_ACCESS;           ! SET REASON TO DROP UNIT
;          2725             DODU (.L$LUN);                      ! DROP UNIT
;          2726             end;                                ! IF ACCESS FAILED
;          2727
;          2728         end;                                    ! ROUTINE ACCESS

                                        .SBTTL  ACCESS INITIALIZATION TEST ROUTINES
000000  004137  000000G       ACCESS: JSR     R1,$SAVE4                       ;                       2627
000004  005003                        CLR     R3                              ; RESULT                2641
000006  012702  000001                MOV     #1,R2                           ; *,PASS
000012  005037  000000G               CLR     SB.CODE                         ;                       2648
000016  005037  000000G               CLR     ST.CODE
000022  013700  000000G               MOV     L$LUN,R0                        ;                       2649
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$:     MOV     CCTLR, (SP)             ;                                       2654
000054  004737  000000G                         JSR     PC,GET.PKT
000060  010037  000000G                         MOV     R0,P.INDEX
000064  010016                                  MOV     R0,(SP)                 ; P.INDEX,*                             2655
000066  012746  000104                          MOV     #104,-(SP)
000072  004737  000000G                         JSR     PC,BL$MUL
000076  013760  000000G 000016G                 MOV     CDISK,MSCP.PKT+16(R0)
000104  112760  000020  000022G                 MOVB    #20,MSCP.PKT+22(R0)     ;                                       2656
000112  012760  001000  000026G                 MOV     #1000,MSCP.PKT+26(R0)   ;                                       2657
000120  010460  000046G                         MOV     R4,MSCP.PKT+46(R0)      ; LBN,*                                 2658
000124  012760  000002  000004G                 MOV     #2,MSCP.PKT+4(R0)       ;                                       2659
000132  013716  000000G                         MOV     P.INDEX,(SP)            ;                                       2661
000136  004737  000000G                         JSR     PC,SEND
000142  005700                                  TST     R0
000144  001007                                  BNE     2$
000146  013716  000000G                         MOV     P.INDEX,(SP)            ;                                       2664
000152  004737  000000G                         JSR     PC,PUT.PKT
000156  012702  000002                          MOV     #2,R2                   ; *,PASS                                2665
000162  000515                                  BR      9$                      ;                                       2661
000164  004737  000000G                 2$:     JSR     PC,WAIT                 ;                                       2672
000170  004737  000000G                         JSR     PC,OUT.ICDQ             ;                                       2673
000174  010037  000000G                         MOV     R0,RP.INDX
000200  010016                                  MOV     R0,(SP)                 ; RP.INDX,*                             2674
000202  012746  000060                          MOV     #60,-(SP)
000206  004737  000000G                         JSR     PC,BL$MUL
000212  062700  000000G                         ADD     #RETPKT,R0
000216  010037  000000G                         MOV     R0,RP.ADDR
000222  132760  000360  000002                  BITB    #360,2(R0)              ;                                       2676
000230  001404                                  BEQ     3$
000232  013716  000000G                         MOV     RP.INDX,(SP)            ;                                       2678
000236  004737  000000G                         JSR     PC,PUT.RETPKT
000242  005726                         3$:     TST     (SP)+                   ;                                       2671
000244  013701  000000G                         MOV     RP.ADDR,R1              ;                                       2681
000250  005000                                  CLR     R0
000252  126127  000003  000003                  CMPB    3(R1),#3
000260  001002                                  BNE     4$
000262  005200                                  INC     R0
000264  000407                                  BR      5$
000266  132761  000360  000002          4$:     BITB    #360,2(R1)              ;                                       2682
000274  001333                                  BNE     2$
000276  105761  000014                          TSTB    14(R1)                  ;                                       2683
000302  100330                                  BPL     2$
000304  006000                         5$:     ROR     R0                      ;                                       2685
000306  103435                                  BLO     7$                      ;                                       2687
000310  126127  000014  000220                  CMPB    14(R1),#220             ;                                       2690
000316  001403                                  BEQ     6$
000320  004737  000000G                         JSR     PC,EMSCMD               ;                                       2694
000324  000430                                  BR      8$                      ;                                       2690
000326  116137  000016  000000G         6$:     MOVB    16(R1),ST.CODE          ;                                       2698
000334  042737  177740  000000G                 BIC     #177740,ST.CODE
```

```
000342  016100  000016                        MOV     16(R1),R0               ;                               2699
000346  006200                                ASR     R0
000350  006200                                ASR     R0
000352  006200                                ASR     R0
000354  006200                                ASR     R0
000356  006200                                ASR     R0
000360  042700  174000                        BIC     #174000,R0
000364  010037  000000G                       MOV     R0,SB.CODE
000370  005737  000000G                       TST     ST.CODE                 ;                               2701
000374  001004                                BNE     8$
000376  012703  000001                        MOV     #1,R3                   ; *,RESULT                      2704
000402  012702  000002              7$:       MOV     #2,R2                   ; *,PASS                        2705
000406  013716  000000G             8$:       MOV     RP.INDX,(SP)            ;                               2710
000412  004737  000000G                       JSR     PC,PUT.RETPKT
000416  005204                      9$:       INC     R4                      ; LBN                           2713
000420  005202                                INC     R2                      ; PASS                          2714
000422  022626                                CMP     (SP)+,(SP)+             ;                               2653
000424  020227  000003                        CMP     R2,#3                   ; PASS,*                        2716
000430  103607                                BLO     1$
000432  005703                                TST     R3                      ; RESULT                        2718
000434  001025                                BNE     10$
000436  013700  000000G                       MOV     T.ADDR,R0               ;                               2721
000442  105260  000062                        INCB    62(R0)
000446  013700  000000G                       MOV     CUOFF,R0                ;                               2722
000452  006300                                ASL     R0
000454  063700  000000G                       ADD     CST.ADDR,R0
000460  052710  010000                        BIS     #10000,(R0)
000464  104455                                TRAP    55                      ;                               2723
000466  000021                                .WORD   21
000470  000000G                               .WORD   EGD.17
000472  000000G                               .WORD   EMS.30
000474  013700  000000G                       MOV     L$LUN,R0                ;                               2724
000500  112760  000010  000000G               MOVB    #10,DUR(R0)
000506  104451                                TRAP    51                      ;                               2725
000510  000207                      10$:       RTS     PC                      ;                               2627

; Routine Size:  165 words,     Routine Base:  $CODE$ + 4534
; Maximum stack depth per invocation:  9 words
```

```
;          2729  %sbttl 'MULTI DRIVE TEST ROUTINES
;          2730
;          2731  routine MULTI_DRIVE : novalue =
;          2732
;          2733  !+
;          2734  !       THIS SUBTEST IS THE MOST SIGNIFICANT PART OF THE ENTIRE PROGRAM. THE
;          2735  !       MULTI-DRIVE TEST IS A HOST-CONTROLLED EXERCISER DESIGNED TO GIVE THE
;          2736  !       USER AN INDICATION OF HOW ONE OR SEVERAL RDRX DRIVES WOULD PERFORM IN
;          2737  !       AN OPERATING SYSTEM ENVIRONMENT.
;          2738  !
;          2739  !       THIS ROUTINE ACTS AS AN "EXECUTIVE" TO THE WHOLE PROCESS. AFTER
;          2740  !       INVOKING MD_INIT TO INITIALIZE MULTI-DRIVE TEST DATA, THIS ROUTINE
;          2741  !       ENTERS A LOOP WHICH ISSUES QIOS TO ALL ACTIVE CONTROLLERS AND PROCESSES
;          2742  !       ANY RESPONSES. IN ADDITION, ALL OUTSTANDING COMMANDS ARE TIMED IN
;          2743  !       DRV_TIMCHK WHICH IS INVOKED EVERY SECOND. NORMAL TERMINATION OF THIS
;          2744  !       LOOP OCCURS WHEN QIOS ARE NO LONGER BEING ISSUED, AND ALL OUTSTANDING
;          2745  !       QIOS HAVE COMPLETED.
;          2746  !-
;          2747
;          2748      begin
;          2749
;          2750      MD_INIT ();                                     ! INIT MULTI DRIVE TEST DATA
;          2751
;          2752      do  begin                                       ! START OF EXECUTIVE LOOP
;          2753
;          2754          incr CTLR from 0 to (MAX_CTLR - 1) do       ! FOR EACH CONTROLLER
;          2755              begin
;          2756              SET_CPAR (.CTLR);                       ! SET UP CURRENT CONTROLLER PARAMETERS
;          2757
;          2758  !           SETPRI (PRI07);                         ! NO INTERRUPTS WHEN EXAMINING SA !JSD REV A
;          2759              SETPRI (PRI06);                         ! (EXCEPT ODT BREAK) !JSD REV A
;          2760              ICTLR = .CCTLR;                         ! FAKE INTERRUPTING CONTROLLER'S NUMBER
;          2761              ICST_ADDR = .CST_ADDR;                  ! FAKE INTERRUPTING CONTROLLER'S CST ADDRESS
;          2762              IDCT_ADDR = .DCT_ADDR;                  ! FAKE INTERRUPTING CONTROLLER'S DCT ADDRESS
;          2763              IRDRX_ADDR = .ICST_ADDR [IP_ADDR];      ! FAKE INTERRUPTING CONTROLLER'S ADDRESS
;          2764              IDCT_ADDR [SA_SAVE] = .IRDRX_ADDR [RCSA, RC_ALL];
;          2765                                                      ! CONTENTS OF THE SA REGISTER
;          2766
;          2767              if BIT_TST (IDCT_ADDR [SA_SAVE], SA_ERR)    ! IF SA SHOWS AN ERROR
;          2768              then
;          2769                  begin
;          2770                  FATAL_ERROR ();                     ! DECLARE FATAL ERROR
;          2771                  SETPRI (PRI00);                     ! LOWER PRIORITY
;          2772                  exitloop;                           ! QUIT
;          2773                  end
;          2774              else
;          2775                  SETPRI (PRI00);                     ! IF NO ERROR, CONTINUE
;          2776
;          2777              if QIO_OK ()                            ! IF O.K. TO ISSUE QIO(S) TO THIS CONTROLLER
;          2778              then
;          2779                  begin                               ! THEN
;          2780                  QIO_GEN ();                         ! GENERATE 1 OR 2 QIOS
;          2781
;          2782                  if (.MX1 geq 0) and                 ! IF SUCCESS ON FIRST QIO
;          2783                     (not .EOP_FLAG)
;          2784                  then
```

```
;         2785
;         2786                    if SEND (.MX1) eql SUCCESS          ! ATTEMPT TO SEND IT. IF SUCCESS
;         2787                    then
;         2788                        QIO [.CTLR] = .QIO [.CTLR] + 1  ! INCR OUTSTANDING QIO COUNT
;         2789                    else
;         2790                        PUT_PKT (.MX1);                 ! RETURN PACKET TO POOL
;         2791
;         2792                if (.MX2 geq 0) and                     ! IF SUCCESS ON SECOND QIO
;         2793                   (not .EOP_FLAG)
;         2794                then
;         2795
;         2796                    if SEND (.MX2) eql SUCCESS          ! ATTEMPT TO SEND IT. IF SUCCESS
;         2797                    then
;         2798                        QIO [.CTLR] = .QIO [.CTLR] + 1  ! INCR OUTSTANDING QIO COUNT
;         2799                    else
;         2800                        PUT_PKT (.MX2);                 ! RETURN PACKET TO POOL
;         2801
;         2802                end;                                    ! O.K. TO ISSUE QIO(S)
;         2803
;         2804            end;                                        ! CONTROLLER LOOP
;         2805
;         2806        BREAK;                                          ! BREAK FOR SUPERVISOR TO CATCH USER REQUESTS
;         2807
;         2808        if not .EOP_FLAG
;         2809        then
;         2810            PROC_RETPKT ();                             ! PROCESS ANY RETURN PACKETS
;         2811
;         2812        end                                             ! EXECUTIVE PROCESSING LOOP
;         2813
;         2814        until ((not QIO_OUT ()) or .EOP_FLAG);
;         2815
;         2816        end;                                                    ! EXERCISER


                                        .SBTTL  MULTI.DRIVE MULTI-DRIVE TEST ROUTINES
000000  004137  000000G           MULTI.DRIVE:
                                        JSR     R1,$SAVE2                       ;                               2731
000004  005746                          TST     -(SP)
000006  004737  000000V                   JSR     PC,MD.INIT                      ;                               2750
000012  005002                  1$:     CLR     R2                              ; CTLR                          2754
000014  010246                  2$:     MOV     R2,-(SP)                        ; CTLR,*                        2756
000016  004737  000000G                   JSR     PC,SET.CPAR
000022  012700  000300                    MOV     #300,R0                         ;                               2759
000026  104441                          TRAP    41
000030  013737  000000G 000156'           MOV     CCTLR,ICTLR                     ;                               2760
000036  013737  000000G 000106'           MOV     CST.ADDR,ICST.ADDR              ;                               2761
000044  013737  000000G 000110'           MOV     DCT.ADDR,IDCT.ADDR              ;                               2762
000052  017737  000106' 000000G           MOV     @ICST.ADDR,IRDRX.ADDR           ;                               2763
000060  013701  000110'                   MOV     IDCT.ADDR,R1                    ;                               2764
000064  013700  000000G                   MOV     IRDRX.ADDR,R0
000070  016066  000002  000002            MOV     2(R0),2(SP)                     ; *,RC.REG
000076  016661  000002  000002            MOV     2(SP),2(R1)                     ; RC.REG,*
000104  016601  000002                    MOV     2(SP),R1                        ;                               2767
000110  042701  077777                    BIC     #77777,R1
000114  020127  100000                    CMP     R1,#-100000
000120  001006                          BNE     3$
```

```
000122  004737  000000V                        JSR     PC,FATAL.ERROR                  ;                               2770
000126  005000                                 CLR     R0                              ;                               2771
000130  104441                                 TRAP    41
000132  005726                                 TST     (SP)+                           ;                               2769
000134  000464                                 BR      8$
000136  005000                          3$:    CLR     R0                              ;                               2775
000140  104441                                 TRAP    41
000142  004737  000000V                        JSR     PC,QIO.OK                       ;                               2777
000146  006000                                 ROR     R0
000150  103052                                 BCC     7$
000152  004737  000000V                        JSR     PC,QIO.GEN                      ;                               2780
000156  013700  000160'                       MOV     MX1,R0                          ;                               2782
000162  002421                                 BLT     5$
000164  132737  000001  000000G               BITB    #1,EOP.FLAG                     ;                               2783
000172  001015                                 BNE     5$
000174  010016                                 MOV     R0,(SP)                         ;                               2786
000176  004737  000000G                        JSR     PC,SEND
000202  020027  000001                         CMP     R0,#1
000206  001003                                 BNE     4$
000210  105262  000000G                        INCB    QIO(R2)                         ; *(CTLR)                       2788
000214  000404                                 BR      5$                              ;                               2786
000216  013716  000160'                  4$:    MOV     MX1,(SP)                        ;                               2790
000222  004737  000000G                        JSR     PC,PUT.PKT
000226  013700  000162'                  5$:    MOV     MX2,R0                          ;                               2792
000232  002421                                 BLT     7$
000234  132737  000001  000000G               BITB    #1,EOP.FLAG                     ;                               2793
000242  001015                                 BNE     7$
000244  010016                                 MOV     R0,(SP)                         ;                               2796
000246  004737  000000G                        JSR     PC,SEND
000252  020027  000001                         CMP     R0,#1
000256  001003                                 BNE     6$
000260  105262  000000G                        INCB    QIO(R2)                         ; *(CTLR)                       2798
000264  000404                                 BR      7$                              ;                               2796
000266  013716  000162'                  6$:    MOV     MX2,(SP)                        ;                               2800
000272  004737  000000G                        JSR     PC,PUT.PKT
000276  005726                          7$:    TST     (SP)+                           ;                               2755
000300  005202                                 INC     R2                              ; CTLR                          2754
000302  000243                                 .WORD CLV!CLC

000304  003643                                 BLE     2$
000306  104422                          8$:    TRAP    22                              ;                               2804
000310  132737  000001  000000G               BITB    #1,EOP.FLAG                     ;                               2808
000316  001002                                 BNE     9$
000320  004737  000000V                        JSR     PC,PROC.RETPKT                  ;                               2810
000324  004737  000000V                  9$:   JSR     PC,QIO.OUT                      ;                               2814
000330  006000                                 ROR     R0
000332  103004                                 BCC     10$
000334  132737  000001  000000G               BITB    #1,EOP.FLAG
000342  001623                                 BEQ     1$
000344  005726                         10$:    TST     (SP)+                           ;                               2731
000346  000207                                 RTS     PC

; Routine Size:  116 words,     Routine Base:  $CODE$ + 5246
; Maximum stack depth per invocation:  7 words
```

```
;          2817  routine MD_INIT : novalue =
;          2818
;          2819  !+
;          2820  !       THIS ROUTINE IS CALLED BY ROUTINE MULTI_DRIVE TO INITIALIZE DATA ITEMS
;          2821  !       USED BY THE MULTI-DRIVE TEST.
;          2822  !
;          2823
;          2824      begin
;          2825
;          2826      if .ENTRY_REASON eql START                                  ! IF THIS IS A START
;          2827      then
;          2828          INIT_IO_BUFF ();                                        ! PARTITION FREE MEMORY INTO I/O BUFFERS
;          2829
;          2830      if (.ENTRY_REASON neq CONT) and                             ! IF START, RESTART, OR PWR FAIL
;          2831         (.ENTRY_REASON neq NEW_PASS)
;          2832      then
;          2833
;          2834          incr CTLR from 0 to (MAX_CTLR - 1) do
;          2835              begin
;          2836              SET_CPAR (.CTLR);
;          2837
;          2838              incr DISK from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;          2839                  begin
;          2840                  SET_UPAR (.DISK);
;          2841                  BST [.L$LUN] = .CST_ADDR [(.DISK + 1), D_BEG];  ! INITIALIZE BLOCK SEQUENCE TABLE
;          2842                  DPST [.L$LUN] = DP_CNT;                         ! INITIALIZE DATA PATTERN SEQUENCE TABLE
;          2843                  end;
;          2844
;          2845              end;
;          2846
;          2847      incr COUNT from 0 to (QIO_PER_CTLR * MAX_CTLR - 1) do       ! INITIALIZE I/O BUFFER ALLOCATION
;          2848          BUFF_OWN [.COUNT] = -1;                                 !     TABLE
;          2849
;          2850      end;                                                        ! ROUTINE MD_INIT


                                        .SBTTL  MD.INIT MULTI DRIVE TEST ROUTINES
000000  004137  000000G           MD.INIT:JSR     R1,$SAVE4                       ;                       2817
000004  123727  000000G 000001            CMPB    ENTRY.REASON,#1                 ;                       2826
000012  001002                          BNE     1$
000014  004737  000000V                   JSR     PC,INIT.IO.BUFF                 ;                       2828
000020  123727  000000G 000003    1$:     CMPB    ENTRY.REASON,#3                 ;                       2830
000026  001444                          BEQ     4$
000030  123727  000000G 000005            CMPB    ENTRY.REASON,#5                 ;                       2831
000036  001440                          BEQ     4$
000040  005004                          CLR     R4                              ; CTLR                  2834
000042  010446                  2$:     MOV     R4,-(SP)                        ; CTLR,*                2836
000044  004737  000000G                 JSR     PC,SET.CPAR
000050  012703  000003                  MOV     #3,R3                           ; *,DISK                2838
000054  010316                  3$:     MOV     R3,(SP)                         ; DISK,*                2840
000056  004737  000000G                 JSR     PC,SET.UPAR
000062  013702  000000G                 MOV     L$LUN,R2                        ;                       2841
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)            ;                                2842
000116  062703  000005                        ADD     #5,R3                   ; *,DISK                         2838
000122  020327  000022                        CMP     R3,#22                  ; DISK,*
000126  003752                                BLE     3$
000130  005726                                TST     (SP)+                   ;                                2835
000132  005204                                INC     R4                      ; CTLR                           2834
000134  000243                                .WORD CLV!CLC

000136  003741                                BLE     2$
000140  005000                        4$:     CLR     R0                      ; COUNT                          2847
000142  112760  000377  000000G       5$:     MOVB    #377,BUFF.OWN(R0)       ; *,*(COUNT)                     2848
000150  005200                                INC     R0                      ; COUNT                          2847
000152  020027  000007                        CMP     R0,#7                   ; COUNT,*
000156  003771                                BLE     5$
000160  000207                                RTS     PC                      ;                                2817

; Routine Size:  57 words,      Routine Base:  $CODE$ + 5616
; Maximum stack depth per invocation:  7 words
```

```
;          2851  routine INIT_IO_BUFF : novalue =
;          2852
;          2853  !+
;          2854  !       THIS ROUTINE IS CALLED BY MD_INIT WHEN THE MULTI-DRIVE TEST IS FIRST
;          2855  !       STARTED. IT IS RESPONSIBLE FOR PARTITIONING FREE MEMORY INTO A
;          2856  !       COLLECTION OF I/O BUFFERS. THE SIZE OF EACH I/O BUFFER IS DETERMINED
;          2857  !       BY A NUMBER OF FACTORS, INCLUDING THE NUMBER OF UNITS, THE NUMBER OF
;          2858  !       CONTROLLERS, AND THE SIZE OF FREE MEMORY.
;          2859  !
;          2860  !       ONCE THE BUFFER SIZE IS DETERMINED, THE NUMBER OF I/O BUFFERS IS
;          2861  !       CALCULATED. FINALLY, THE BUFFER ADDRESS (BUFF_ADDR) TABLE IS LOADED
;          2862  !       WITH FIXED BUFFER DESCRIPTORS THAT ARE USED IN THE ALLOCATION AND
;          2863  !       DEALLOCATION PROCESS.
;          2864  !
;          2865  !       IMPLICIT INPUTS:
;          2866  !               CTLR_CNT - THE NUMBER OF CONTROLLERS CONFIGURED
;          2867  !               L$UNIT - THE NUMBER OF UNITS AVAILABLE FOR TESTING
;          2868  !               FREE_MEM_ADDR - START OF FREE MEMORY
;          2869  !
;          2870
;          2871      begin
;          2872
;          2873      local
;          2874          DRS_START;
;          2875
;          2876      DRS_START = .FREE_MEM_ADDR + 2 + (..FREE_MEM_ADDR + 2);      ! START OF SUPERVISOR
;          2877      BUFF_ADDR [0] = (.FREE_MEM_ADDR + 2 + 1) and %o'177776';     ! START OF READ/WRITE BUFFERS
;          2878
;          2879      while (.BUFF_ADDR [0] and %o'37') neq 0 do                   ! FORCE FIRST I/O BUFFER TO START
;          2880          BUFF_ADDR [0] = .BUFF_ADDR [0] + 2;                      ! ON EVEN BOUNDARY
;          2881
;          2882      BYTS_PER_QIO = ((.DRS_START - .BUFF_ADDR [0]) / (QIO_PER_CTLR * MAX_CTLR)) and %o'177740';
;          2883                                                                   ! MAX TRANSFER SIZE
;          2884
;          2885      if .BYTS_PER_QIO gtru MAX_XFER_SIZE
;          2886      then
;          2887          BYTS_PER_QIO = MAX_XFER_SIZE;                            ! ADJUST TRANSFER SIZE LOWER
;          2888
;          2889      if .BYTS_PER_QIO lssu 32
;          2890      then
;          2891          begin
;          2892          ERRSF (2, EGS_02, 0);                                    ! ERROR IF NOT ENOUGH MEMORY
;          2893          DOCLN;
;          2894          end;
;          2895
;          2896      if (QIO_PER_CTLR * MAX_CTLR) gtru 1
;          2897      then
;          2898
;          2899      incr index from 1 to (QIO_PER_CTLR * MAX_CTLR - 1) do        ! INIT REMAINING TABLE ENTRIES
;          2900          BUFF_ADDR [.index] = .BUFF_ADDR [.index - 1] + .BYTS_PER_QIO;   ! FIXED BUFFER ADDRESS
;          2901
;          2902      end;                                                         ! ROUTINE INIT_IO_BUFF


                                        .SBTTL  INIT.IO.BUFF MULTI-DRIVE TEST ROUTINES
000000  004137  000000G                INIT.IO.BUFF:
```

N6


```
                                                JSR     R1,$SAVE3                       ;                               2851
000004  013701  000000G                         MOV     FREE.MEM.ADDR,R1                ;                               2876
000010  011100                                  MOV     (R1),R0
000012  006300                                  ASL     R0
000014  060100                                  ADD     R1,R0                           ;                               2871
000016  062700  000002                          ADD     #2,R0                           ;                               2876
000022  062701  000003                          ADD     #3,R1                           ;                               2877
000026  010137  000000G                         MOV     R1,BUFF.ADDR
000032  042737  000001  000000G                 BIC     #1,BUFF.ADDR
000040  032737  000037  000000G         1$:     BIT     #37,BUFF.ADDR                   ;                               2879
000046  001404                                  BEQ     2$
000050  062737  000002  000000G                 ADD     #2,BUFF.ADDR                    ;                               2880
000056  000770                                  BR      1$                              ;                               2879
000060  010046                          2$:     MOV     R0,-(SP)                        ; DRS.START,*                   2882
000062  163716  000000G                         SUB     BUFF.ADDR,(SP)
000066  012746  000010                          MOV     #10,-(SP)
000072  004737  000000G                         JSR     PC,BL$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              ;                               2885
000116  101403                                  BLOS    3$
000120  012737  002000  000000G                 MOV     #2000,BYTS.PER.QIO              ;                               2887
000126  023727  000000G 000040          3$:     CMP     BYTS.PER.QIO,#40                ;                               2889
000134  103005                                  BHIS    4$
000136  104454                                  TRAP    54                              ;                               2892
000140  000002                                  .WORD   2
000142  000000G                                 .WORD   EGS.02
000144  000000                                  .WORD   0
000146  104444                                  TRAP    44
000150  012702  000001                  4$:     MOV     #1,R2                           ; *,INDEX                       2896
000154  010201                          5$:     MOV     R2,R1                           ; INDEX,*                       2900
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                         2896
000202  020227  000007                          CMP     R2,#7                           ; INDEX,*
000206  003762                                  BLE     5$
000210  022626                                  CMP     (SP)+,(SP)+                     ;                               2871
000212  000207                                  RTS     PC                              ;                               2851

; Routine Size:  70 words,      Routine Base:  $CODE$ + 6000
; Maximum stack depth per invocation:  8 words
```

```
;        routine QIO OK =
;
;        !+
;        !       THIS ROUTINE IS CALLED BY THE MULTI_DRIVE "EXECUTIVE" IN ORDER TO
;        !       DETERMINE WHETHER OR NOT A QIO REQUEST (OR QIO PAIR) SHOULD BE
;        !       GENERATED TO THE CURRENT CONTROLLER. A VALUE OF "TRUE" IS RETURNED IF
;        !       THE CONTROLLER MEETS 3 REQUIREMENTS:
;        !
;        !               A. THE CONTROLLER IS ONLINE;
;        !               B. THE NUMBER OF OUTSTANDING QIOS IS AT LEAST 2 LESS THAN THE
;        !                       MAXIMUM ALLOWED FOR ANY ONE CONTROLLER;
;        !               C. THERE IS AT LEAST ONE DISK ONLINE TO THE CONTROLLER.
;        !
;        !       IF ANY OF THESE TEST FAIL, THEN A VALUE OF "FALSE" IS RETURNED.
;        !
;        !       IMPLICIT INPUTS:
;        !               CCTLR - CURRENT CONTROLLER NUMBER
;        !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;        !-
;
;            if (.CST_ADDR [STATE] eql ONLINE) and                ! IF CONTROLLER IS ONLINE
;               (not .EOP_FLAG) and
;               ((.QIO [.CCTLR] + 2) lequ QIO_PER_CTLR) and       ! IF OUTSTANDING QIO COUNT IS O.K.
;               (.CST_ADDR [U_CNT] neq 0)                         ! IF THERE IS VALID UNIT
;            then
;                return TRUE                                      ! "TRUE" EXIT POINT
;            else
;                return FALSE;                                    ! "FALSE" EXIT POINT

                                           .SBTTL  QIO.OK MULTI DRIVE TEST ROUTINES
000000  013700  000000G           QIO.OK:  MOV     CST.ADDR,R0                  ;        2923
000004  005760  000002                     TST     2(R0)
000010  100027                             BPL     1$
000012  132737  000001  000000G            BITB    #1,EOP.FLAG                  ;        2924
000020  001023                             BNE     1$
000022  013700  000000G                    MOV     CCTLR,R0                     ;        2925
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                  ;        2926
000054  105760  000005                     TSTB    5(R0)
000060  001403                             BEQ     1$
000062  012700  000001                     MOV     #1,R0                        ;        2903
000066  000207                             RTS     PC
000070  005000                    1$:      CLR     R0
000072  000207                             RTS     PC

; Routine Size:  30 words,     Routine Base:  $CODE$ + 6214
; Maximum stack depth per invocation:  0 words
```

```
;        2931  routine QIO_OUT =
;        2932
;        2933  !+
;        2934  !       THIS ROUTINE IS CALLED BY THE MULTI_DRIVE EXECUTIVE FOR DETERMINING THE
;        2935  !       END OF THE MULTI-DRIVE TEST. ITS PURPOSE IS TO EXAMINE THE QIO VECTOR
;        2936  !       FOR ANY OUTSTANDING QIOS ON ANY CONTROLLER. A VALUE OF "TRUE" IS
;        2937  !       RETURNED IF THERE IS AT LEAST ONE QIO OUTSTANDING ON ANY CONTROLLER.
;        2938  !       OTHERWISE, "FALSE" IS RETURNED INDICATING NO OUTSTANDING QIOS.
;        2939  !-
;        2940
;        2941      begin
;        2942
;        2943      incr CTLR from 0 to (MAX_CTLR - 1) do
;        2944          begin
;        2945          SET_CPAR (.CTLR);                       ! SET UP CURRENT CONTROLLER PARAMETERS
;        2946
;        2947          if .CST_ADDR [STATE] eql ONLINE         ! IF CONTROLLER IS ONLINE
;        2948          then
;        2949              return TRUE;
;        2950
;        2951          end;
;        2952
;        2953      return FALSE;                               ! EXIT   NO CONTROLLERS ONLINE
;        2954      end;


                                        .SBTTL  QIO.OUT MULTI-DRIVE TEST ROUTINES
000000  010146                  QIO.OUT:MOV     R1,-(SP)                ;                       2931
000002  005001                          CLR     R1                      ; CTLR                  2943
000004  010146                  1$:     MOV     R1,-(SP)                ; CTLR,*                2945
000006  004737  000000G                 JSR     PC,SET.CPAP
000012  013700  000000G                 MOV     CST.ADDR,R0             ;                       2947
000016  005760  000002                  TST     2(R0)
000022  100004                          BPL     2$
000024  005726                          TST     (SP)+                   ;                       2949
000026  012700  000001                  MOV     #1,R0
000032  000405                          BR      3$
000034  005726                  2$:     TST     (SP)+                   ;                       2944
000036  005201                          INC     R1                      ; CTLR                  2943
000040  000243                          .WORD CLV!CLC

000042  003760                          BLE     1$
000044  005000                          CLR     R0                      ;                       2941
000046  012601                  3$:     MOV     (SP)+,R1                ;                       2931
000050  000207                          RTS     PC

; Routine Size:  21 words,      Routine Base:  $CODE$ + 6310
; Maximum stack depth per invocation:  3 words
```

```
;          2955  routine QIO_GEN : novalue =
;          2956
;          2957  !+
;          2958  !       THIS ROUTINE IS CALLED BY THE MULTI_DRIVE EXECUTIVE FOR AN ONLINE
;          2959  !       CONTROLLER ELIGIBLE TO RECEIVE I/O TRANSFER REQUESTS. IT IS
;          2960  !       RESPONSIBLE FOR SECURING ONE OR TWO MSCP PACKETS AND LOADING THEM
;          2961  !       WITH VARIOUS PARAMETERS COMPRISING THE I/O REQUEST. THE I/O REQUEST
;          2962  !       GENERATED HERE IS DESTINED TO A PARTICULAR UNIT SELECTED AT RANDOM FROM
;          2963  !       THOSE CONFIGURED UNDER THE CURRENT CONTROLLER.
;          2964  !
;          2965  !       EACH FIELD OF THE PACKET(S) IS LOADED WITHIN INDIVIDUAL ROUTINES
;          2966  !       (QIO_FUNC, QIO_LBN, QIO_SIZE, ETC.). MOST OF THE VALUES SELECTED FOR
;          2967  !       EACH FIELD ARE BASED ON A SET OF RANDOM NUMBER GENERATED AT THE START.
;          2968  !
;          2969  !       UNDER NORMAL CIRCUMSTANCES, ONLY ONE I/O REQUEST IS GENERATED. HOWEVER,
;          2970  !       IF THIS I/O REQUEST IS A "WRITE", AND IF THE OPERATOR SELECTED THE
;          2971  !       OPTION FOR HOST WRITE-COMPARES, THEN A SECOND "READ" REQUEST WILL BE
;          2972  !       GENERATED WITH THE SAME LBN AND BYTE COUNT.
;          2973  !
;          2974  !       AFTER THE PACKET(S) HAVE BEEN LOADED, THIS ROUTINE REGAINS CONTROL
;          2975  !       AND ATTEMPTS TO GET ONE OR TWO I/O BUFFERS FOR THE ACTUAL DATA
;          2976  !       TRANSFERS. THE SUCCESS / FAIL STATUS OF THIS ENTIRE OPERATION IS
;          2977  !       PASSED BACK TO THE CALLER THROUGH THE GLOBALS "MX1" AND "MX2"; THEY
;          2978  !       CONTAIN VALID MSCP PACKET INDECES, OR -1.
;          2979  !
;          2980  !       Note that the DUP Exerciser is located inside the QIO_FUNC routine.
;          2981  !       Every  o often the Dup exerciser will run and return the MSCP Exerciser
;          2982  !       to it's normal state.
;          2983  !
;          2984  !
;          2985  !       IMPLICIT INPUTS:
;          2986  !               CCTLR - CURRENT CONTROLLER NUMBER
;          2987  !-
;          2988
;          2989      begin
;          2990      MX2 = -1;                                       ! ASSUME FAILURE IN SECURING 2ND PACKET
;          2991
;          2992      if (MX1 = GET_PKT (.CCTLR)) lss 0               ! TRY TO GET 1ST PACKET. IF FAILURE
;          2993      then
;          2994          return;                                     ! NO POINT IN CONTINUING
;          2995
;          2996      if (MX2 = GET_PKT (.CCTLR)) lss 0               ! TRY TO GET 2ND PACKET. IF FAILURE
;          2997      then
;          2998          begin
;          2999          PUT_PKT (.MX1);                             ! RETURN 1ST PACKET TO POOL
;          3000          MX1 = -1;                                   ! INDICATE FAILURE
;          3001          return;                                     ! DONE
;          3002          end;
;          3003
;          3004      MAD1 = MSCP_PKT + (.MX1 * PKT_LEN * 2);         ! CALCULATE STARTING ADDRESSES
;          3005      MAD2 = MSCP_PKT + (.MX2 * PKT_LEN * 2);         !     OF BOTH PACKETS
;          3006      GET_RANDOM ();                                  ! GENERATE A SET OF RANDOM NUMBERS
;          3007      QIO_UNIT ();                                    ! LOAD RANDOM UNIT NUMBER INTO PACKETS
;          3008
;          3009      if .EOP_FLAG                                    ! RETURN IF NO UNIT ONLINE
;          3010      then
```

```
;          3011          return;                                  ! LOAD RANDOM !MSCP FUNCTION CODE (OPCODE)0 OR !DUP EXERCISER TEST A
;          3012
;          3013      QIO_FUNC ();                                 ! THIS IS THE POINT WHERE THE DUP EXERCISER WILL CUT IN TO THE MSCP
NO MSCP OPCODE0
;          3014                                                   ! START WRITTING AND READING DBN S ONCE FINISHED IT WILL RETURN THE
EXERCISER !?
;          3015                                                   ! TO ITS NORMAL MSCP MODE....
EXERCISER
;          3016                                                   ! IF IT WAS IN DUP TEST AND FAILED TO GET A ENVOLOPE RETURN
;          3017      if (.MX1 lss 0) OR (.EOP_FLAG)               ! NO POINT IN CONTINUING
;          3018      then return;
;          3019
;          3020      QIO_LBN ();                                  ! LOAD LBN (RANDOM OR SEQUENTIAL)
;          3021      QIO_SIZE ();                                 ! LOAD RANDOM BYTE COUNT
;          3022      GET_IO_BUFF (MAD1 [BUF_0]);                  ! TRY TO GET AN I/O BUFFER
;          3023
;          3024      if .MX2 geq 0                                ! IF TWO QIOS ARE TO BE ISSUED
;          3025      then
;          3026          begin
;          3027          GET_IO_BUFF (MAD2 [BUF_0]);              ! TRY TO GET 2ND I/O BUFFER
;          3028
;          3029          if .MAD2 [BUF_0] eqla 0                  ! IF 2ND BUFFER ALLOCATION FAILED
;          3030          then
;          3031              begin
;          3032
;          3033              if .MAD1 [BUF_0] neqa 0              ! IF 1ST I/O BUFFER WAS ALLOCATED
;          3034              then
;          3035                  begin
;          3036                  PUT_IO_BUFF (MAD1 [BUF_0]);      ! RETURN 1ST I/O BUFFER TO POOL
;          3037                  MAD1 [BUF_0] = 0;                ! MARK IT AS FAILED
;          3038                  end;
;          3039
;          3040              PUT_PKT (.MX2);                      ! RETURN 2ND PACKET TO POOL
;          3041              MX2 = -1;                            ! INDICATE FAILURE
;          3042              end;                                 ! IF 2ND I/O BUFFER ALLOCATION FAILED
;          3043
;          3044          end;                                     ! IF TWO QIOS ARE TO BE ISSUED
;          3045
;          3046      if .MAD1 [BUF_0] eqla 0                      ! IF 1ST I/O BUFFER ALLOCATION FAILED
;          3047      then
;          3048          begin
;          3049          PUT_PKT (.MX1);                          ! RETURN 1ST PACKET TO POOL
;          3050          MX1 = -1;                                ! INDICATE FAILURE
;          3051          end
;          3052      else
;          3053
;          3054          if .MAD1 [OPCODE] eql OP_WRT             ! OTHERWISE, IF 1ST OPCODE IS A WRITE  (ALL IS O.K.)
;          3055          then
;          3056              FILL_BUFF ();                        ! FILL 1ST I/O BUFFER WITH APPROPRIATE DATA PATTERN
;          3057
;          3058      end;                                         ! ROUTINE QIO_GEN

                                        .SBTTL  QIO.GEN MULTI-DRIVE TEST ROUTINES
000000  012737  177777  000162'  QIO.GEN:MOV     #-1,MX2                          ;                                    2990
000006  013746  000000G                 MOV     CCTLR,-(SP)                      ;                                    2992
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$                      ;                       2994
000030  013746  000000G                 MOV     CCTLR, (SP)             ;                       2995
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)               ;                       2999
000056  004737  000000G                 JSR     PC,PUT.PKT
000062  012737  177777  000160         MOV     #-1,MX1                 ;                       3000
000070  005726                          TST     (SP)+                   ;                       2996
000072  000207                          RTS     PC                      ;                       2998
000074  013746  000160          1$:     MOV     MX1,-(SP)               ;                       3004
000100  012746  000104                  MOV     #104,-(SP)
000104  004737  000000G                 JSR     PC,BL$MUL
000110  062700  000000G                 ADD     #MSCP.PKT.R0
000114  010037  000164'                 MOV     R0,MAD1
000120  013716  000162                  MOV     MX2,(SP)                ;                       3005
000124  012746  000104                  MOV     #104,-(SP)
000130  004737  000000G                 JSR     PC,BL$MUL
000134  062700  000000G                 ADD     #MSCP.PKT,R0
000140  010037  000166'                 MOV     R0,MAD2
000144  004737  000000V                 JSR     PC,GET.RANDOM           ;                       3006
000150  004737  000000V                 JSR     PC,QIO.UNIT             ;                       3007
000154  132737  000001  000000G         BITB    #1,EOP.FLAG             ;                       3009
000162  001112                          BNE     5$                      ;                       2955
000164  004737  000000V                 JSR     PC,QIO.FUNC             ;                       3013
000170  005737  000160'                 TST     MX1                     ;                       3017
000174  002505                          BLT     5$
000176  132737  000001  000000G         BITB    #1,EOP.FLAG
000204  001101                          BNE     5$                      ;                       2955
000206  004737  000000V                 JSR     PC,QIO.LBN              ;                       3020
000212  004737  000000V                 JSR     PC,QIO.SIZE             ;                       3021
000216  013716  000164'                 MOV     MAD1,(SP)               ;                       3022
000222  062716  000032                  ADD     #32,(SP)
000226  004737  000000G                 JSR     PC,GET.IO.BUFF
000232  005737  000162'                 TST     MX2                     ;                       3024
000236  002437                          BLT     3$
000240  013716  000166'                 MOV     MAD2,(SP)               ;                       3027
000244  062716  000032                  ADD     #32,(SP)
000250  004737  000000G                 JSR     PC,GET.IO.BUFF
000254  013700  000166'                 MOV     MAD2,R0                 ;                       3029
000260  005760  000032                  TST     32(R0)
000264  001024                          BNE     3$
000266  013700  000164'                 MOV     MAD1,R0                 ;                       3033
000272  062700  000032                  ADD     #32,R0
000276  005710                          TST     (R0)
000300  001407                          BEQ     2$
000302  010016                          MOV     R0,(SP)                 ;                       3036
000304  004737  000000G                 JSR     PC,PUT.IO.BUFF
000310  013700  000164'                 MOV     MAD1,R0                 ;                       3037
000314  005060  000032                  CLR     32(R0)
000320  013716  000162'          2$:     MOV     MX2,(SP)                ;                       3040
000324  004737  000000G                 JSR     PC,PUT.PKT
000330  012737  177777  000162'         MOV     # 1,MX2                 ;                       3041
```

```
000356  013700  000164          3$:     MOV     MAD1,R0                         ;                       3046
000342  005760  000032                  TST     32(R0)
000346  001010                          BNE     4$
000350  013716  000160'                 MOV     MX1,(SP)                        ;                       3049
000354  004737  000000G                 JSR     PC,PUT.PKT
000360  012737  177777  000160'         MOV     #-1,MX1                         ;                       3050
000366  000410                          BR      5$                              ;                       3046
000370  013700  000164'          4$:     MOV     MAD1,R0                         ;                       3054
000374  126027  000022  000042          CMPB    22(R0),#42
000402  001002                          BNE     5$
000404  004737  000000V                 JSR     PC,FILL.BUFF                    ;                       3056
000410  062706  000006          5$:     ADD     #6,SP                           ;                       2989
000414  000207                  6$:     RTS     PC                              ;                       2955

; Routine Size:  135 words,    Routine Base:  $CODE$ + 6362
; Maximum stack depth per invocation:  4 words
```

```
;          3059  routine GET_RANDOM : novalue =
;          3060
;          3061  !+
;          3062  !       THIS ROUTINE IS CALLED BY QIO_GEN TO GENERATE A SET OF RANDOM NUMBERS,
;          3063  !       AND TO STORE THEM INTO THE RANDOM NUMBER TABLE (RANDOM). THE RANDOM
;          3064  !       NUMBERS ARE USED TO SELECT I/O REQUEST PARAMETERS FOR THE CURRENT QIO
;          3065  !       OR QIO PAIR. IN ADDITION, IF DATA PATTERN #1 IS BEING USED, THESE
;          3066  !       RANDOM NUMBERS WILL BE USED IN THE WRITE OPERATION.
;          3067  !
;          3068
;          3069      begin
;          3070
;          3071      own
;          3072          SEED : word initial (173),
;          3073          NEXT_RANDNUM : word initial (245);
;          3074
;          3075      incr COUNT from 0 to (RDM_LEN   1) do
;          3076          begin
;          3077          SEED = (.SEED * .NEXT_RANDNUM + 1) ^ 4;
;          3078          NEXT_RANDNUM = (.NEXT_RANDNUM / 4) + .SEED;
;          3079          RANDOM [.COUNT] = .NEXT_RANDNUM;
;          3080          end;
;          3081
;          3082      end;


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

```
                                        .SBTTL  GET.RANDOM MULTI DRIVE TEST ROUTINES
007000                                  .PSECT  $CODE$,  RO

000000  004137  000000G            GET.RANDOM:
                                        JSR     R1,$SAVE3                       ;                               3059
000004  013702  001242'                 MOV     SEED,R2                         ;                               3077
000010  013703  001244'                 MOV     NEXT.RANDNUM,R3
000014  005001                          CLR     R1                              ; COUNT                         3075
000016  010300                  1$:     MOV     R3,R0                           ;                               3077
000020  060200                          ADD     R2,R0
000022  006300                          ASL     R0                              ;                               3076
000024  006300                          ASL     R0
000026  010037  001242'                 MOV     R0,SEED                         ;                               307
000032  062737  000004  001242'         ADD     #4,SEED
000040  010346                          MOV     R3,-(SP)                        ;                               3078
000042  012746  000004                  MOV     #4,-(SP)
000046  004737  000000G                 JSR     PC,BL$DIV
000052  013702  001242'                 MOV     SEED,R2
000056  060200                          ADD     R2,R0
000060  010037  001244'                 MOV     R0,NEXT.RANDNUM
000064  010003                          MOV     R0,R3                           ; NEXT.RANDNUM,*                3079
000066  010361  000116'                 MOV     R3,RANDOM(R1)                   ; *,*(COUNT)
000072  022626                          CMP     (SP)+,(SP)+                     ;                               3076
000074  062701  000002                  ADD     #2,R1                           ; *,COUNT                       3075
000100  020127  000036                  CMP     R1,#36                          ; COUNT,*
000104  003744                          BLE     1$
000106  000207                          RTS     PC                              ;                               3059

; Routine Size:  36 words,      Routine Base:  $CODE$ + 7000
; Maximum stack depth per invocation:  7 words
```

```
;          3083  routine QIO_UNIT : novalue =
;          3084
;          3085  !+
;          3086  !       THIS ROUTINE IS CALLED BY QIO_GEN TO RANDOMLY SELECT ONE UNIT
;          3087  !       CONFIGURED UNDER THE CURRENT CONTROLLER (CCTLR) TO BE USED FOR THE
;          3088  !       CURRENT QIO OR QIO PAIR. THE UNIT SELECTED IS BASED ON THE NUMBER OF
;          3089  !       UNITS ELIGIBLE TO RECEIVE AN I/O REQUEST (FROM 1 TO 4) AND THE FIRST
;          3090  !       RANDOM NUMBER IN THE RANDOM NUMBER TABLE (RANDOM).
;          3091  !
;          3092  !       IMPLICIT INPUTS:
;          3093  !               CST_ADDR   ADDRESS OF CURRENT CONTROLLER'S CST
;          3094  !
;          3095  !       IMPLICIT OUTPUTS:
;          3096  !               THE RD/RX DISK NUMBER (DISK ADDRESS) IS LOADED INTO THE
;          3097  !                       APPROPRIATE FIELD OF BOTH MSCP PACKETS.
;          3098  !
;          3099
;          3100      begin
;          3101      own
;          3102          RAT_COUNT : word initial (0);
;          3103      local
;          3104          MOD_COUNT : byte,
;          3105          TBL_COUNT : byte,
;          3106          SELECT_RD : byte;
;          3107
;          3108  !
;          3109  ! THE UNITS WILL BE SELECTED ON AN ADJUSTABLE RATIO, RD51 TO RX50,
;          3110  ! SELECTED VIA THE SOFTWARE PARAMETERS
;          3111  !
;          3112  ! THIS MODE IS FOR SELECTING DEVICES ON THE FOLLOWING SCHEME:
;          3113  ! CHOOSE A DEVICE AND KEEP IT SELECTED FOR A CONSTANT TIME, THEN
;          3114  ! MOVE TO THE NEXT. THIS IS NON-RANDOM, FIXED SEQUENTIAL OPERATIONAL
;          3115  ! MODE
;          3116  !
;          3117
;          3118      if ((.SWP_FLAGS and SWF_RDM) neq SWF_RDM) and       ! NOT RANDOM MODE
;          3119         ((.SWP_FLAGS and SWF_SEQ) neq SWF_SEQ)           ! NOT RANDOM SEQUEUNTIAL MODE
;          3120      then
;          3121
;          3122          if (.BST_CNT neq 0) and
;          3123             (.CST_ADDR [.BST_DEV, D_PRES] eql PRESENT) and
;          3124             (.CST_ADDR [.BST_DEV, D_STAT] eql ONLINE) and
;          3125             (not .CST_ADDR [.BST_DEV, D_FATAL])
;          3126          then
;          3127              begin                                       ! ALREADY WITHIN DEVICE
;          3128              BST_CNT = .BST_CNT - 1;
;          3129              SET_UPAR (.BST_DEV);
;          3130              MAD1 [DK_NUM] = .CDISK;
;          3131              MAD2 [DK_NUM] = .CDISK;
;          3132              return;
;          3133              end
;          3134          else
;          3135              begin                                       ! GET NEW DEVICE
;          3136
;          3137              incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;          3138                  begin
```

```
;          3139
;          3140                    f (.BST_DEV eql 0) or
;          3141                      (.BST_DEV eql (3 * UNIT_SIZE + OF_UN))
;          3142                    then
;          3143                        BST_DEV = OF_UN
;          3144                    else
;          3145                        BST_DEV = .BST_DEV + UNIT_SIZE;
;          3146
;          3147                    if (.CST_ADDR [.BST_DEV, D_PRES] eql PRESENT) and
;          3148                       (.CST_ADDR [.BST_DEV, D_STAT] eql ONLINE) and
;          3149                       (not .CST_ADDR [.BST_DEV, D_FATAL])
;          3150                    then
;          3151                        begin
;          3152
;          3153                        if .CST_ADDR [.BST_DEV, D_TYPE] eql RX_50
;          3154                        then
;          3155                            BST_CNT = RX_SEQ_CNT
;          3156                        else
;          3157                            BST_CNT = RD_SEQ_CNT;
;          3158
;          3159                        SET_UPAR (.BST_DEV);
;          3160                        MAD1 [DK_NUM] = .CDISK;
;          3161                        MAD2 [DK_NUM] = .CDISK;
;          3162                        return;
;          3163                        end;
;          3164
;          3165                    end;
;          3166
;          3167                end;
;          3168
;          3169    !
;          3170    ! RANDOM SELECTION OF DRIVES
;          3171    !
;          3172    !       this part selects the device by the ratio
;          3173
;          3174    rat_count = .rat_count + 1;                         ! increment counter from 0 to 100
;          3175    if (.rat_count geq 100) then rat_count = 0;         ! in case counter gets to large reinit to 0
;          3176    if (.rat_count lss .swp_rat)
;          3177    then
;          3178        SELECT_RD = true                                ! if counter greater than swap ratio then do a rx-50
;          3179    else                                                ! and reiniate the counter
;          3180        SELECT_RD = false;                              ! if counter less than ratio do a rd-51
;          3181
;          3182    if (100 eql .swp_rat)                               ! if ratio equal 100 do rd
;          3183    then
;          3184        SELECT_RD = true;
;          3185
;          3186    if (0 eql .swp_rat)                                 ! if ratio equals 0 do rx
;          3187    then
;          3188        SELECT_RD = FALSE;
;          3189
;          3190    !
;          3191    ! IF RD51s SELECTED
;          3192    !
;          3193    ! COUNT NUMBER OF RD51s AVAILABLE
;          3194    !
```

```
;          3195
;          3196      if .SELECT RD
;          3197      then
;          3198          begin
;          3199          MOD_COUNT = 0;                                  ! COUNT THE NUMBER OF RDs IN THE SYSTEM
;          3200
;          3201          incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT SIZE do
;          3202
;          3203              if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and
;          3204                 (.CST_ADDR [.OFFSET, D_STAT] eql ONLINE) and
;          3205                 (.CST_ADDR [.OFFSET, D_TYPE] eql RD_51) and
;          3206                 (not .CST_ADDR [.OFFSET, D FATAL])
;          3207              then
;          3208                  begin
;          3209                  STORAGE [.MOD_COUNT] = .OFFSET;
;          3210                  MOD_COUNT = .MOD COUNT + 1;
;          3211                  end;
;          3212
;          3213  !
;          3214  ! SELECT ON OF THE RD51s
;          3215  !
;          3216
;          3217          if .MOD_COUNT neq 0                             ! IF AT LEAST ON RD51 PRESENT
;          3218          then
;          3219              begin
;          3220              TBL_COUNT = 0;
;          3221
;          3222              do
;          3223                  begin
;          3224                  SET_UPAR (.STORAGE [(.RANDOM [.TBL_COUNT] and %o'077777') mod .MOD COUNT]);
;          3225                  TBL_COUNT = .TBL_COUNT + 1;
;          3226                  end
;          3227              until ((.CST_ADDR [.CUOFF, D_PRES] eql PRESENT) and
;          3228                     (.CST_ADDR [.CUOFF, D_STAT] eql ONLINE) and
;          3229                     (not .CST_ADDR [.CUOFF, D_FATAL])) or
;          3230                    (.TBL_COUNT eql RDM_LEN);
;          3231
;          3232              MAD1 [DK_NUM] = .CDISK;
;          3233              MAD2 [DK_NUM] = .CDISK;
;          3234              return;
;          3235              end;
;          3236
;          3237          end;
;          3238
;          3239  !
;          3240  ! IF NO RD51 SELECTED, SELECT AN RX50
;          3241  !
;          3242  ! COUNT THE NUMBER OF RX50s
;          3243  !
;          3244
;          3245      MOD_COUNT = 0;
;          3246
;          3247      incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;          3248
;          3249          if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and
;          3250             (.CST_ADDR [.OFFSET, D_STAT] eql ONLINE) and
```

```
;          3251              (.CST_ADDR [.OFFSET, D_TYPE] eql RX_50) and
;          3252              (not .CST_ADDR [.OFFSET, D_FATAL])
;          3253          then
;          3254              begin
;          3255              STORAGE [.MOD_COUNT] = .OFFSET;
;          3256              MOD_COUNT = .MOD_COUNT + 1;
;          3257              end;
;          3258
;          3259  !
;          3260  ! AND CHOOSE ONE!
;          3261  !
;          3262
;          3263      if .MOD_COUNT neq 0
;          3264      then
;          3265          begin
;          3266          TBL_COUNT = 0;
;          3267
;          3268          do
;          3269              begin
;          3270              SET_UPAR (.STORAGE [(.RANDOM [.TBL_COUNT] and %o'077777') mod .MOD_COUNT]);
;          3271              TBL_COUNT = .TBL_COUNT + 1;
;          3272              end
;          3273          until ((.CST_ADDR [.CUOFF, D_PRES] eql PRESENT) and
;          3274                 (.CST_ADDR [.CUOFF, D_STAT] eql ONLINE) and
;          3275                 (not .CST_ADDR [.CUOFF, D_FATAL])) or
;          3276                (.TBL_COUNT eql RDM_LEN);
;          3277
;          3278          MAD1 [DK_NUM] = .CDISK;
;          3279          MAD2 [DK_NUM] = .CDISK;
;          3280          return;
;          3281          end;
;          3282
;          3283  !
;          3284  ! IF NO UNIT SELECTED SO FAR BY ABOVE METHOD, SELECT ANY ONE AT RANDOM
;          3285  !
;          3286  ! COUNT ALL UNITS AVAILABLE
;          3287  !
;          3288
;          3289      MOD_COUNT = 0;
;          3290
;          3291      incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do
;          3292
;          3293          if (.CST_ADDR [.OFFSET, D_PRES] eql PRESENT) and
;          3294             (.CST_ADDR [.OFFSET, D_STAT] eql ONLINE) and
;          3295             (not .CST_ADDR [.OFFSET, D_FATAL])
;          3296          then
;          3297              begin
;          3298              STORAGE [.MOD_COUNT] = .OFFSET;
;          3299              MOD_COUNT = .MOD_COUNT + 1;
;          3300              end;
;          3301
;          3302  !
;          3303  ! SELECT ANY ONE ONE UNIT AT RANDOM
;          3304  !
;          3305      if .MOD_CO  T neq 0
;          3306      then
```

```
;          3307          begin
;          3308          TBL_COUNT = 0;
;          3309
;          3310          do
;          3311              begin
;          3312              SET_UPAR (.STORAGE [(.RANDOM [.TBL_COUNT] and %o'077777') mod .MOD_COUNT]);
;          3313              TBL_COUNT = .TBL_COUNT + 1;
;          3314              end
;          3315          until ((.CST_ADDR [.CUOFF, D_PRES] eql PRESENT) and
;          3316                 (.CST_ADDR [.CUOFF, D_STAT] eql ONLINE) and
;          3317                 (not .CST_ADDR [.CUOFF, D_FATAL])) or
;          3318                (.TBL_COUNT eql RDM_LEN);
;          3319
;          3320          MAD1 [DK_NUM] = .CDISK;
;          3321          MAD2 [DK_NUM] = .CDISK;
;          3322          return
;          3323          end
;          3324
;          3325      !
;          3326      ! DECLARE END-OF-PASS IF NO UNIT ONLINE
;          3327      !
;          3328
;          3329          else
;          3330              begin
;          3331              EOP_FLAG = TRUE;
;          3332              DCT_ADDR [IG_INT] = TRUE;
;          3333              end;
;          3334
;          3335          end;                                    ! ROUTINE QIO_UNIT


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


                                        .SBTTL  QIO.UNIT MULTI-DRIVE TEST ROUTINES
007110                                  .PSECT  $CODE$,  RO

000000  004137  000000G       QIO.UNIT:
                                        JSR     R1,$SAVE3                       ;  3083
000004  032737  000002  000000G        BIT     #2,SWP.FLAGS                    ;  3118
000012  001136                          BNE     9$
000014  032737  001000  000000G        BIT     #1000,SWP.FLAGS                 ;  3119
000022  001132                          BNE     9$
000024  005737  001230'               TST     BST.CNT                         ;  3122
000030  001433                          BEQ     1$
000032  013700  001232'               MOV     BST.DEV,R0                      ;  3123
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                      ;  3124
000056  006300                          ASL     R0
000060  063700  000000G               ADD     CST.ADDR,R0
```

```
000064  032710  020000                          BIT     #020000,(R0)
000070  001413                                  BEQ     1$
000072  013700  001232                          MOV     BST.DEV,R0              ;                                       3127
000076  006300                                  ASL     R0
000100  063700  000000G                         ADD     CST.ADDR,R0
000104  032710  010000                          BIT     #10000,(R0)
000110  001003                                  BNE     1$
000112  005537  001230                          DEC     BST.CNT                 ;                                       3128
000116  000447                                  BR      7$                      ;                                       3129
000120  012701  000005              1$:         MOV     #5,R1                   ; #,OFFSET                              3131
000124  013700  001232              2$:         MOV     BST.DEV,R0              ;                                       3140
000130  001405                                  BEQ     3$
000132  020027  000022                          CMP     R0,#22                  ;                                       3141
000136  001004                                  BNE     4$
000140  012737  000003  001232      3$:         MOV     #3,BST.DEV              ;                                       3143
000146  000405                                  BR      5$                      ;                                       3140
000150  062737  000005  001232      4$:         ADD     #5,BST.DEV              ;                                       3145
000156  013700  001232              5$:         MOV     BST.DEV,R0              ;                                       3147
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)             ;                                       3148
000202  001455                                  BEQ     8$
000204  032710  010000                          BIT     #10000,(R0)             ;                                       3149
000210  001032                                  BNE     8$
000212  132710  000004                          BITB    #4,(R0)                 ;                                       3153
000216  001004                                  BNE     6$
000220  012737  000120  001230                  MOV     #120,BST.CNT            ;                                       3155
000226  000405                                  BR      7$                      ;                                       3153
000230  012737  004160  001230      6$:         MOV     #4160,BST.CNT           ;                                       3157
000236  013746  001232              7$:         MOV     BST.DEV,-(SP)           ;                                       3159
000242  004737  000000G                         JSR     PC,SET.UPAR
000246  013700  000164'                         MOV     MAD1,R0                 ;                                       3160
000252  013760  000000G 000016                 MOV     CDISK,16(R0)
000260  013700  000166'                         MOV     MAD2,R0                 ;                                       3161
000264  013760  000000G 000016                 MOV     CDISK,16(R0)
000272  005726                                  TST     (SP)+                   ;                                       3147
000274  000207                                  RTS     PC                      ;                                       3151
000276  062701  000005              8$:         ADD     #5,R1                   ; #,OFFSET                              3137
000302  020127  000022                          CMP     R1,#22                  ; OFFSET,#
000306  003706                                  BLE     2$
000310  005237  001246'              9$:         INC     RAT.COUNT               ;                                       3174
000314  023727  001246' 000144                 CMP     RAT.COUNT,#144          ;                                       3175
000322  002402                                  BLT     10$
000324  005037  001246                          CLR     RAT.COUNT
000330  013701  000000G             10$:        MOV     SWP.RAT,R1              ;                                       3176
000334  023701  001246'                         CMP     RAT.COUNT,R1
000340  002003                                  BGE     11$
000342  112700  000001                          MOVB    #1,R0                   ; #,SELECT.RD                           3178
000346  000401                                  BR      12$                     ;                                       3176
000350  105000              11$:        CLRB    R0                      ; SELECT.RD                             3180
000352  020127  000144          12$:        CMP     R1,#144                 ;                                       3182
000356  001002                                  BNE     13$
000360  112700  000001                          MOVB    #1,R0                   ; #,SELECT.RD                           3184
000364  005701              13$:        TST     R1                      ;                                       3186
```

```
000366  001001                          BNE     14$
000370  105000                          CLRB    R0                      ; SELECT.RD                     3188
000372  006000                  14$:    ROR     R0                      ; SELECT.RD                     3196
000374  103105                          BCC     19$
000376  105003                          CLRB    R3                      ; MOD.COUNT                     3199
000400  012701  000003                  MOV     #3,R1                   ; *,OFFSET                      3201
000404  010100                  15$:    MOV     R1,R0                   ; OFFSET,*                      3203
000406  006300                          ASL     R0
000410  063700  000000G                 ADD     CST.ADDR,R0
000414  032710  040000                  BIT     #40000,(R0)
000420  001417                          BEQ     16$
000422  032710  020000                  BIT     #20000,(R0)             ;                               3204
000426  001414                          BEQ     16$
000430  132710  000004                  BITB    #4,(R0)                 ;                               3205
000434  001411                          BEQ     16$
000436  032710  010000                  BIT     #10000,(R0)             ;                               3206
000442  001006                          BNE     16$
000444  005000                          CLR     R0                      ;                               3209
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                     3210
000460  062701  000005          16$:    ADD     #5,R1                   ; *,OFFSET                      3201
000464  020127  000022                  CMP     R1,#22                  ; OFFSET,*
000470  003745                          BLE     15$
000472  105703                          TSTB    R3                      ; MOD.COUNT                     3217
000474  001445                          BEQ     19$
000476  105002                          CLRB    R2                      ; TBL.COUNT                     3220
000500  005000                  17$:    CLR     R0                      ;                               3224
000502  150200                          BISB    R2,R0                   ; TBL.COUNT,*
000504  006300                          ASL     R0
000506  016046  000116'                 MOV     RANDOM(R0),-(SP)
000512  042716  100000                  BIC     #100000,(SP)
000516  005046                          CLR     -(SP)
000520  110316                          MOVB    R3,(SP)                 ; MOD.COUNT,*
000522  004737  000000G                 JSR     PC,BL$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                     3225
000542  022626                          CMP     (SP)+,(SP)+             ;                               3223
000544  013700  000000G                 MOV     CUOFF,R0                ;                               3227
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)             ;                               3228
000570  001403                          BEQ     18$
000572  032710  010000                  BIT     #10000,(R0)             ;                               3229
000576  001510                          BEQ     24$
000600  120227  000020          18$:    CMPB    R2,#20                  ; TBL.COUNT,*                   3230
000604  001335                          BNE     17$
000606  000504                          BR      24$                     ;                               3232
000610  105003                  19$:    CLRB    R3                      ; MOD.COUNT                     3245
000612  012701  000003                  MOV     #3,R1                   ; *,OFFSET                      3247
000616  010100                  20$:    MOV     R1,R0                   ; OFFSET,*                      3249
```

```
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)             ;                       3250
000640  001414                          BEQ     21$
000642  132710  000004                  BITB    #4,(R0)                 ;                       3251
000646  001011                          BNE     21$
000650  032710  010000                  BIT     #10000,(R0)             ;                       3252
000654  001006                          BNE     21$
000656  005000                          CLR     R0                      ;                       3255
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             3256
000672  062701  000005          21$:    ADD     #5,R1                   ; *,OFFSET              3247
000676  020127  000022                  CMP     R1,#22                  ; OFFSET,*
000702  003745                          BLE     20$
000704  105703                          TSTB    R3                      ; MOD.COUNT             3263
000706  001445                          BEQ     25$
000710  105002                          CLRB    R2                      ; TBL.COUNT             3266
000712  005000                  22$:    CLR     R0                      ;                       3270
000714  150200                          BISB    R2,R0                   ; TBL.COUNT,*
000716  006300                          ASL     R0
000720  016046  000116'                 MOV     RANDOM(R0),-(SP)
000724  042716  100000                  BIC     #100000,(SP)
000730  005046                          CLR     -(SP)
000732  110316                          MOVB    R3,(SP)                 ; MOD.COUNT,*
000734  004737  000000G                 JSR     PC,BL$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             5271
000754  022626                          CMP     (SP)+,(SP)+             ;                       3269
000756  013700  000000G                 MOV     CUOFF,R0                ;                       3273
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)             ;                       3274
001002  001403                          BEQ     23$
001004  032710  010000                  BIT     #10000,(R0)             ;                       3275
001010  001505                          BEQ     30$
001012  120227  000020          23$:    CMPB    R2,#20                  ; TBL.COUNT,*           3276
001016  001335                          BNE     22$
001020  000501                  24$:    BR      30$                     ;                       3278
001022  105003                  25$:    CLRB    R3                      ; MOD.COUNT             3289
001024  012701  000003                  MOV     #3,R1                   ; *,OFFSET              3291
001030  010100                  26$:    MOV     R1,R0                   ; OFFSET,*              3293
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)             ;                       3294
001052  001411                          BEQ     27$
001054  032710  010000                  BIT     #10000,(R0)             ;                       3295
```

```
001060  001006                          BNE     27$
001062  005000                          CLR     R0                      ;                               3298
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                     3299
001076  062701  000005          27$:    ADD     #5,R1                   ; +,OFFSET                      3291
001102  020127  000022                  CMP     R1,#22                  ; OFFSET,+
001106  003750                          BLE     26$
001110  105703                          TSTB    R3                      ; MOD.COUNT                     3305
001112  001457                          BEQ     31$
001114  105002                          CLRB    R2                      ; TBL.COUNT                     3308
001116  005000          28$:    CLR     R0                      ;                               3312
001120  150200                          BISB    R2,R0                   ; TBL.COUNT,+
001122  006300                          ASL     R0
001124  016046  000116'                 MOV     RANDOM(R0),-(SP)
001130  042716  100000                  BIC     #100000,(SP)
001134  005046                          CLR     -(SP)
001136  110316                          MOVB    R3,(SP)                 ; MOD.COUNT,+
001140  004737  000000G                 JSR     PC,BL$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                     3313
001160  022626                          CMP     (SP)+,(SP)+             ;                               3311
001162  013700  000000G                 MOV     CUOFF,R0                ;                               3315
001166  006300                          ASL     R0
001170  063700  000000G                 ADD     CST.ADDR,R0
001174  052710  040000                  BIT     #40000,(R0)
001200  001406                          BEQ     29$
001202  032710  020000                  BIT     #20000,(R0)             ;                               3316
001206  001403                          BEQ     29$
001210  032710  010000                  BIT     #10000,(R0)             ;                               3317
001214  001403                          BEQ     30$
001216  120227  000020          29$:    CMPB    R2,#20                  ; TBL.COUNT,+                   3318
001222  001335                          BNE     28$
001224  013700  000164'          30$:    MOV     MAD1,R0                 ;                               3320
001230  013760  000000G 000016          MOV     CDISK,16(R0)
001236  013700  000166'                 MOV     MAD2,R0                 ;                               3321
001242  013760  000000G 000016          MOV     CDISK,16(R0)
001250  000207                          RTS     PC                      ;                               3307
001252  112737  000001  000000G  31$:   MOVB    #1,EOP.FLAG             ;                               3331
001260  052777  040000  000000G         BIS     #40000,@DCT.ADDR        ;                               3332
001266  000207                          RTS     PC                      ;                               3083

; Routine Size:  348 words,     Routine Base:  $CODE$ + 7110
; Maximum stack depth per invocation:  7 words
```

```
;         3336  routine QIO_FUNC : novalue =
;         3337
;         3338  !+
;         3339  !       THIS ROUTINE IS CALLED BY QIO_GEN TO SELECT THE I/O FUNCTION (OPCODE)
;         3340  !       TO BE USED FOR THE CURRENT QIO OR QIO PAIR. THE FUNCTION IS DETERMINED
;         3341  !       BY THE FOLLWING ALGORITHM:
;         3342  !
;         3343  !               IF THE CHOSEN UNIT IS PROTECTED
;         3344  !               THEN
;         3345  !                       FUNCTION = READ
;         3346  !               ELSE (UNPROTECTED)
;         3347  !                       FUNCTION (WRITE OR READ) IS BASED ON A RANDOM
;         3348  !                       NUMBER
;         3349  !
;         3350  !       IN ADDITION, IF THE OPERATOR SELECTED THE OPTION OF PERFORMING WRITE
;         3351  !       COMPARES AT THE HOST, AND IF A "WRITE" FUNCTION WAS CHOSEN ABOVE FOR
;         3352  !       THE FIRST QIO, THEN A "READ" OPCODE IS LOADED INTO THE SECOND MSCP
;         3353  !       ENVELOPE. OTHERWISE, THE SECOND MSCP ENVELOPE IS RETURNED TO THE POOL.
;         3354  !
;         3355  !       THIS ROUTINE ALSO DECIDES WHEATHER IT IS TIME TO RUN THE DUP EXERCISER.
;         3356  !       THE EXERCISER WRITES 25 LBNS FOR EVERY 1 DBN  FOR INITIALIZATION
;         3357  !       REASONS THE MULTIPLE OF 25 LBNS CAN BE READ FOR 1 * MULTIPLE OF DBNS.
;         3358  !       DUP EXERCISER WRITES X BLOCKS PER PASS THRU THE DUP ROUTINE.  THERFORE
;         3359  !       THE RATIO OF 1 TO 25 MUST BE MULTIPLIED BY  X FOR A RATIO OF 1 TO
;         3360  !       25 * X.  SO EVERY 25*X LBN'S READ OFF A WINCHESTER UNIT X  DBN'S WILL
;         3361  !       BE READ IF ASKED BY THE USER.  X is represented by the VARIABLE "DUPROUND".
;         3362  !       DUPROUND MAY BE CHANGED IN THE SOFTWARE QUESTION GIVEN AT THE START OF
;         3363  !       RUNNING THE EXERCISER PROGARM.  THE DUP EXERCISER THEN REINITIALIZES
;         3364  !       THE CONTROLLER AND CONTINUES AS IF THE REGULAR MSCP EXERCISER WAS
;         3365  !       NEVER INTERRUPTED.  NOTE THAT THE REINITIALIZATION PROCESSES TAKES
;         3366  !       A COUPLE OF SECONDS WHICH ONCE ADDED UP AFTER A COUPLE MILLION READS
;         3367  !       CAN PROVE TO BE QUITE TIMELY.  THERFORE I SUGGEST THAT IF A LARGE AMOUNT OF
;         3368  !       I/O TRANFERS ARE TO BE DONE THE "DUPROUND" VARIABLE BE RAISED TO SPEED UP
;         3369  !       UP THE PROCESS.
;         3370  !
;         3371  !       IMPLICIT INPUTS:
;         3372  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;         3373  !               CUOFF - CURRENT UNIT CST OFFSET
;         3374  !
;         3375  !       IMPLICIT OUTPUTS:
;         3376  !               THE OPCODE FIELD OF ONE OR BOTH MSCP ENVELOPES IS LOADED.
;         3377  !-
;         3378
;         3379      begin
;         3380
;         3381      local
;         3382          FUNC : word;                                ! OPCODE (READ OR WRITE)
;         3383  !PRINTX (DER9);
;         3384  !PRINTF (DER9);
;         3385  !*
;         3386      IF ((.CST_ADDR [.CUOFF + 4,D_COUNT] LEQ 0) AND              ! IF MSCP FUNC CNT EQUAL TO 0
;         3387          (.CST_ADDR [.CUOFF,D_TYPE] EQL RD_51) AND               !IF WINCHESTER DISK
;         3388          (.CST_ADDR [.CUOFF + 3, NODUPMEDIA] NEQ 1))             !IF NODUPMEDIA FLAG BIT IS CLEAR
;         3389          THEN
;         3390  BEGIN
;         3391
```

```
;          3392                    PUT_PKT (.MX2);                         ! RETURN 1ST ENVELOPE TO POOL
;          3393                    MX2 = -1;                                       ! INDICATE FAILURE
;          3394                    CST_ADDR [.CUOFF + 3, DUPERROR] = 0;            !CLEAR DUP ERROR FLAG
;          3395                    DUP ();                                 ! DO DUP TEST
;          3396                    CST_ADDR [.CUOFF + 3, DUPERROR] = 0;            !CLEAR DUP ERROR FLAG
;          3397                    !PRINTX (DBM111);                       ! PRINT MSCP EXERCISER REINIT
;          3398                    CST_ADDR [.CUOFF + 4, D_COUNT]  = (25 * .duproun);      ! REINITIALIZE MSCP FUNC COUNTER
;          3399
;          3400    ! ********************************* THIS SECTION REINTIALIZES 2 ENVOLOPES SO THE MSCP EXERCISER CAN
;          3401    ! ********************************* PROCEED AS BEFORE THE DUP EXERCISER STARTED ********************
;          3402
;          3403    DUP_FLAGS = .DUP_FLAGS OR SWP_DINT;             ! SET DUP INI  FLAG
;          3404    INIT_TEST ();                                   ! THIS REINITIALIZES THE CONTROLLER FOR MSCP MODE
;          3405    DUP_FLAGS = .DUP_FLAGS AND (NOT SWP_DINT);      ! CLEAR DUP INIT FLAG
;          3406
;          3407    MX2 = -1;                                       ! ASSUME FAILURE IN SECURING 2ND ENVELOPE
;          3408    IF (MX1 = GET_PKT (.CCTLR)) LSS 0               ! TRY TO GET 1ST ENVELOPE.
;          3409            OR (.EOP_FLAG)                          ! IF FAILURE
;          3410    THEN    RETURN;                                 ! NO POINT IN CONTINUING
;          3411    IF (MX2 = GET_PKT (.CCTLR)) LSS 0               ! TRY TO GET 2ND ENVELOPE.
;          3412            OR (.EOP_FLAG)                          ! IF FAILURE
;          3413    THEN    BEGIN
;          3414            PUT_PKT (.MX1);                         ! RETURN 1ST ENVELOPE TO POOL
;          3415            MX1 = -1;                                       ! INDICATE FAILURE
;          3416            RETURN;                                 ! DONE
;          3417            END;
;          3418
;          3419    MAD1 = MSCP_PKT + (.MX1 * PKT_LEN * 2);         ! CALCULATE STARTING ADDRESSES
;          3420    MAD2 = MSCP_PKT + (.MX2 * PKT_LEN * 2);         !     OF BOTH ENVELOPES
;          3421    GET_RANDOM ();                                  ! GENERATE A SET OF RANDOM NUMBERS
;          3422    QIO_UNIT ();                                    ! LOAD RANDOM UNIT NUMBER INTO ENVELOPES
;          3423
;          3424    END;
;          3425
;          3426    !*****************************************************************************
;          3427    !                       START OF ROUTINE MSCP
;          3428    !*****************************************************************************
;          3429
;          3430    CST_ADDR [.CUOFF + 4, D_COUNT]  = .CST_ADDR [.CUOFF + 4, D_COUNT]  - 1;                 ! DECREMENT MSCP FUN
( COUNTER
;          3431
;          3432        MAD2 [OPCODE] = 0;                          ! ASSUME 2ND PACKET NOT NEEDED
;          3433
;          3434        if .CST_ADDR [.CUOFF, D_PROT] eql PROTECTED ! IF UNIT IS PROTECTED
;          3435        then
;          3436            FUNC = OP_RD                            ! SET FUNCTION TO READ
;          3437        else
;          3438
;          3439            if (.RANDOM [1] and 1)                  ! USE 2ND RANDOM NUMBER TO SELECT
;          3440            then
;          3441                FUNC = OP_RD                        !     READ
;          3442            else
;          3443                FUNC = OP_WRT;                      !     WRITE
;          3444
;          3445        if (MAD1 [OPCODE] = .FUNC) eql OP_WRT       ! LOAD CHOSEN OPCODE. IF WRITE
;          3446        then
;          3447            begin
```

```
;          3448          MAD1 [CMD_TYPE] = NON_SEQ_CMD;          ! NON SEQUENTIAL COMMAND
;          3449
;          3450          if BIT_TST (SWP_FLAGS, SWF_CWC)         ! IF CONTROLLER DOES WRITE COMPARES
;          3451          then
;          3452              MAD1 [MODIFY] = MD_CMP;             ! ADD COMPARE MODIFIER
;          3453
;          3454          if BIT_TST (SWP_FLAGS, SWF_HWC)         ! IF HOST DOES WRITE COMPARES
;          3455          then
;          3456              begin
;          3457              MAD2 [OPCODE] = OP_RD;              ! SET READ OPCODE INTO 2ND MSCP PACKET
;          3458              MAD2 [CMD_TYPE] = NON_SEQ_CMD;      ! NON SEQUENTIAL COMMAND
;          3459              end;
;          3460
;          3461          end
;          3462      else
;          3463          begin
;          3464          MAD1 [CMD_TYPE] = NON_SEQ_CMD;          ! NON-SEQUENTIAL COMMAND
;          3465
;          3466          if BIT_TST (SWP_FLAGS, SWF_CRC)         ! IF CONTROLLER DOES READ COMPARES   FUNCTION IS READ
;          3467          then
;          3468              MAD1 [MODIFY] = MD_CMP;             ! ADD COMPARE MODIFIER
;          3469
;          3470          end;
;          3471
;          3472      if .MAD2 [OPCODE] eql 0                     ! IF NO OPCODE IN 2ND PACKET
;          3473      then
;          3474          begin
;          3475          PUT_PKT (.MX2);                         ! RETURN 2ND PACKET TO POOL
;          3476          MX2 = -1;                               ! MARK IT UNUSED
;          3477          end;
;          3478
;          3479      end;                                        ! ROUTINE QIO_FUNC

                                .SBTTL  QIO.FUNC MULTI-DRIVE TEST ROUTINES
000000  004137  000000G    QIO.FUNC:
                            JSR     R1,$SAVE3                   ;                   3336
000004  013702  000000G             MOV     CST.ADDR,R2         ;                   3386
000010  013701  000000G             MOV     CUOFF,R1
000014  010100                      MOV     R1,R0
000016  006300                      ASL     R0
000020  060200                      ADD     R2,R0
000022  005760  000010              TST     10(R0)
000026  003166                      BGT     4$
000030  010100                      MOV     R1,R0               ;                   3387
000032  006300                      ASL     R0
000034  060200                      ADD     R2,R0
000036  132710  000004              BITB    #4,(R0)
000042  001560                      BEQ     4$
000044  010100                      MOV     R1,R0               ;                   3388
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)           ;                   3392
000064  004737  000000G             JSR     PC,PUT.PKT
```

```
000070  012737  177777  000162'          MOV     #-1,MX2                         ;                       3393
000076  013700  000000G                  MOV     CUOFF,R0                        ;                       3394
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                          ;                       3395
000122  013700  000000G                  MOV     CUOFF,R0                        ;                       3396
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                        ;                       3398
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$MUL
000170  010061  000010                   MOV     R0,10(R1)
000174  052737  000002  000000G          BIS     #2,DUP.FLAGS                    ;                       3403
000202  004737  000212'                  JSR     PC,INIT.TEST                    ;                       3404
000206  042737  000002  000000G          BIC     #2,DUP.FLAGS                    ;                       3405
000214  012737  177777  000162'          MOV     #-1,MX2                         ;                       3407
000222  013716  000000G                  MOV     CCTLR,(SP)                      ;                       3408
000226  004737  000000G                  JSR     PC,GET.PKT
000232  010037  000160'                  MOV     R0,MX1
000236  002426                           BLT     2$
000240  132737  000001  000000G          BITB    #1,EOP.FLAG                     ;                       3409
000246  001022                           BNE     2$                              ;                       3336
000250  013716  000000G                  MOV     CCTLR,(SP)                      ;                       3411
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                     ;                       3412
000274  001411                           BEQ     3$
000276  013716  000160'          1$:     MOV     MX1,(SP)                        ;                       3414
000302  004737  000000G                  JSR     PC,PUT.PKT
000306  012737  177777  000160'          MOV     #-1,MX1                         ;                       3415
000314  022626                   2$:     CMP     (SP)+,(SP)+                     ;                       3411
000316  000207                           RTS     PC                              ;                       3413
000320  013716  000160'          3$:     MOV     MX1,(SP)                        ;                       3419
000324  012746  000104                   MOV     #104,-(SP)
000330  004737  000000G                  JSR     PC,BL$MUL
000334  062700  000000G                  ADD     #MSCP.PKT,R0
000340  010037  000164'                  MOV     R0,MAD1
000344  013716  000162'                  MOV     MX2,(SP)                        ;                       3420
000350  012746  000104                   MOV     #104,-(SP)
000354  004737  000000G                  JSR     PC,BL$MUL
000360  062700  000000G                  ADD     #MSCP.PKT,R0
000364  010037  000166'                  MOV     R0,MAD2
000370  004737  007000'                  JSR     PC,GET.RANDOM                   ;                       3421
000374  004737  007110'                  JSR     PC,QIO.UNIT                     ;                       3422
000400  062706  000010                   ADD     #10,SP                          ;                       3390
000404  013700  000000G          4$:     MOV     CUOFF,R0                        ;                       3430
000410  006300                           ASL     R0
000412  063700  000000G                  ADD     CST.ADDR,R0
000416  005360  000010                   DEC     10(R0)
000422  013700  000166'                  MOV     MAD2,R0                         ;                       3432
```

```
000426  012703  000022                        MOV     #22,R3
000432  060003                                ADD     R0,R3
000434  105013                                CLRB    (R3)
000436  013700  000000G                       MOV     CUOFF,R0                ;                       3434
000442  006300                                ASL     R0
000444  063700  000000G                       ADD     CST.ADDR,R0
000450  032710  100000                        BIT     #100000,(R0)
000454  001404                                BEQ     5$                      ;                       3436
000456  032737  000001  000120'               BIT     #1,RANDOM+2             ;                       3439
000464  001403                                BEQ     6$
000466  012701  000041              5$:       MOV     #41,R1                  ; *,FUNC                3441
000472  000402                                BR      7$                      ;                       3439
000474  012701  000042              6$:       MOV     #42,R1                  ; *,FUNC                3443
000500  013700  000164'              7$:       MOV     MAD1,R0                 ;                       3445
000504  013702  000000G                       MOV     SWP.FLAGS,R2            ;                       3450
000510  110160  000022                        MOVB    R1,22(R0)               ; FUNC,*                3445
000514  020127  000042                        CMP     R1,#42                  ; FUNC,*
000520  001024                                BNE     9$
000522  012760  000002  000004                MOV     #2,4(R0)                ;                       3448
000530  032702  000020                        BIT     #20,R2                  ;                       3450
000534  001403                                BEQ     8$
000536  012760  040000  000024                MOV     #40000,24(R0)           ;                       3452
000544  032702  000040              8$:       BIT     #40,R2                  ;                       3454
000550  001421                                BEQ     10$
000552  112713  000041                        MOVB    #41,(R3)                ;                       3457
000556  013701  000166'                       MOV     MAD2,R1                 ;                       3458
000562  012761  000002  000004                MOV     #2,4(R1)
000570  000411                                BR      10$                     ;                       3445
000572  012760  000002  000004      9$:       MOV     #2,4(R0)                ;                       3464
000600  032702  000004                        BIT     #4,R2                   ;                       3466
000604  001403                                BEQ     10$
000606  012760  040000  000024                MOV     #40000,24(R0)           ;                       3468
000614  105713                      10$:      TSTB    (R3)                    ;                       3472
000616  001010                                BNE     11$
000620  013746  000162'                       MOV     MX2,-(SP)               ;                       3475
000624  004737  000000G                       JSR     PC,PUT.PKT
000630  012737  177777  000162'               MOV     #-1,MX2                 ;                       3476
000636  005726                                TST     (SP)+                   ;                       3474
000640  000207                      11$:      RTS     PC                      ;                       3336

; Routine Size:  209 words,    Routine Base:  $CODE$ + 10400
; Maximum stack depth per invocation:  9 words
```

```
;        3480  !+
;        3481  ROUTINE DUP : NOVALUE =
;        3482
;        3483  !-
;        3484  !       THIS ROUTINE IS CALLED BY QIO_FUNC AFTER 25 = "DUPROUND" RD/WTS .
;        3485  !       THIS EXERCISER WAS PLACED IN THE MIDDLE OF THE MSCP EXERCISER SO A
;        3486  !       COMMON INITIALIZATION AND OTHER ROUTINES COULD BE USED.
;        3487  !               THE DUP EXCERSIZER WILL RUN A READ ONLY OR A WRITE/READ COMPARE
;        3488  !       TO THE DIAGONOSTIC BLOCKS OR DBN'S.  IT WILL RECORD
;        3489  !       THE STATICS IN THE TALLY TABLES.
;        3490  !
;        3491  !       THE PROGRAM USES CONTROLLER LOCAL PROGRAMS TO WRITE AND READ THE DBN S.
;        3492  !       WHEN WRITTING TO THE DBN'S A ONE WORD PATTERN WILL BE SELECTED
;        3493  !       AND COPY TO A 256 WORD BLOCK. THE ROUTINE WILL WRITE TO "DUPROUND' AMOUNT
;        3494  !       OF DBN SEQUENTIAL BLOCKS.  IF A THERE ARE CONTROLLER LOCAL  PROGRAMS AND
;        3495  !       IF THE USER SO DESIRES A WRITE AND  READ OF EACH BLOCK AND A COMPARISON
;        3496  !       TO THE DATA PATTERN WILL BE GIVEN TO TEST FOR FAULTY DBN'S.  IF THE USER
;        3497  !       DOES NOT WANT TO WRITE TO THE DBN'S ONLY A READ WILL BE GIVEN AND NO
;        3498  !       COMPARISON WILL BE DONE.  THE BAD BLOCKS FOUND IN THE COMPARISON TEST
;        3499  !       WILL NOT BE LISTED IN THE RCT TABLES.
;        3500  !
;        3501  !       AFTER THE EXERCISER HAS EXAMINE "DUPROUND" AMOUNT OF  BLOCKS IT WILL
;        3502  !       REINITIATE THE ENVOLOPES SO THAT THE MSCP EXERCISER MAY CONTINUE AS
;        3503  !       BEFORE.
;        3504  !
;        3505  !       IMPLICIT INPUTS:
;        3506  !               CCTLR - CURRENT CONTROLLER NUMBER
;        3507  !               CST_ADDRS  - CONTAINS THE CURRENT CONTROLLER STATUS TABLE
;        3508  !               CUOFF      - CURRENT OFFSET IN CST TABLE FOR PARTICULAR DRIVE
;        3509  !
;        3510  !       IMPLICIT OUTPUTS:
;        3511  !               S_PATTERN - PATTERN BEING WRITTEN TO AND READ FROM DBN'S
;        3512  !-
;        3513  BEGIN
;        3514  OWN
;        3515          TEMP : WORD;
;        3516  !PRINTX (DBM110);
;        3517  !PRINTX (DER10);
;        3518
;        3519  S_PATTERN = .RANDOM [1];
;        3520  IF (.CST_ADDR [.CUOFF + 3, D_DBN] + .duproud) GEQ 144 THEN (CST_ADDR [.CUOFF + 3, D_DBN] = 0);
;        3521                                                  ! TEST TO SEE IF NEXT DBN'S TO LARGE
;        3522                                                  ! CIRCLE ARROUND IF DBN TO LARGE
;        3523  DO
;        3524          BEGIN
;        3525          MSCP_PKT [.MX1, MSGLEN] = SZ_GDS;               ! PACKET SIZE                                   GET DUST STATUS
;        3526          MSCP_PKT [.MX1, OPCODE] = OP_GDS;               ! OPCODE = GET DUST STATUS
;        3527          MSCP_PKT [.MX1, MODIFY] = 0;                    !
;        3528          DUPCOMMAND ();                                  ! SENDS AND RECEIVES THE COMMAND
;        3529          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1
;        3530                  THEN
;        3531                   RETURN;                                !IF DUP ERROR THEN CLR FLAG & RETURN
;        3532          IF .CST_ADDR [.CUOFF + 3, NODUPMEDIA] EQL 1 THEN RETURN; ! IF DUP LOCAL MEDIA NOT THERE THEN RETURN
;        3533          IF (MX1 = GET_PKT (.CCTLR)) LSS 0                       ! TRY TO GET AN ENVELOPE. IF FAILURE
;        3534                  OR (.EOP_FLAG)
;        3535                  THEN RETURN;                                    ! NO POINT IN CONTINUINC
```

```
;          3536          END
;          3537  UNTIL .CST_ADDR [.CUOFF + 3, D ACTIVE] EQL IDLE;
;          3538
;          3539  TEMP = .CST ADDR [.CUOFF + 3, D DBN];
;          3540  INCR DBNCNT FROM (.TEMP + 1) TO (.TEMP + .dupround) DO                  ! INCREMENT FROM RELATIVE DBN TO DBN + dupro
und
.          3541          BEGIN
.          3542
;          3543          IF .CST_ADDR [.CUOFF + 3, DUPWRITE]                          ! IF WRITE FLAG SET IN CST TABLE THEN WRITE DBN S
;          3544           THEN
;          3545                  BEGIN
;          3546                  T_ADDR [T_DBN_WT] = .T_ADDR [T DBN_WT] + 1;
;          3547                  DUPWRTDBN();                                        ! CALL ROUTINE TO HANDLE WRITTING ROUTINES
;          3548                  IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1
;          3549                          THEN
.          3550                           RETURN;                                    !IF DUP ERROR THEN CLR FLAG & RETURN
;          3551                  END;
;          3552          T_ADDR [T_DBN_RD] = .T_ADDR [T_DBN_RD] + 1;
;          3553          DUPREDDBN();                                  ! CALL ROUTINE TO HANDLE READING DBN'S
;          3554          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1
;          3555                  THEN
;          3556                   RETURN;                              !IF DUP ERROR THEN CLR FLAG & RETURN
;          3557
;          3558          END;
;          3559
;          3560  END;
```

```
001250                                  .PSECT  $GGG$,  RO
001250                          TEMP:   .BLKW   1


                                        .SBTTL  DUP MULTI-DRIVE TEST ROUTINES
011242                                  .PSECT  $CODE$,  RO

000000  004137  000000G                DUP:    JSR     R1,$SAVE3                       ;                       3481
000004  013737  000120'  000000G              MOV     RANDOM+2,S.PATTERN              ;                       3519
000012  013700  000000G                        MOV     CUOFF,R0                        ;                       3520
000016  006300                                ASL     R0
000020  063700  000000G                        ADD     CST.ADDR,R0
000024  005001                                CLR     R1
000026  156001  000006                        BISB    6(R0),R1
000032  063701  000000G                        ADD     DUPROUND,R1
000036  020127  000220                        CMP     R1,#220
000042  002402                                BLT     1$
000044  105060  000006                        CLRB    6(R0)
000050  013746  000160'                1$:     MOV     MX1,-(SP)                       ;                       3525
000054  012746  000104                        MOV     #104,-(SP)
000060  004737  000000G                        JSR     PC,BL$MUL
000064  012760  000014  000006G              MOV     #14,MSCP.PKT+6(R0)
000072  112760  000001  000022G              MOVB    #1,MSCP.PKT+22(R0)              ;                       3526
000100  005060  000024G                        CLR     MSCP.PKT+24(R0)                 ;                       3527
000104  004737  000000V                        JSR     PC,DUPCOMMAND                   ;                       3528
000110  013700  000000G                        MOV     CUOFF,R0                        ;                       3529
000114  006300                                ASL     R0
000116  063700  000000G                        ADD     CST.ADDR,R0
```

```
000122  032760  040000  000006          BIT     #40000,6(R0)
000130  001016                          BNE     2$                      ;                       3481
000132  005760  000006                  TST     6(R0)                   ;                       3532
000136  100413                          BMI     2$                      ;                       3481
000140  013716  000000G                 MOV     CCTLR,(SP)              ;                       3533
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             ;                       3534
000164  001402                          BEQ     3$
000166  022626                  2$:     CMP     (SP)+,(SP)+             ;                       3481
000170  000207                          RTS     PC                      ;                       3535
000172  022626                  3$:     CMP     (SP)+,(SP)+             ;                       3524
000174  013700  000000G                 MOV     CUOFF,R0                ;                       3537
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               ;                       3539
000224  105037  001251'                 CLRB    TEMP+1
000230  013702  001250'                 MOV     TEMP,R2                 ;                       3540
000234  063702  000000G                 ADD     DUPROUND,R2
000240  013700  000000G                 MOV     CUOFF,R0                ;                       3543
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              3540
000264  000443                          BR      6$
000266  032713  010000          4$:     BIT     #10000,(R3)             ;                       3543
000272  001417                          BEQ     5$
000274  013700  000000G                 MOV     T.ADDR,R0               ;                       3546
000300  005260  000052                  INC     52(R0)
000304  004737  000000V                 JSR     PC,DUPWRTDBN            ;                       3547
000310  013700  000000G                 MOV     CUOFF,R0                ;                       3548
000314  006300                          ASL     R0
000316  063700  000000G                 ADD     CST.ADDR,R0
000322  032760  040000  000006          BIT     #40000,6(R0)
000330  001024                          BNE     7$                      ;                       3550
000332  013700  000000G          5$:     MOV     T.ADDR,R0               ;                       3552
000336  005260  000056                  INC     56(R0)
000342  004737  000000V                 JSR     PC,DUPREDDBN            ;                       3553
000346  013700  000000G                 MOV     CUOFF,R0                ;                       3554
000352  006300                          ASL     R0
000354  063700  000000G                 ADD     CST.ADDR,R0
000360  010003                          MOV     R0,R3
000362  062703  000006                  ADD     #6,R3
000366  032713  040000                  BIT     #40000,(R3)
000372  001003                          BNE     7$                      ;                       3556
000374  005201                  6$:     INC     R1                      ; DBNCNT                3540
000376  020102                          CMP     R1,R2                   ; DBNCNT,*
000400  003732                          BLE     4$
000402  000207                  7$:     RTS     PC                      ;                       3481
; Routine Size:  130 words,    Routine Base:  $CODE$ + 11242
; Maximum stack depth per invocation:  7 words
```

```
;       3561  ROUTINE DUPWRTDBN : NOVALUE =
;       3562
;       3563  !+
;       3564  ! THIS ROUTINE IS CALLED BY DUP ROUTINE TO USE THE CONTROLLER LOCAL PROGRAM
;       3565  ! "WRTDBN".  TO USE THE PROGRAM THE OPTIONAL DUP SUB-PROTOCAL IS USED TO
;       3566  ! COMMUNICATE WITH THE CONTROLLER.  THE PROGRAM WRITES TO A DIAGONOSTIC BLOCK (DBN)
;       3567  ! THE WORD IN "S_PATTERN" IS WRITTEN TO THE 256 WORDS IN THE DBN.  IF AN ERROR OCCURS
;       3568  ! WHILE RUNNING THE CONTROLLER LOCAL PROGRAM THE ERROR IS USSUALLY REPORTED IN THE
;       3569  ! DUP BUFFER. (EX. ILLEGAL UNIT NUMBER, ILLEGAL BLK #, DEVICE ERROR, ZERO LENGHT MSG)
;       3570  !
;       3571  !       IMPLICIT INPUTS:
;       3572  !               CST_ADDRS    CONTAINS THE CURRENT CONTROLLER STATUS TABLE
;       3573  !               CUOFF       - CURRENT OFFSET IN CST TABLE FOR PARTICULAR DRIVE
;       3574  !               S_PATTERN   - CONTAINS PATTERN WORD!-
;       3575  BEGIN
;       3576
;       3577  !PRINTX (DER11);
;       3578
;       3579  MSCP_PKT [.MX1, MSGLEN] = SZ_ELP;              ! PACKET SIZE                                   EXECUTE LOCAL PROGRAM WRT DB
N
;       3580  MSCP_PKT [.MX1, OPCODE] = OP_ELP;              ! OPCODE = EXECUTE LOCAL PROGRAM
;       3581  MSCP_PKT [.MX1, MODIFY] = 1;                  ! STANDALONE MODIFIER
;       3582  MSCP_PKT [.MX1, L1] =  %ascii'W';             ! FILL IN PROGRAM NAME WITH ASCII LETTERS
;       3583  MSCP_PKT [.MX1, L2] = %ascii'R';
;       3584  MSCP_PKT [.MX1, L3] = %ascii'T';
;       3585  MSCP_PKT [.MX1, L4] = %ascii'D';
.       3586  MSCP_PKT [.MX1, L5] = %ascii'B';
;       3587  MSCP_PKT [.MX1, L6] = %ascii'N';
;       3588  DUPCOMMAND ();                               ! SENDS AND RECEIVES THE COMMAND
;       3589  IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
.       3590  IF (MX1 = GET_PKT (.CCTLR)) LSS 0             ! TRY TO GET AN ENVELOPE. IF FAILURE
.       3591          OR (.EOP_FLAG)
;       3592          THEN
.       3593                  (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
.       3594                   RETURN;);                                  ! NO POINT IN CONTINUING
.       3595
;       3596          MSCP_PKT [.MX1, MSGLEN] = SZ_REC;          ! PACKET SIZE                                RECIEVE DATA
;       3597          MSCP_PKT [.MX1, OPCODE] = OP_RCD;          ! OPCODE = RECEIVE DATA
.       3598          MSCP_PKT [.MX1, BC_LO] = 2;                ! BYTE COUNT TO BE TRANSFERED EQUALS 2
;       3599          MSCP_PKT [.MX1, BUF_0] = DUPPKT;           ! LOAD DESCRIBTOR BUFFER
;       3600          MSCP_PKT [.MX1, MODIFY] = 0;               !
.       3601          DUPCOMMAND ();                             ! SENDS AND RECEIVES THE COMMAND
;       3602          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
;       3603          IF .DUPPKT [DUPTYPE]  NEQ 1
;       3604             OR .DUPPKT [DUPMESG] NEQ 6              ! IF CORRECT RESPONSE
.       3605             OR (MX1 = GET_PKT (.CCTLR)) LSS 0       ! TRY TO GET AN ENVELOPE. IF FAILURE
;       3606             OR (.EOP_FLAG)
;       3607             THEN
.       3608                  (HARD_ERROR ();
;       3609                   CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;       3610                   RETURN;);                                 ! NO POINT IN CONTINUING
.       3611
.       3612          MSCP_PKT [.MX1, MSGLEN] = SZ_SEN;          ! PACKET SIZE                                SEND DATA
;       3613          MSCP_PKT [.MX1, OPCODE] = OP_SDD;          ! OPCODE = SEND DATA
.       3614          MSCP_PKT [.MX1, BC_LO] = 6;                ! BYTE COUNT TO BE TRANSFERED EQUALS 6
.       3615          MSCP_PKT [.MX1, BUF_0] = DUPPKT;           ! LOAD DESCRIBTOR BUFFER
;       3616          DUPPKT [DUPBF0] = .Cd;ek;                  !LOAD UNIT NUMBER (RDRX)
```

```
;          3617          DUPPKT [DUPBF1] = .CST_ADDR [.CUOFF + 3, D_DBN]; ! LOAD DBN NUMBER
;          3618          DUPPKT [DUPBF2] = .S_PATTERN;                    ! LOAD PATTERN
;          3619          MSCP_PKT [.MX1, MODIFY] = 0;                     !
;          3620          DUPCOMMAND ();                                   ! SENDS AND RECEIVES THE COMMAND
;          3621          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN RETURN; !IF DUP ERROR THEN  RETURN
;          3622          IF (MX1 = GET_PKT (.CCTLR)) LSS 0                ! TRY TO GET AN ENVELOPE. IF FAILURE
;          3623                OR (.EOP_FLAG)
;          3624                THEN
;          3625                (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;          3626                 RETURN;);                                        ! NO POINT IN CONTINUING
;          3627
;          3628          MSCP_PKT [.MX1, MSGLEN] = SZ_REC;                ! PACKET SIZE                                   RECEIVE DATA
;          3629          MSCP_PKT [.MX1, OPCODE] = OP_RCD;                ! OPCODE = RECEIVE DATA
;          3630          MSCP_PKT [.MX1, BC_LO] = 4;                      ! BYTE COUNT TO BE TRANSFERED EQUALS 4
;          3631          MSCP_PKT [.MX1, BUF_0] = DUPPKT;                 ! LOAD DESCRIBTOR BUFFER
;          3632          MSCP_PKT [.MX1, MODIFY] = 0;                     !
;          3633          DUPCOMMAND ();                                   ! SENDS AND RECEIVES THE COMMAND
;          3634          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN RETURN; !IF DUP ERROR THEN  RETURN
;          3635          IF (.DUPPKT [DUPTYPE]  NEQ 3)
;          3636             OR (.DUPPKT [DUPMESG] NEQ 3)                  ! IF CORRECT RESPONSE
;          3637              OR (.DUPPKT [DUPBF1] NEQ 0)
;          3638              THEN
;          3639                  (HARD_ERROR ();
;          3640                   CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;          3641                   RETURN;);                                      ! NO POINT IN CONTINUING
;          3642
;          3643          IF ((MX1 = GET_PKT (.CCTLR)) LSS 0)              ! TRY TO GET AN ENVELOPE. IF FAILURE
;          3644              OR (.EOP_FLAG)
;          3645              THEN
;          3646                  (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;          3647                   RETURN;);                                      ! NO POINT IN CONTINUING
;          3648
;          3649          T_ADDR [T_BLK_WT] = .T_ADDR [T_BLK_WT] + 1;
;          3650          END;


                                                .SBTTL  DUPWRTDBN MULTI-DRIVE TEST ROUTINES
000000  010146                          DUPWRTDBN:
                                                MOV     R1,-(SP)                ;                                       3561
000002  013746  000160'                         MOV     MX1,-(SP)               ;                                       3579
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)      ;                                       3580
000032  012760  000001  000024G                 MOV     #1,MSCP.PKT+24(R0)      ;                                       3581
000040  112760  000127  000026G                 MOVB    #127,MSCP.PKT+26(R0)    ;                                       3582
000046  112760  000122  000027G                 MOVB    #122,MSCP.PKT+27(R0)    ;                                       3583
000054  112760  000124  000030G                 MOVB    #124,MSCP.PKT+30(R0)    ;                                       3584
000062  112760  000104  000031G                 MOVB    #104,MSCP.PKT+31(R0)    ;                                       3585
000070  112760  000102  000032G                 MOVB    #102,MSCP.PKT+32(R0)    ;                                       3586
000076  112760  000116  000033G                 MOVB    #116,MSCP.PKT+33(R0)    ;                                       3587
000104  004737  000000V                         JSR     PC,DUPCOMMAND           ;                                       3588
000110  013700  000000G                         MOV     CUOFF,R0                ;                                       3589
000114  006300                                  ASL     R0
000116  063700  000000G                         ADD     CST.ADDR,R0
000122  032760  040000  000006                  BIT     #40000,6(R0)
```

```
000130  001023                                  BNE     2$                      ;                       3561
000132  013716  000000G                         MOV     CCTLR,(SP)              ;                       3590
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             ;                       3591
000156  001412                                  BEQ     3$
000160  013700  000000G                 1$:     MOV     CUOFF,R0                ;                       3593
000164  006300                                  ASL     R0
000166  063700  000000G                         ADD     CST.ADDR,R0
000172  052760  040000  000006                 BIS     #40000,6(R0)
000200  022626                          2$:     CMP     (SP)+,(SP)+             ;                       3590
000202  000504                                  BR      6$                      ;                       3593
000204  013716  000160'                 3$:     MOV     MX1,(SP)                ;                       3596
000210  012746  000104                          MOV     #104,-(SP)
000214  004737  000000G                         JSR     PC,BL$MUL
000220  012760  000034  000006G                 MOV     #34,MSCP.PKT+6(R0)
000226  112760  000005  000022G                 MOVB    #5,MSCP.PKT+22(R0)      ;                       3597
000234  012760  000002  000026G                 MOV     #2,MSCP.PKT+26(R0)      ;                       3598
000242  012760  000000G 000032G                 MOV     #DUPPKT,MSCP.PKT+32(R0) ;                       3599
000250  005060  000024G                         CLR     MSCP.PKT+24(R0)         ;                       3600
000254  004737  000000V                         JSR     PC,DUPCOMMAND           ;                       3601
000260  013700  000000G                         MOV     CUOFF,R0                ;                       3602
000264  006300                                  ASL     R0
000266  063700  000000G                         ADD     CST.ADDR,R0
000272  032760  040000  000006                 BIT     #40000,6(R0)
000300  001043                                  BNE     5$                      ;                       3561
000302  013700  000000G                         MOV     DUPPKT,R0               ;                       3603
000306  042700  007777                          BIC     #7777,R0
000312  020027  010000                          CMP     R0,#10000
000316  001022                                  BNE     4$
000320  013700  000000G                         MOV     DUPPKT,R0               ;                       3604
000324  042700  170000                          BIC     #170000,R0
000330  020027  000006                          CMP     R0,#6
000334  001013                                  BNE     4$
000336  013716  000000G                         MOV     CCTLR,(SP)              ;                       3605
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             ;                       3606
000362  001415                                  BEQ     7$
000364  004737  000000V                 4$:     JSR     PC,HARD.ERROR           ;                       3608
000370  013700  000000G                         MOV     CUOFF,R0                ;                       3609
000374  006300                                  ASL     R0
000376  063700  000000G                         ADD     CST.ADDR,R0
000402  052760  040000  000006                 BIS     #40000,6(R0)
000410  062706  000006                  5$:     ADD     #6,SP                   ;                       3603
000414  000504                          6$:     BR      10$                     ;                       3608
000416  013716  000160'                 7$:     MOV     MX1,(SP)                ;                       3612
000422  012746  000104                          MOV     #104,-(SP)
000426  004737  000000G                         JSR     PC,BL$MUL
000432  012760  000034  000006G                 MOV     #34,MSCP.PKT+6(R0)
000440  112760  000004  000022G                 MOVB    #4,MSCP.PKT+22(R0)      ;                       3613
000446  012760  000006  000026G                 MOV     #6,MSCP.PKT+26(R0)      ;                       3614
000454  012760  000000G 000032G                 MOV     #DUPPKT,MSCP.PKT+32(R0) ;                       3615
000462  013737  000000G 000000G                 MOV     CDISK,DUPPKT            ;                       3616
```

```
000470  013701  000000G                        MOV     CUOFF,R1                        ;                       3617
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              ;                       3618
000522  005060  000024G                        CLR     MSCP.PKT+24(R0)                 ;                       3619
000526  004737  000000V                        JSR     PC,DUPCOMMAND                   ;                       3620
000532  013700  000000G                        MOV     CUOFF,R0                        ;                       3621
000536  006300                                  ASL     R0
000540  063700  000000G                        ADD     CST.ADDR,R0
000544  032760  040000  000006                BIT     #40000,6(R0)
000552  001023                                  BNE     9$                              ;                       3561
000554  013716  000000G                        MOV     CCTLR,(SP)                      ;                       3622
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                     ;                       3623
000600  001413                                  BEQ     11$
000602  013700  000000G              8$:       MOV     CUOFF,R0                        ;                       3625
000606  006300                                  ASL     R0
000610  063700  000000G                        ADD     CST.ADDR,R0
000614  052760  040000  000006                BIS     #40000,6(R0)
000622  062706  000010                9$:       ADD     #10,SP                          ;                       3622
000626  000515                      10$:       BR      17$                             ;                       3625
000630  013716  000160'               11$:     MOV     MX1,(SP)                        ;                       3628
000634  012746  000104                        MOV     #104,-(SP)
000640  004737  000000G                        JSR     PC,BL$MUL
000644  012760  000034  000006G                MOV     #34,MSCP.PKT+6(R0)
000652  112760  000005  000022G                MOVB    #5,MSCP.PKT+22(R0)              ;                       3629
000660  012760  000004  000026G                MOV     #4,MSCP.PKT+26(R0)              ;                       3630
000666  012760  000000G  000032G                MOV     #DUPPKT,MSCP.PKT+32(R0)         ;                       3631
000674  005060  000024G                        CLR     MSCP.PKT+24(R0)                 ;                       3632
000700  004737  000000V                        JSR     PC,DUPCOMMAND                   ;                       3633
000704  013700  000000G                        MOV     CUOFF,R0                        ;                       3634
000710  006300                                  ASL     R0
000712  063700  000000G                        ADD     CST.ADDR,R0
000716  032760  040000  000006                BIT     #40000,6(R0)
000724  001054                                  BNE     16$                             ;                       3561
000726  013700  000000G                        MOV     DUPPKT,R0                       ;                       3635
000732  042700  007777                        BIC     #7777,R0
000736  020027  030000                        CMP     R0,#30000
000742  001012                                  BNE     12$
000744  013700  000000G                        MOV     DUPPKT,R0                       ;                       3636
000750  042700  170000                        BIC     #170000,R0
000754  020027  000003                        CMP     R0,#3
000760  001003                                  BNE     12$
000762  005737  000002G                        TST     DUPPKT+2                        ;                       3637
000766  001403                                  BEQ     13$
000770  004737  000000V              12$:      JSR     PC,HARD.ERROR                   ;                       3639
000774  000413                                  BR      14$                             ;                       3640
000776  013716  000000G              13$:      MOV     CCTLR,(SP)                      ;                       3643
001002  004737  000000G                        JSR     PC,GET.PKT
001006  010037  000160'                       MOV     R0,MX1
001012  002404                                  BLT     14$
001014  132737  000001  000000G                BITB    #1,EOP.FLAG                     ;                       3644
```

```
001022  001411                          BEQ     15$
001024  013700  000000G           14$:  MOV     CUOFF,R0                ;                3646
001030  006300                        ASL     R0
001032  063700  000000G                ADD     CST.ADDR,R0
001036  052760  040000  000006        BIS     #40000,6(R0)
001044  000404                        BR      16$                     ;                3643
001046  013700  000000G           15$:  MOV     T.ADDR,R0               ;                3649
001052  005260  000050                INC     50(R0)
001056  062706  000012           16$:  ADD     #12,SP                  ;                3575
001062  012601                     17$:  MOV     (SP)+,R1                ;                3561
001064  000207                        RTS     PC

; Routine Size:  283 words,     Routine Base:  $CODE$ + 11646
; Maximum stack depth per invocation:  7 words
```

```
;       3651  ROUTINE DUPREDDBN : NOVALUE =
;       3652
;       3653  !+
;       3654  ! THIS ROUTINE IS CALLED BY DUP ROUTINE TO USE THE CONTROLLER LOCAL PROGRAM
;       3655  ! "REDDBN".  TO USE THE PROGRAM THE OPTIONAL DUP SUB-PROTOCAL IS USED TO
;       3656  ! COMMUNICATE WITH THE CONTROLLER.  THE PROGRAM READS A DIAGONOSTIC BLOCK (DBN)
;       3657  ! AND PLACES IT IN THE DUP BUFFER CALLED "DUPPKT".  IF AN ERROR OCCURS WHILE
;       3658  ! RUNNING THE CONTROLLER LOCAL PROGRAM THE ERROR IS USSUALLY REPORTED IN THE
;       3659  ! DUP BUFFER. (EX. ILLEGAL UNIT NUMBER, ILLEGAL BLK #, DEVICE ERROR, ZERO LENGHT MSG)
;       3660  !
;       3661  !
;       3662  !       IMPLICIT INPUTS:
;       3663  !               CST_ADDR    - CONTAINS THE CURRENT CONTROLLER STATUS TABLE
;       3664  !               CUOFF       - CURRENT OFFSET IN CST TABLE FOR PARTICULAR DRIVE
;       3665  !
;       3666  BEGIN
;       3667
;       3668  !PRINTX (DER12);
;       3669
;       3670  MSCP_PKT [.MX1, MSGLEN] = SZ_ELP;                ! PACKET SIZE                                   EXECUTE REDDBN PROGRAM
;       3671  MSCP_PKT [.MX1, OPCODE] = OP_ELP;                ! OPCODE = EXECUTE LOCAL PROGRAM
;       3672  MSCP_PKT [.MX1, MODIFY] = 1;                     ! STANDALONE MODIFIER
;       3673  MSCP_PKT [.MX1, L1] = %asci 'R';                  ! FILL IN PROGRAM NAME WITH ASCII LETTERS
;       3674  MSCP_PKT [.MX1, L2] = %asci 'E';
;       3675  MSCP_PKT [.MX1, L3] = %asci 'D';
;       3676  MSCP_PKT [.MX1, L4] = %asci 'D';
;       3677  MSCP_PKT [.MX1, L5] = %asci 'B';
;       3678  MSCP_PKT [.MX1, L6] = %asci 'N';
;       3679  DUPCOMMAND ();                                   ! SENDS AND RECEIVES THE COMMAND
;       3680  IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
;       3681  IF (MX1 = GET_PKT (.CCTLR)) LSS 0               ! TRY TO GET AN ENVELOPE. IF FAILURE
;       3682          OR (.EOP_FLAG)
;       3683          THEN
;       3684                  (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;       3685                   RETURN;);                                ! NO POINT IN CONTINUING
;       3686
;       3687          MSCP_PKT [.MX1, MSGLEN] = SZ_REC;          ! PACKET SIZE                                 RECIEVE DATA
;       3688          MSCP_PKT [.MX1, OPCODE] = OP_RCD;          ! OPCODE = RECEIVE DATA
;       3689          MSCP_PKT [.MX1, BC_LO] = 2;                ! BYTE COUNT TO BE TRANSFERED EQUALS 2
;       3690          MSCP_PKT [.MX1, BUF_0] = DUPPKT;           ! LOAD DESCRIBTOR BUFFER
;       3691          MSCP_PKT [.MX1, MODIFY] = 0;               !
;       3692          DUPCOMMAND ();                             ! SENDS AND RECEIVES THE COMMAND
;       3693          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN  RETURN; !IF DUP ERROR THEN  RETURN
;       3694          IF .DUPPKT [DUPTYPE]  NEQ 1
;       3695             OR .DUPPKT [DUPMESG] NEQ 5              ! IF CORRECT RESPONSE
;       3696             OR (MX1 = GET_PKT (.CCTLR)) LSS 0       ! TRY TO GET AN ENVELOPE. IF FAILURE
;       3697             OR (.EOP_FLAG)
;       3698             THEN
;       3699                  (HARD_ERROR ();
;       3700                   CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;       3701                   RETURN;);                                ! NO POINT IN CONTINUING
;       3702
;       3703          MSCP_PKT [.MX1, MSGLEN] = SZ_SEN;          ! PACKET SIZE                                 SEND DATA
;       3704          MSCP_PKT [.MX1, OPCODE] = OP_SDD;          ! OPCODE = SEND DATA
;       3705          MSCP_PKT [.MX1, BC_LO] = 4;                ! BYTE COUNT TO BE TRANSFERED EQUALS 4
;       3706          MSCP_PKT [.MX1, BUF_0] = DUPPKT;           ! LOAD DESCRIBTOR BUFFER
```

```
;         3707          DUPPKT [DUPBF0] = .CDISK;                          ! LOAD UNIT NUMBER (RDRX)
;         3708          DUPPKT [DUPBF1] = .CST_ADDR [.CUOFF + 3, D_DBN];   ! LOAD DBN NUMBER
;         3709          MSCP_PKT [.MX1, MODIFY] = 0;                       !
;         3710          DUPCOMMAND ();                                     ! SENDS AND RECEIVES THE COMMAND
;         3711          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN     RETURN; !IF DUP ERROR THEN  RETURN
;         3712          IF (MX1 = GET_PKT (.CCTLR)) LSS 0                  ! TRY TO GET AN ENVELOPE. IF FAILURE
;         3713                 OR (.EOP_FLAG)
;         3714                 THEN
;         3715                 (CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;         3716                  RETURN;);                                                 ! NO POINT IN CONTINUING
;         3717
;         3718          MSCP_PKT [.MX1, MSGLEN] = SZ_REC;                  ! PACKET SIZE                                RECEIVE DATA
;         3719          MSCP_PKT [.MX1, OPCODE] = OP_RCD;                  ! OPCODE = GET DUST STATUS
;         3720          MSCP_PKT [.MX1, BC_LO] = 514;                      ! BYTE COUNT TO BE TRANSFERED EQUALS 512
;         3721          MSCP_PKT [.MX1, BUF_0] = DUPPKT;                   ! LOAD DESCRIBTOR BUFFER
;         3722          MSCP_PKT [.MX1, MODIFY] = 0;                       !
;         3723          DUPCOMMAND ();                                     ! SENDS AND RECEIVES THE COMMAND
;         3724          IF .CST_ADDR [.CUOFF + 3, DUPERROR] EQL 1 THEN     RETURN; !IF DUP ERROR THEN  RETURN
;         3725          IF .DUPPKT [DUPTYPE]  NEQ 6
;         3726             OR .DUPPKT [DUPMESG] NEQ 2                      ! IF CORRECT RESPONSE
;         3727             OR (MX1 = GET_PKT (.CCTLR)) LSS 0               ! TRY TO GET AN ENVELOPE. IF FAILURE
;         3728             OR (.EOP_FLAG)
;         3729             THEN
;         3730                  (HARD_ERROR ();
;         3731                   CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;         3732                   RETURN;);                                                ! NO POINT IN CONTINUING
;         3733
;         3734  CST_ADDR [.CUOFF + 3, D_DBN] = .CST_ADDR [.CUOFF + 3, D_DBN] + 1;       ! INCREMENT RELATIVE DBN COUNTER
;         3735  T_ADDR [T_BLK_RD] = .T_ADDR [T_BLK_RD] + 1;
;         3736
;         3737  END;


                                                .SBTTL  DUPREDDBN MULTI-DRIVE TEST ROUTINES
000000  010146                          DUPREDDBN:
                                                MOV     R1,-(SP)                ;                                       3651
000002  013746  000160'                         MOV     MX1,-(SP)               ;                                       3670
000006  012746  000104                          MOV     #104,-(SP)
000012  004737  000000G                         JSR     PC,BL$MUL
000016  012760  000022  000006G                 MOV     #22,MSCP.PKT+6(R0)
000024  112760  000003  000022G                 MOVB    #3,MSCP.PKT+22(R0)      ;                                       3671
000032  012760  000001  000024G                 MOV     #1,MSCP.PKT+24(R0)      ;                                       3672
000040  112760  000122  000026G                 MOVB    #122,MSCP.PKT+26(R0)    ;                                       3673
000046  112760  000105  000027G                 MOVB    #105,MSCP.PKT+27(R0)    ;                                       3674
000054  112760  000104  000030G                 MOVB    #104,MSCP.PKT+30(R0)    ;                                       3675
000062  112760  000104  000031G                 MOVB    #104,MSCP.PKT+31(R0)    ;                                       3676
000070  112760  000102  000032G                 MOVB    #102,MSCP.PKT+32(R0)    ;                                       3677
000076  112760  000116  000033G                 MOVB    #116,MSCP.PKT+33(R0)    ;                                       3678
000104  004737  000000V                         JSR     PC,DUPCOMMAND           ;                                       3679
000110  013700  000000G                         MOV     CUOFF,R0                ;                                       3680
000114  006300                                  ASL     R0
000116  063700  000000G                         ADD     CST.ADDR,R0
000122  032760  040000  000006                  BIT     #40000,6(R0)
000130  001023                                  BNE     2$                      ;                                       3651
000132  013716  000000G                         MOV     CCTLR,(SP)              ;                                       3681
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             ;                       3682
000156  001412                                 BEQ     3$
000160  013700  000000G                1$:     MOV     CUOFF,R0                ;                       3684
000164  006300                                 ASL     R0
000166  063700  000000G                        ADD     CST.ADDR,R0
000172  052760  040000  000006                BIS     #40000,6(R0)
000200  022626                         2$:     CMP     (SP)+,(SP)+             ;                       3681
000202  000504                                 BR      6$                      ;                       3684
000204  013716  000160'                3$:     MOV     MX1,(SP)                ;                       3687
000210  012746  000104                        MOV     #104,-(SP)
000214  004737  000000G                        JSR     PC,BL$MUL
000220  012760  000034  000006G                MOV     #34,MSCP.PKT+6(R0)
000226  112760  000005  000022G                MOVB    #5,MSCP.PKT+22(R0)      ;                       3688
000234  012760  000002  000026G                MOV     #2,MSCP.PKT+26(R0)      ;                       3689
000242  012760  000000G 000032G                MOV     #DUPPKT,MSCP.PKT+32(R0) ;                       3690
000250  005060  000024G                        CLR     MSCP.PKT+24(R0)         ;                       3691
000254  004737  000000V                        JSR     PC,DUPCOMMAND           ;                       3692
000260  013700  000000G                        MOV     CUOFF,R0                ;                       3693
000264  006300                                 ASL     R0
000266  063700  000000G                        ADD     CST.ADDR,R0
000272  032760  040000  000006                BIT     #40000,6(R0)
000300  001043                                 BNE     5$                      ;                       3651
000302  013700  000000G                        MOV     DUPPKT,R0               ;                       3694
000306  042700  007777                        BIC     #7777,R0
000312  020027  010000                        CMP     R0,#10000
000316  001022                                 BNE     4$
000320  013700  000000G                        MOV     DUPPKT,R0               ;                       3695
000324  042700  170000                        BIC     #170000,R0
000330  020027  000005                        CMP     R0,#5
000334  001013                                 BNE     4$
000336  013716  000000G                        MOV     CCTLR,(SP)              ;                       3696
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             ;                       3697
000362  001415                                 BEQ     7$
000364  004737  000000V                4$:     JSR     PC,HARD.ERROR           ;                       3699
000370  013700  000000G                        MOV     CUOFF,R0                ;                       3700
000374  006300                                 ASL     R0
000376  063700  000000G                        ADD     CST.ADDR,R0
000402  052760  040000  000006                BIS     #40000,6(R0)
000410  062706  000006                5$:     ADD     #6,SP                   ;                       3694
000414  000501                         6$:     BR      10$                     ;                       3699
000416  013716  000160'                7$:     MOV     MX1,(SP)                ;                       3703
000422  012746  000104                        MOV     #104,-(SP)
000426  004737  000000G                        JSR     PC,BL$MUL
000432  012760  000034  000006G                MOV     #34,MSCP.PKT+6(R0)
000440  112760  000004  000022G                MOVB    #4,MSCP.PKT+22(R0)      ;                       3704
000446  012760  000004  000026G                MOV     #4,MSCP.PKT+26(R0)      ;                       3705
000454  012760  000000G 000032G                MOV     #DUPPKT,MSCP.PKT+32(R0) ;                       3706
000462  013737  000000G 000000G                MOV     CDISK,DUPPKT            ;                       3707
000470  013701  000000G                        MOV     CUOFF,R1                ;                       3708
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)          ;                3709
000520  004737  000000V                  JSR     PC,DUPCOMMAND            ;                3710
000524  013700  000000G                  MOV     CUOFF,R0                 ;                3711
000530  006300                           ASL     R0
000532  063700  000000G                  ADD     CST.ADDR,R0
000536  032760  040000  000006          BIT     #40000,6(R0)
000544  001023                           BNE     9$                       ;                3651
000546  013716  000000G                  MOV     CCTLR,(SP)               ;                3712
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              ;                3713
000572  001413                           BEQ     11$
000574  013700  000000G          8$:     MOV     CUOFF,R0                 ;                3715
000600  006300                           ASL     R0
000602  063700  000000G                  ADD     CST.ADDR,R0
000606  052760  040000  000006          BIS     #40000,6(R0)
000614  062706  000010          9$:     ADD     #10,SP                   ;                3712
000620  000520                  10$:    BR      15$                      ;                3715
000622  013716  000160'          11$:    MOV     MX1,(SP)                 ;                3718
000626  012746  000104                   MOV     #104,-(SP)
000632  004737  000000G                  JSR     PC,BL$MUL
000636  012760  000034  000006G          MOV     #34,MSCP.PKT+6(R0)
000644  112760  000005  000022G          MOVB    #5,MSCP.PKT+22(R0)       ;                3719
000652  012760  001002  000026G          MOV     #1002,MSCP.PKT+26(R0)    ;                3720
000660  012760  000000G  000032G          MOV     #DUPPKT,MSCP.PKT+32(R0)  ;                3721
000666  005060  000024G                  CLR     MSCP.PKT+24(R0)          ;                3722
000672  004737  000000V                  JSR     PC,DUPCOMMAND            ;                3723
000676  013700  000000G                  MOV     CUOFF,R0                 ;                3724
000702  006300                           ASL     R0
000704  063700  000000G                  ADD     CST.ADDR,R0
000710  032760  040000  000006          BIT     #40000,6(R0)
000716  001057                           BNE     14$                      ;                3651
000720  013700  000000G                  MOV     DUPPKT,R0                ;                3725
000724  042700  007777                   BIC     #7777,R0
000730  020027  060000                   CMP     R0,#60000
000734  001022                           BNE     12$
000736  013700  000000G                  MOV     DUPPKT,R0                ;                3726
000742  042700  170000                   BIC     #170000,R0
000746  020027  000002                   CMP     R0,#2
000752  001013                           BNE     12$
000754  013716  000000G                  MOV     CCTLR,(SP)               ;                3727
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              ;                3728
001000  001413                           BEQ     13$
001002  004737  000000V          12$:    JSR     PC,HARD.ERROR            ;                3730
001006  013700  000000G                  MOV     CUOFF,R0                 ;                3731
001012  006300                           ASL     R0
001014  063700  000000G                  ADD     CST.ADDR,R0
001020  052760  040000  000006          BIS     #40000,6(R0)
001026  00041.                           BR      14$                      ;                3725
001030  013700  000000G          13$:    MOV     CUOFF,R0                 ;                3734
```

```
001034  006300                                  ASL     R0
001036  063700  000000G                         ADD     CST.ADDR,R0
001042  105260  000006                          INCB    6(R0)
001046  013700  000000G                         MOV     T.ADDR,R0                       ;                       3735
001052  005260  000054                          INC     54(R0)
001056  062706  000012                  14$:    ADD     #12,SP                          ;                       3666
001062  012601                          15$:    MOV     (SP)+,R1                        ;                       3651
001064  000207                                  RTS     PC

; Routine Size:  283 words,     Routine Base:  $CODE$ + 12734
; Maximum stack depth per invocation:  7 words
```

```
;        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          IF SEND (.MX1) EQLU FAILURE                     ! ATTEMPT SEND; IF CTLR IS OFFLINE
;        3758          THEN
;        3759                  BEGIN
;        3760                  PUT_PKT (.MX1);
;        3761                  MX1 = -1;                               ! RETURN ENVELOPE TO POOL
;        3762                  CST_ADDR [.CUOFF + 3, DUPERROR] = 1;
;        3763                  END
;        3764
;        3765          ELSE    BEGIN                                   ! IF SEND WAS SUCCESSFUL
;        3766                  QIO [.CCTLR] = .QIO [.CCTLR] + 1;               ! INCR OUTSTANDING QIO COUNT
;        3767                  BREAK;                                  ! BREAK FOR SUPERVISOR TO CATCH USER REQUESTS
;        3768                  WAIT ();                                ! WAIT FOR RETPKT RESPONSE
;        3769                  PROC_RETPKT();                          ! PROCESS RETURN PACKET TO SEE IF OK FOR DUP
;        3770                  END;
;        3771          END;
;        3772  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$MUL
000026  012701  000010G                 MOV     #MSCP.PKT+10,R1
000032  060001                          ADD     R0,R1
000034  112711  000001                  MOVB    #1,(R1)                         ;                       3753
000040  112761  000002  000001         MOVB    #2,1(R1)                        ;                       3754
000046  005060  000016G                 CLR     MSCP.PKT+16(R0)                 ;                       3755
000052  013716  000160'                 MOV     MX1,(SP)                        ;                       3757
000056  004737  000000G                 JSR     PC,SEND
000062  005700                          TST     R0
000064  001020                          BNE     1$
000066  013716  000160'                 MOV     MX1,(SP)                        ;                       3760
000072  004737  000000G                 JSR     PC,PUT.PKT
```

```
000076  012737  177777  000160'                   MOV     #-1,MX1                         ;                       3761
000104  013700  000000G                           MOV     CUOFF,R0                        ;                       3762
000110  006300                                    ASL     R0
000112  063700  000000G                           ADD     CST.ADDR,R0
000116  052760  040000  000006                    BIS     #40000,6(R0)
000124  000411                                    BR      2$                              ;                       3757
000126  013700  000000G                 1$:       MOV     CCTLR,R0                        ;                       3766
000132  105260  000000G                           INCB    QIO(R0)
000136  104422                                    TRAP    22
000140  004737  000000G                           JSR     PC,WAIT                         ;                       3768
000144  004737  000000V                           JSR     PC,PROC.RETPKT                  ;                       3769
000150  022626                          2$:       CMP     (SP)+,(SP)+                     ;                       3751
000152  012601                          3$:       MOV     (SP)+,R1                        ;                       3738
000154  000207                                    RTS     PC

; Routine Size:  55 words,      Routine Base:  $CODE$ + 14022
; Maximum stack depth per invocation:  5 words
```

```
;        3773  routine QIO_LBN : novalue =
;        3774
;        3775  !+
;        3776  !       THIS ROUTINE IS CALLED BY QIO GEN TO SELECT THE LOGICAL BLOCK NUMBER TO
;        3777  !       BE USED FOR THE CURRENT QIO OR QIO PAIR.
;        3778  !
;        3779  !       IF THE OPERATOR CHOSE THE RANDOM SEEK MODE OPTION, THEN THE LBN IS
;        3780  !       RANDOMLY CHOSEN WITHIN THE SPECIFIED LIMITS FOR THE LBN.
;        3781  !       OTHERWISE, THE NEXT SEQUENTIAL LBN IS DERIVED FROM THE BLOCK SEQUENCE
;        3782  !       TABLE (BST).
;        3783  !
;        3784  !       IMPLICIT INPUTS:
;        3785  !               L$LUN   CURRENT (DIAGNOSTIC SUPERVIOR) UNIT NUMBER
;        3786  !
;        3787  !       IMPLICIT OUTPUTS:
;        3788  !               THE LBN IS LOADED INTO ONE OR BOTH MSCP PACKETS.
;        3789  !-
;        3790
;        3791      begin
;        3792
;        3793      own
;        3794          LBN_SAVE : word initial (0);            ! LBN SELECTED IN LAST PASS
;        3795
;        3796      local
;        3797          S_TEMP : word,                          ! TEMPORARY STORAGE FOR START LBN
;        3798          E_TEMP : word,                          ! TEMPORARY STORAGE FOR END LBN
;        3799          LBN : word,                             ! LOGICAL BLOCK NUMBER
;        3800          RD51_DISK : byte initial (byte (TRUE)); ! FLAG TO INDICATE WINCHESTER DISK SELECTED
;        3801
;        3802      S_TEMP = .CST_ADDR [.CUOFF + 1, D_BEG];     ! STARTING LBN
;        3803      E_TEMP = .CST_ADDR [.CUOFF + 2, D_END];     ! ENDING LBN
;        3804
;        3805      if .CST_ADDR [.CUOFF, D_TYPE] eql RX_50
;        3806      then
;        3807          RD51_DISK = FALSE;
;        3808
;        3809      if BIT_TST (SWP_FLAGS, SWF_RDM)             ! IF RANDOM SEEK MODE
;        3810      then
;        3811          begin
;        3812
;        3813          if (.RD51_DISK) and
;        3814             (((.RANDOM [0] and %o'077777') mod (99)) lequ 33)
;        3815          then
;        3816              LBN = .LBN_SAVE                     ! REDUCE SEEKS ON RDs by 33%
;        3817          else
;        3818              LBN = .S_TEMP + ((.RANDOM [3] and %o'077777') mod (.E_TEMP - .S_TEMP + 1));
;        3819
;        3820          end
;        3821      else
;        3822          begin                                   ! ELSE - SEQUENTIAL LBN MODE
;        3823          LBN = .BST [.L$LUN];                    ! GET LBN FROM BST
;        3824          BST [.L$LUN] = .BST [.L$LUN] + .TRK_SGN [.L$LUN];! MODIFY LBN (INC OR DEC FOR NEXT PASS)
;        3825
;        3826          if .TRK_SGN [.L$LUN] lss 0
;        3827          then
;        3828              begin
```

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


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


                                        .SBTTL  QIO.LBN MULTI-DRIVE TEST ROUTINES
014200                                  .PSECT  $CODE$,  RO

000000  004137  000000G         QIO.LBN:JSR     R1,$SAVE5                       ;                       3773
000004  112705  000001                  MOVB    #1,R5                           ; ^,RD51.DISK           3791
000010  013702  000000G                 MOV     CST.ADDR,R2                     ;                       3802
000014  013701  000000G                 MOV     CUOFF,R1
000020  010100                          MOV     R1,R0
```

```
000000  006500                          ASL     R0
000024  060200                          ADD     R2,R0
000026  016005  000002                  MOV     2(R0),R5        ; *,S.TEMP
000032  010100                          MOV     R1,R0           ;                                       3803
000034  006500                          ASL     R0
000036  060200                          ADD     R2,R0
000040  016004  000004                  MOV     4(R0),R4        ; *,E.TEMP
000044  006501                          ASL     R1              ;                                       3805
000046  060201                          ADD     R2,R1
000050  132711  000004                  BITB    #4,(R1)
000054  001001                          BNE     1$
000056  105005                          CLRB    R5              ; RD51.DISK                             3807
000060  032737  000002  0000006   1$:   BIT     #2,SWP.FLAGS    ;                                       3809
000066  001437                          BEQ     3$
000070  006005                          ROR     R5              ; RD51.DISK                             3813
000072  103017                          BCC     2$
000074  013746  000116                  MOV     RANDOM, (SP)    ;                                       3814
000100  042716  100000                  BIC     #100000,(SP)
000104  012746  000143                  MOV     #143,-(SP)
000110  004737  0000006                 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                                 3816
000130  000445                          BR      5$              ;                                       3813
000132  013746  000124'          2$:    MOV     RANDOM+6,-(SP)  ;                                       3818
000136  042716  100000                  BIC     #100000,(SP)
000142  010400                          MOV     R4,R0           ; E.TEMP,*
000144  160300                          SUB     R3,R0           ; S.TEMP,*
000146  010046                          MOV     R0,-(SP)
000150  005216                          INC     (SP)
000152  004737  0000006                 JSR     PC,BL$MOD
000156  060300                          ADD     R3,R0           ; S.TEMP,*
000160  010001                          MOV     R0,R1           ; *,LBN
000162  022626                          CMP     (SP)+,(SP)+
000164  000427                          BR      5$              ;                                       3809
000166  013700  0000006         3$:    MOV     L$LUN,R0        ;                                       3823
000172  006500                          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     ;                                       3824
000210  061012                          ADD     (R0),(R2)
000212  005710                          TST     (R0)            ;                                       3826
000214  002006                          BGE     4$
000216  021203                          CMP     (R2),R3         ; *,S.TEMP                              3830
000220  103011                          BHIS    5$
000222  010312                          MOV     R3,(R2)         ; S.TEMP,*                              3833
000224  012710  000001                  MOV     #1,(R0)         ;                                       3834
000230  000405                          BR      5$              ;                                       3836
000232  021204                  4$:     CMP     (R2),R4         ; *,E.TEMP                              3841
000234  101403                          BLOS    5$
000236  010412                          MOV     R4,(R2)         ; E.TEMP,*                              3844
000240  012710  177777                  MOV     #-1,(R0)        ;                                       3845
000244  020103                  5$:     CMP     R1,R3           ; LBN,S.TEMP                            3852
000246  103001                          BHIS    6$
```

```
000250  010301                          MOV     R3,R1                   ; S.TEMP,LBN                    3854
000252  020104                  6$:     CMP     R1,R4                   ; LBN,E.TEMP                    3856
000254  101401                          BLOS    7$
000256  010401                          MOV     R4,R1                   ; E.TEMP,LBN                    3858
000260  013700  000164                  MOV     MAD1,R0                 ;                               3860
000264  010160  000046                  MOV     R1,46(R0)               ; LBN,*
000270  005737  000162'                 TST     MX2                     ;                               3862
000274  002404                          BLT     8$
000276  013700  000166'                 MOV     MAD2,R0                 ;                               3864
000302  010160  000046                  MOV     R1,46(R0)               ; LBN,*
000306  010137  001252'          8$:    MOV     R1,LBN.SAVE             ; LBN,*                         3866
000312  000207                          RTS     PC                      ;                               3773

; Routine Size:  102 words,     Routine Base:  $CODE$ + 14200
; Maximum stack depth per invocation:  9 words
```

```
;          3869  routine QIO_SIZE : novalue =
;          3870
;          3871  !+
;          3872  !       THIS ROUTINE IS CALLED BY QIO_GEN TO SELECT THE I/O TRANSFER BYTE COUNT
;          3873  !       TO BE USED FOR THE CURRENT QIO OR QIO PAIR. THE BYTE COUNT IS
;          3874  !       DETERMINED BY A RANDOM NUMBER, AND WILL ALWAYS FALL BETWEEN 1 AND THE
;          3875  !       I/O BUFFER SIZE (BYTS_PER_QIO).
;          3876  !
;          3877  !       IMPLICIT OUTPUTS:
;          3878  !               THE BYTE COUNT IS LOADED INTO ONE OR BOTH MSCP PACKETS.
;          3879  !-
;          3880
;          3881      begin
;          3882
;          3883      local
;          3884          SIZE : word,                                                    ! BYTE COUNT
;          3885          BLOCKS_LEFT : word;                                             ! REMAINING BLOCKS LEFT
;          3886
;          3887      SIZE = ((.RANDOM [4] and %o'077777') mod (.BYTS_PER_QIO + 1)) and %o'177760';!GET BYTE COUNT FROM RANDOM NUMBER
;          3888
;          3889      if .SIZE eql 0
;          3890      then
;          3891          SIZE = 16;
;          3892
;          3893      BLOCKS_LEFT = .CST_ADDR [.CUOFF + 2, D_END] - .MAD1 [LBN_L] + 1;        ! REMAINING BLOCK COUNT
;          3894
;          3895      if ((.SIZE + BYTES_PER_SECT - 1) / BYTES_PER_SECT) gtru .BLOCKS_LEFT    ! IF BLOCK COUNT NOT ENOUGH
;          3896      then
;          3897          SIZE = .BLOCKS_LEFT * BYTES_PER_SECT;                           ! ADJUST BYTE COUNT DOWN
;          3898
;          3899      MAD1 [BC_LO] = .SIZE;                                               ! LOAD SIZE INTO 1ST MSCP PACKET
;          3900
;          3901      if .MX2 geq 0                                                       ! IF 2 QIOS
;          3902      then
;          3903          MAD2 [BC_LO] = .SIZE;                                           ! LOAD SIZE INTO 2ND MSCP PACKET
;          3904
;          3905      end;                                                                ! ROUTINE QIO SIZE


                                        .SBTTL  QIO.SIZE MULTI-DRIVE TEST ROUTINES
000000  004137  000000G          QIO.SIZE:
                                        JSR     R1,$SAVE2                       ;                                       3869
000004  013746  000126'                  MOV     RANDOM+10,-(SP)                 ;                                       3887
000010  042716  100000                    BIC     #100000,(SP)
000014  013746  000000G                  MOV     BYTS.PER.QIO,-(SP)
000020  005216                          INC     (SP)
000022  004737  000000G                  JSR     PC,BL$MOD
000026  010002                          MOV     R0,R2                           ; *.SIZE
000030  042702  000017                    BIC     #17,R2                          ; *.SIZE
000034  001002                          BNE     1$                              ;                                       3889
000036  012702  000020                  MOV     #20,R2                          ; *.SIZE                                3891
000042  013701  000000G          1$:     MOV     CUOFF,R1                        ;                                       3893
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,*                    3895
000074  062716  000777                          ADD     #777,(SP)
000100  012746  001000                          MOV     #1000,-(SP)
000104  004737  0000006                         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,*             389
000120  000300                                  SWAB    R0
000122  105000                                  CLRB    R0
000124  006300                                  ASL     R0
000126  010002                                  MOV     R0,R2           ; *,SIZE
000130  013700  000164'                 2$:     MOV     MAD1,R0         ;                           3899
000134  010260  000026                          MOV     R2,26(R0)       ; SIZE,*
000140  005737  000162                          TST     MX2             ;                           3901
000144  002404                                  BLT     3$
000146  013700  000166'                         MOV     MAD2,R0         ;                           3903
000152  010260  000026                          MOV     R2,26(R0)       ; SIZE,*
000156  022626                          3$:     CMP     (SP)+,(SP)+     ;                           3881
000160  000207                                  RTS     PC              ;                           3869

; Routine Size:  57 words,    Routine Base:  $CODE$ + 14514
; Maximum stack depth per invocation:  7 words
```

```
;          3906  routine FILL_BUFF : novalue =
;          3907
;          3908  !+
;          3909  !       THIS ROUTINE IS CALLED BY QIO_GEN TO LOAD THE I/O BUFFER DESCRIBED IN
;          3910  !       THE FIRST MSCP PACKET WITH THE APPROPRIATE DATA PATTERN.
;          3911  !
;          3912  !       THE DATA PATTERN TO BE SELECTED IS BASED ON THE FOLLOWING ALGORITHM:
;          3913  !
;          3914  !               IF THE OPERATOR DEFINED A DATA PATTERN
;          3915  !               THEN
;          3916  !                       SELECT IT
;          3917  !               ELSE
;          3918  !                       GET DATA PATTERN NUMBER FROM SW P TABLE
;          3919  !                       IF DATA PATTERN NUMBER = 0
;          3920  !                       THEN
;          3921  !                               GET DATA PATTERN NUMBER FROM THE UNIT'S ENTRY
;          3922  !                               IN THE DATA PATTERN SEQUENCE TABLE (DPST)
;          3923  !
;          3924  !       NOTE THAT PATTERN # 1 CONSISTS OF RANDOM NUMBERS, AND PATTERNS # 17
;          3925  !       21 USE THE ACTUAL LBN OF THE WRITE REQUEST.
;          3926  !
;          3927  !       IMPLICIT INPUTS:
;          3928  !               L$LUN - CURRENT (DRS) UNIT NUMBER
;          3929  !
;          3930
;          3931      begin
;          3932
;          3933      local
;          3934          DP_NUM : word,                          ! DATA PATTERN NUMBER SELECTED
;          3935          DP_ADDR,                                ! ADDR OF DATA PATTERN (LENGTH)
;          3936          IOB_ADDR,                               ! I/O BUFFER ADDRESS (DESTINATION)
;          3937          SRC_ADDR,                               ! WORKING SOURCE ADDRESS
;          3938          COUNT : word;                           ! NO. OF WORDS IN DATA PATTERN
;          3939
;          3940      if BIT_TST (SWP_FLAGS, SWF_UDP)             ! IF USER DEFINED A DATA PATTERN
;          3941      then
;          3942          DP_ADDR = SWP_UCNT                      ! SELECT IT
;          3943      else
;          3944          begin
;          3945
;          3946          if .SWP_DPAT neq 0                      ! IF USER SELECTED A PRE DEFINED DATA PATTERN
;          3947          then
;          3948              DP_NUM = .SWP_DPAT                  ! SELECT IT
;          3949          else
;          3950              begin
;          3951              DP_NUM = .DPST [.L$LUN];            ! GET PATTERN NUMBER FROM SEQUENCE TABLE
;          3952              DPST [.L$LUN] = .DPST [.L$LUN] + 1; ! ADVANCE TO NEXT PATTERN NUMBER
;          3953
;          3954              if .DPST [.L$LUN] gtru DP_CNT       ! CHECK FOR HIGH LIMIT
;          3955              then
;          3956                  DPST [.L$LUN] = 1;
;          3957
;          3958              end;
;          3959
;          3960          DP_ADDR = .DPA_TBL [.DP_NUM - 1];       ! ADDRESS OF DATA PATTERN (COUNT)
;          3961
```

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

                                .SBTTL  FILL.BUFF MULTI DRIVE TEST ROUTINES
000000  004137  000000G    FILL.BUFF:
                            JSR     R1,$SAVES                   ;                       3906
000004  005746              TST     -(SP)
000006  032737  000100  000000G    BIT     #100,SWP.FLAGS      ;                       3940
000014  001403              BEQ     1$
000016  012703  000000G     MOV     #SWP.UCNT,R3                ; *,DP.ADDR             3942
000022  000443              BR      5$                          ;                       3940
000024  013700  000000G     1$:     MOV     SWP.DPRT,R0         ;                       3946
000030  001402              BEQ     2$
000032  010001              MOV     R0,R1                       ; *,DP.NUM              3948
000034  000414              BR      3$                          ;                       3946
000036  013700  000000G     2$:     MOV     L$LUN,R0            ;                       3951
000042  062700  000060'      ADD     #DPST,R0
000046  005001              CLR     R1                          ; DP.NUM
000050  151001              BISB    (R0),R1                     ; *,DP.NUM
000052  105210              INCB    (R0)                        ;                       3952
000054  121027  000025      CMPB    (R0),#25                    ;                       3954
000060  101402              BLOS    3$
000062  112710  000001      MOVB    #1,(R0)                     ;                       3956
000066  010100              3$:     MOV     R1,R0               ; DP.NUM,*              3960
000070  006300              ASL     R0
```

```
000072  016003  001144'                        MOV     DPA.TBL 2(R0),R3                ; #,DP.ADDR
000076  020127  000021                          CMP     R1,#21                          ; DP.NUM,#                      3962
000102  103413                                 BLO     5$
000104  013700  000164'                        MOV     MAD1,R0                         ;                               3967
000110  006001                                 ROR     R1                              ; DP.NUM                        3965
000112  103004                                 BCC     4$
000114  016063  000046  000002                 MOV     46(R0),2(R3)                    ; #,#(DP.ADDR)                  3967
000122  000403                                 BR      5$                              ;                               3965
000124  016063  000046  000004        4$:     MOV     46(R0),4(R3)                    ; #,#(DP.ADDR)                  3969
000132  013700  000164'                5$:     MOV     MAD1,R0                         ;                               3973
000136  016005  000032                         MOV     32(R0),R5                       ; #,IOB.ADDR
000142  011302                                 MOV     (R3),R2                         ; DP.ADDR,COUNT                 3974
000144  012704  000002                         MOV     #2,R4                           ;                               3975
000150  060304                                 ADD     R3,R4                           ; DP.ADDR,#
000152  010416                                 MOV     R4,(SP)                         ; #,SRC.ADDR
000154  016046  000026                         MOV     26(R0),-(SP)                    ;                               3977
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$:     MOV     @4(SP),(R5)+                    ; SRC.ADDR,IOB.ADDR             3979
000202  062766  000002  000004                ADD     #2,4(SP)                        ; #,SRC.ADDR                    3981
000210  005302                                 DEC     R2                              ; COUNT                         3982
000212  001003                                 BNE     7$                              ;                               3984
000214  011302                                 MOV     (R3),R2                         ; DP.ADDR,COUNT                 3987
000216  010466  000004                         MOV     R4,4(SP)                        ; #,SRC.ADDR                    3988
000222  005201                        7$:     INC     R1                              ; N                             3977
000224  020100                                 CMP     R1,R0                           ; N,#
000226  003763                                 BLE     6$
000230  062706  000006                         ADD     #6,SP                           ;                               3906
000234  000207                                 RTS     PC

; Routine Size:  79 words,      Routine Base:  $CODE$ + 14676
; Maximum stack depth per invocation:  10 words
```

```
;       3994  routine PROC_RETPKT : novalue =
;       3995
;       3996  !+
;       3997  !       THIS ROUTINE IS CALLED FROM THE MULTI_DRIVE "EXECUTIVE" AND DUP_COMMAND TO CHECK FOR
;       3998  !       AND PROCESS ANY RETURN PACKETS THAT HAVE BEEN "SENT" BY THE "DRIVER"
;       3999  !       PORTION OF THE PROGRAM. THE I/O DONE QUEUE (IODQ) ACTS AS THE LINK
;       4000  !       BETWEEN THE TWO PROGRAM PARTS; IT HOLDS INDECES OF RETURN PACKETS WHICH
;       4001  !       REQUIRE PROCESSING.
;       4002  !
;       4003  !       UNDER THE MULTI-DRIVE TEST, RETURN PACKETS ORIGINATE FROM TWO SOURCES:
;       4004  !               1. MSCP         - THE MORE COMMON, DESCRIBING A COMPLETED I/O
;       4005  !                                 OPERATION.
;       4006  !               2. DUP          - THE LESS COMMON, DESCRIBING A PORTION OF I/O
;       4007  !                                 COMMUNICATIONS WITH THE CONTROLLER PROGRAM.
;       4008  !               3. THE PROGRAM "DRIVER" - DESCRIBING A CONTROLLER ERROR OR
;       4009  !                                 COMMAND TIMEOUT.
;       4010  !-
;       4011
;       4012      while .IODQ_IN neq .IODQ_OUT do                   ! DO UNTIL I/O DONE QUEUE IS EMPTY
;       4013          begin
;       4014          RP_INDX = OUT_IODQ ();                        ! GET INDEX OF NEXT RETPKT AND ADVANCE OUT POINTER
;       4015          RP_ADDR = RETPKT + (.RP_INDX * RP_LEN * 2);   ! CALCULATE RETPKT ADDRESS
;       4016          SET_CPAR (.RP_ADDR [CTLR]);                   ! SET UP CURRENT CONTROLLER PARAMETERS
;       4017
;       4018          selectoneu .RP_ADDR [CONID] of                ! CONNECTION ID INDICATES PACKET SOURCE
;       4019              set
;       4020
;       4021              [CID_MSCP] :        IO_RETPKT ();         ! DISK MSCP (I/O TRANSFER DONE)
;       4022              [CID_DUP] :         DIO_RETPKT ();        ! DUP (I/O TRANSFER DONE)
;       4023              [CID_DRIVER] :      DR_RETPKT ();         ! MESSAGE FROM "DRIVER"
;       4024
;       4025  !           [otherwise] :       PRINTF (DBM12, .RP_ADDR [CONID]);!"CONN ID = XXXXX RECEIVED"    !JSD REV A
;       4026              tes;
;       4027
;       4028          end;                                          ! UNITL I/O DONE QUEUE IS EMPTY


                                    .SBTTL  PROC.RETPKT MULTI-DRIVE TEST ROUTINES
000000  010146                      PROC.RETPKT:
                                    MOV     R1,-(SP)                ;                                       3994
000002  023737  000000G 000000G     1$:     CMP     IODQ.IN,IODQ.OUT        ;                               4012
000010  001452                              BEQ     5$
000012  004737  000000G                     JSR     PC,OUT.IODQ             ;                               4014
000016  010037  000000G                     MOV     R0,RP.INDX
000022  010046                              MOV     R0,-(SP)                ; RP.INDX,*                     4015
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)              ;                               4016
000050  042716  177760                      BIC     #177760,(SP)
000054  004737  000000G                     JSR     PC,SET.CPAR
000060  013700  000000G                     MOV     RP.ADDR,R0              ;                               4018
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                    ;                       4021
000102  000413                          BR      4$                              ;                       4018
000104  020127  000002          2$:     CMP     R1,#2
000110  001003                          BNE     3$
000112  004737  000000V                 JSR     PC,DIO.RETPKT                   ;                       4022
000116  000405                          BR      4$                              ;                       4018
000120  020127  000003          3$:     CMP     R1,#3
000124  001002                          BNE     4$
000126  004737  000000V                 JSR     PC,DR.RETPKT                    ;                       4023
000132  022626                  4$:     CMP     (SP)+,(SP)+                     ;                       4013
000134  000722                          BR      1$                              ;                       4012
000136  012601                  5$:     MOV     (SP)+,R1                        ;                       3994
000140  000207                          RTS     PC

; Routine Size:  49 words.      Routine Base:  $CODE$ + 15134
; Maximum stack depth per invocation:  4 words
```

```
;       4029  !+
;       4030  ROUTINE DIO RETPKT : NOVALUE =
;       4031
;       4032  !*
;       4033  !       THIS ROUTINE IS CALLED BY PROC_RETPKT TO HANDLE ALL DUP I/O TRANSFER
;       4034  !       RETURN PACKETS. PROCESSING OF THESE PACKETS INCLUDES DECLARING ANY
;       4035  !       HARD ERRORS THAT MAY HAVE OCCURRED, UPDATING THE STATISTICS.
;       4036  !
;       4037  !       IMPLICIT INPUTS:
;       4038  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;       4039  !               T_ADDR - ADDRESS OF THE CURRENT UNIT'S STATISTICS BLOCK (TALLY)
;       4040  !               CST_ADDR - ADDRESS OF THE CURRENT CONTROLLER'S CST
;       4041  !               CUOFF - CST OFFSET FOR THE CURRENT UNIT
;       4042  !               L$LUN - CURRENT UNIT NUMBER
;       4043  !               CCTLR - CURRENT CONTROLLER NUMBER
;       4044  !
;       4045  !       IMPLICIT OUTPUTS
;       4046  !               CST_ADDR [.CUOFF + 3, NODUPMEDIA] - IF THIS BIT SET NO DUP EXERCISER
;       4047  !
;       4048  !-
;       4049  BEGIN
;       4050
;       4051  LOCAL   FLAG : BYTE INITIAL(BYTE(TRUE)),
;       4052          SUM2 : WORD,
;       4053          SUM : WORD;                                     ! TOTAL NUMBER OF BYTES TRANSFERRED TO/FROM A UNIT
;       4054  !PRINTX (DER18);
;       4055  FSET_UPAR ();                                         ! FIND UNIT'S ENTRY IS CST AND SET UP UNIT-RELATED DATA
;       4056
;       4057  IF .RP_ADDR [STATUS] NEQU ST_SUC                     ! IF STATUS CODE INDICATES ERROR
;       4058  THEN    BEGIN
;       4059          CST_ADDR [.CUOFF + 3, DUPERROR] = 1;          ! SET DUP ERROR FLAG
;       4060          HARD_ERROR ();
;       4061          IF .RP_ADDR [ENDCOD] EQLU (OP_ELP + OP_END) OR  ! IF ENDCODE IS EXECUTE LOCAL PROGRAM
;       4062             .RP_ADDR [ENDCOD] EQLU (OP_GDS + OP_END)     !             OR GET DUST STATUS
;       4063          THEN    BEGIN
;       4064                  CST_ADDR [.CUOFF + 3, NODUPMEDIA] = 1;          ! TURN OFF DUP EXERCISR
;       4065                  END;
;       4066          END
;       4067
;       4068  ELSE                                                  ! ELSE - I/O WAS SUCCESSFUL
;       4069
;       4070  BEGIN
;       4071
;       4072  IF .RP_ADDR [ENDCOD] EQLU (OP_GDS + OP_END)          ! IF ENDCODE IS GET DUST STATUS
;       4073  THEN    BEGIN
;       4074          IF .RP_ADDR [9,11,1,0] EQL 1
;       4075                  THEN CST_ADDR [.CUOFF + 3, D_ACTIVE] = ACTIVE           ! CONTROLLER IN AN ACTIVE STAE
;       4076                  ELSE CST_ADDR [.CUOFF + 3, D_ACTIVE] = IDLE;            ! CONTROLLER IN AN IDLE STATE
;       4077          IF .RP_ADDR [9,9,1,0] NEQ 1 THEN        ! TEST TO SEE IF CONTROLLER LOCAL PROGRAMS(PG 18 OF DUP DOC)
;       4078                  BEGIN
;       4079                  HARD_ERROR ();
;       4080                  CST_ADDR [.CUOFF + 3, NODUPMEDIA] = 1;          ! TURN OFF DUP EXERCISR
;       4081                  END;
;       4082          END;
;       4083
;       4084  IF (.RP_ADDR [ENDCOD] EQL (OP_RCD + OP_END)) AND
```

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

                                        .SBTTL  DIO.RETPKT MULTI-DRIVE TEST ROUTINES
000000  010146                  DIO.RETPKT:
                                        MOV     R1,-(SP)                        ;                                       4030
000002  112700  000001                  MOVB    #1,R0                           ; #.FLAG                                4049
000006  004737  000000V                 JSR     PC,FSET.UPAR                    ;                                       4055
000012  013701  000000G                 MOV     RP.ADDR,R1                      ;                                       4057
000016  005761  000016                  TST     16(R1)
000022  001435                          BEQ     2$
000024  013700  000000G                 MOV     CUOFF,R0                        ;                                       4059
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                   ;                                       4060
000050  013700  000000G                 MOV     RP.ADDR,R0                      ;                                       4061
000054  126027  000014  000203          CMPB    14(R0),#203
000062  001404                          BEQ     1$
000064  126027  000014  000201          CMPB    14(R0),#201                     ;                                       4062
000072  001112                          BNE     6$
000074  013700  000000G          1$:     MOV     CUOFF,R0                        ;                                       4064
000100  006300                          ASL     R0
000102  063700  000000G                 ADD     CST.ADDR,R0
000106  052760  100000  000006          BIS     #100000,6(R0)
000114  000501                          BR      6$                              ;                                       4057
000116  126127  000014  000201  2$:     CMPB    14(R1),#201                     ;                                       4072
000124  001036                          BNE     5$
000126  013700  000000G                 MOV     CUOFF,R0                        ;                                       4075
000132  006300                          ASL     R0
000134  063700  000000G                 ADD     CST.ADDR,R0
000140  032761  004000  000022          BIT     #4000,22(R1)                    ;                                       4074
000146  001404                          BEQ     3$
000150  052760  020000  000006          BIS     #20000,6(R0)                    ;                                       4075
000156  000403                          BR      4$                              ;                                       4074
000160  042760  020000  000006  3$:     BIC     #20000,6(R0)                    ;                                       4076
000166  032761  001000  000022  4$:     BIT     #1000,22(R1)                    ;                                       4077
000174  001012                          BNE     5$
000176  004737  000000V                 JSR     PC,HARD.ERROR                   ;                                       4079
000202  013700  000000G                 MOV     CUOFF,R0                        ;                                       4080
000206  006300                          ASL     R0
000210  063700  000000G                 ADD     CST.ADDR,R0
000214  052760  100000  000006          BIS     #100000,6(R0)
000222  013700  000000G          5$:     MOV     RP.ADDR,R0                      ;                                       4084
000226  126027  000014  000205          CMPB    14(R0),#205
000234  001031                          BNE     6$
```

```
000236  013700  000000G                        MOV     DUPPKT,R0                       ;                       4085
000242  042700  007777                          BIC     #7777,R0
000246  020027  060000                          CMP     R0,#60000
000252  001022                                  BNE     6$
000254  013700  000000G                        MOV     DUPPKT,R0                       ;                       4086
000260  042700  170000                          BIC     #170000,R0
000264  020027  000002                          CMP     R0,#2
000270  001013                                  BNE     6$
000272  013700  000000G                        MOV     CUOFF,R0                        ;                       4087
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                  ;                       4088
000320  013746  000000G                6$:     MOV     RP.INDX,-(SP)                   ;                       4092
000324  004737  000000G                        JSR     PC,PUT.RETPKT
000330  013700  000000G                        MOV     CCTLR,R0                        ;                       4093
000334  105360  000000G                        DECB    QIO(R0)
000340  005726                                  TST     (SP)+                           ;                       4049
000342  012601                                  MOV     (SP)+,R1                        ;                       4030
000344  000207                                  RTS     PC

; Routine Size:  115 words,    Routine Base:  $CODE$ + 15276
; Maximum stack depth per invocation:  3 words
```

```
;       4096  ROUTINE DUP_COMPARE : NOVALUE =
;       4097
;       4098  !+
;       4099  !       THIS ROUTINE IS CALLED BY DIO_RETPKT WHEN THE RECEIVE DATA COMMAND
;       4100  !       IS BEING PROCESSED. THIS COMMAND COMPARES THE WRITTEN BUFFER WITH
;       4101  !       THE PATERN WORD GIVEN IN SEND DATA COMMAND.  FOR EVERY WORD COMPARED
;       4102  !       THE ROUTINE INCREMENTS THE TALLY TABLE.  IF THE COMPARE SHOWS AN
;       4103  !       ERROR. THE DBN HARD ERROR COUNTER WILL BE INCREMENTED AND THE
;       4104  !       THE DBN NUMBER AND BYTE COUNT WILL BE PRINTED.
;       4105  !
;       4106  !       IMPLICIT INPUTS:
;       4107  !               S_PATTERN       ! THE SAVED PATTERN WRITTEN TO THE DBN'S
;       4108  !               S_DUPPKT        ! THE POINTER FOR DUP BUFFER
;       4109  !               T_ADDR          ! THE ADDRESS OF THE TALLY TABLE FOR THIS UNIT
;       4110  !               CST_ADDR        ! THE ADDRESS OF PRESENT CONTROLLER STATUS TABLE
;       4111  !-
;       4112  BEGIN
;       4113
;       4114  OWN
;       4115          COUNT : WORD;
;       4116
;       4117  !PRINTX (DER19);
;       4118  S_DUPPKT = 0;
;       4119  INCR COUNT FROM 1 TO 256 DO     !INDEX PIONTER FOR DATA STORED IN MSCP ENV PACKET
;       4120          BEGIN
;       4121          S_DUPPKT = .S_DUPPKT + 2;               ! INITIALLY THIS SKIPS THE FIRST WORD OF DUPPKT
;       4122          IF .(DUPPKT + .S_DUPPKT) NEQ .S_PATTERN THEN    !IF THE CONTENTS OF DBN DOESN T EQUAL PATTERN
;       4123                  BEGIN
;       4124                  ERRHRD (46, EH_10, EMS_22);     !LIST ERROR
;       4125                  EXITLOOP;
;       4126                  END;
;       4127          END;                                    !GO THROUGH ALL DBN WORDS
;       4128  END;                                            !END ROUTINE DUP COMPARE
```

```
NRQAMS          MD RX EXERCISER                               15 Dec 1985 10:55:57    VAX 11 BLISS-16 V3.???
V01.C           MULTI DRIVE TEST ROUTINE                      14 Dec 1985 11:55:15    SPIDERS [ERS:[DOUCETTE...

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

                                          .SBTTL  DUP.COMPARE MULTI DRIVE TEST ROUTINE
015644                                    .PSECT  $CODE$,  RO

000000  010146                     DUP.COMPARE:
                                          MOV     R1,-(SP)                     ;                              4096
000002  005037  000000G                   CLR     S.DUPPKT                     ;                              4118
000006  012701  000400                    MOV     #400,R1                      ; *,COUNT                      4119
000012  062737  000002  000000G    1$:    ADD     #2,S.DUPPKT                  ;                              4121
000020  013700  000000G                   MOV     S.DUPPKT,R0                  ;                              4122
000024  026037  000000G 000000G           CMP     DUPPKT(R0),S.PATTERN
000032  001405                            BEQ     2$
000034  104456                            TRAP    56                           ;                              4124
000036  000056                            .WORD   56
000040  000000G                           .WORD   EM.10
000042  000000G                           .WORD   EMS.22
000044  000402                            BR      3$                           ;                              4125
000046  005301                     2$:    DEC     R1                           ; COUNT                        4119
000050  001360                            BNE     1$
000052  012601                     3$:    MOV     (SP)+,R1                     ;                              4096
000054  000207                            RTS     PC

; Routine Size:  23 words,      Routine Base:  $CODE$ + 15644
; Maximum stack depth per invocation:  3 words
```

```
;       4129  routine IO_RETPKT : novalue =
;       4130
;       4131  !+
;       4132  !       THIS ROUTINE IS CALLED BY PROC_RETPKT TO HANDLE ALL I/O TRANSFER
;       4133  !       RETURN PACKETS. PROCESSING OF THESE PACKETS INCLUDE  DECLARING ANY
;       4134  !       HARD ERRORS THAT MAY HAVE OCCURRED, UPDATING THE STATISTICS, AND
;       4135  !       PERFORMING HOST WRITE-COMPARES IF REQUIRED.
;       4136  !
;       4137  !       IMPLICIT INPUTS:
;       4138  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;       4139  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;       4140  !               T_ADDR - ADDRESS OF CURRENT UNIT'S STATISTICS BLOCK (TALLY)
;       4141  !               CCTLR - CURRENT CONTROLLER NUMBER
;       4142  !               L$LUN - CURRENT UNIT NUMBER
;       4143  !
;       4144
;       4145      begin
;       4146
;       4147      local
;       4148          FLAG : byte initial (byte (TRUE));
;       4149
;       4150      FSET_UPAR ();                                   ! FIND UNIT'S ENTRY IS CST AND SET UP UNIT-RELATED DATA
;       4151      ST_CODE = .RP_ADDR [STSCOD];                    ! GET STATUS CODE FROM RETPKT
;       4152      SB_CODE = .RP_ADDR [SUBCOD];                    ! GET SUB CODE, IF ANY
;       4153
;       4154      if (.ST_CODE neq ST_SUC)              ! IF STATUS CODE INDICATES ERROR
;       4155      then
;       4156          begin
;       4157          HARD_ERROR ();                              ! UPDATE ERROR COUNT
;       4158          COMPARE_DATA = FALSE;                       ! NO POINT IN DOING HOST COMPARES ON ERROR
;       4159
;       4160          if (.ST_CODE neq ST_OFL) and                ! DROP UNIT IF ERROR COUNTS EXCEEDS LIMIT
;       4161             (.st_code neq ST_AVL) and
;       4162             (.T_ADDR [ERR_HARD] gequ .SWP_ERROR)
;       4163          then
;       4164              begin
;       4165              DUR [.L$LUN] = DU_HERR;                 ! LOAD REASON FOR DROPPING UNIT
;       4166              DODU (.L$LUN);                          ! DROP UNIT
;       4167              end;
;       4168
;       4169          end;
;       4170
;       4171      if .ST_CODE eql ST_SUC                          ! IF I/O WAS SUCCESSFUL
;       4172      then
;       4173          begin
;       4174          UPD_IO_TALLY ();                            ! UPDATE I/O TALLY (STATISTICS)
;       4175
;       4176          if .RP_ADDR [ENDCOD] eql (OP_WRT or OP_END)
;       4177          then COMPARE_DATA = TRUE;                   ! HOST COMPARES MAY BE ALLOWED IF NO FURTHER ERRORS
;       4178
;       4179          if (BIT_TST (SWP_FLAGS, SWF_HWC)) and       ! IF HOST IS DOING WRITE-COMPARES
;       4180             (.COMPARE_DATA)
;       4181          then
;       4182              FLAG = HOST_WRT_CHK ();                 ! SAVE I/O PACKET OR DO WRITE CHECK
;       4183
;       4184          end;
```

```
;        4185
;        4186      ;f .FLAG                                  ! IF FLAG IS STILL TRUE
;        4187      then
;        4188          SWEEP ();                             ! DEALLOCATE BUFFER(S) AND RETPKT(S)
;        4189
;        4190      QIO [.CCTLR] = .QIO [.CCTLR]   1;         ! DECREMENT NO. OF OUTSTANDING QIOS
;        4191      end;                                      ! ROUTINE IO_RETPKT

                                        .SBTTL  IO.RETPKT MULTI DRIVE TEST ROUTINES
000000  004137  000000G           IO.RETPKT:
                                        JSR     R1,$SAVE2                       ;                       4129
000004  112701  000001                  MOVB    #1,R1                           ; *.FLAG                4145
000010  004737  000000V                 JSR     PC,FSET.UPAR                    ;                       4150
000014  013700  000000G                 MOV     RP.ADDR,R0                      ;                       4151
000020  116037  000016  000000G         MOVB    16(R0),ST.CODE
000026  042737  177740  000000G         BIC     #177740,ST.CODE
000034  016002  000016                  MOV     16(R0),R2                       ;                       4152
000040  006202                          ASR     R2
000042  006202                          ASR     R2
000044  006202                          ASR     R2
000046  006202                          ASR     R2
000050  006202                          ASR     R2
000052  042702  174000                  BIC     #174000,R2
000056  010237  000000G                 MOV     R2,SB.CODE
000062  005737  000000G                 TST     ST.CODE                         ;                       4154
000066  001433                          BEQ     2$
000070  004737  000000V                 JSR     PC,HARD.ERROR                   ;                       4157
000074  105037  001240'                 CLRB    COMPARE.DATA                    ;                       4158
000100  023727  000000G 000003         CMP     ST.CODE,#3                      ;                       4160
000106  001420                          BEQ     1$
000110  023727  000000G 000004         CMP     ST.CODE,#4                      ;                       4161
000116  001414                          BEQ     1$
000120  013700  000000G                 MOV     T.ADDR,R0                       ;                       4162
000124  026037  000014  000000G         CMP     14(R0),SWP.ERROR
000132  103406                          BLO     1$
000134  013700  000000G                 MOV     L$LUN,R0                        ;                       4165
000140  112760  000004  000000G         MOVB    #4,DUR(R0)
000146  104451                          TRAP    51                              ;                       4166
```

```
000150  005737  000000G                1$:     TST     ST.CODE                         ;                       4171
000154  001026                                 BNE     4$
000156  004737  000000V                2$:     JSR     PC,UPD.IO.TALLY                 ;                       4174
000162  013700  000000G                        MOV     RP.ADDR,R0                      ;                       4176
000166  126027  000014  000242                CMPB    14(R0),#242
000174  001003                                 BNE     3$
000176  112737  000001  001240'               MOVB    #1,COMPARE.DATA                 ;                       4177
000204  032737  000040  000000G        3$:     BIT     #40,SWP.FLAGS                   ;                       4179
000212  001407                                 BEQ     4$
000214  032737  000001  001240'               BIT     #1,COMPARE.DATA                 ;                       4180
000222  001403                                 BEQ     4$
000224  004737  000000V                        JSR     PC,HOST.WRT.CHK                 ;                       4182
000230  110001                                 MOVB    R0,R1                           ; *,FLAG
000232  006001                        4$:     ROR     R1                              ; FLAG                  4186
000234  103002                                 BCC     5$
000236  004737  000000V                        JSR     PC,SWEEP                        ;                       4188
000242  013700  000000G                5$:     MOV     CCTLR,R0                        ;                       4190
000246  105360  000000G                        DECB    QIO(R0)
000252  000207                                 RTS     PC                              ;                       4129

; Routine Size:  86 words,      Routine Base:  $CODE$ + 15722
; Maximum stack depth per invocation:  5 words
```

```
;        4192  routine FSET_UPAR : novalue =
;        4193
;        4194  !+
;        4195  !       THIS ROUTINE IS CALLED BY IO_RETPKT AND OTHERS TO SEARCH THE CURRENT
;        4196  !       CONTROLLER STATUS TABLE (CST) FOR THE DISK ADDRESS WHICH IS
;        4197  !       CONTAINED IN THE CURRENT RETURN PACKET. WHEN FOUND, THE OFFSET INTO THE
;        4198  !       CST IS USED AS INPUT TO SET UPAR, WHICH SETS UP CURRENT UNIT RELATED
;        4199  !       DATA PARAMETERS.
;        4200  !
;        4201  !       IMPLICIT INPUTS:
;        4202  !               RP_ADDR   ADDRESS OF CURRENT RETURN PACKET
;        4203  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;        4204  !-
;        4205
;        4206      begin
;        4207
;        4208      incr OFFSET from (0 + OF_UN) to (3 * UNIT_SIZE + OF_UN) by UNIT_SIZE do     ! FOR EACH UNIT IN CST
;        4209
;        4210          if .CST_ADDR [.OFFSET, D_DISK_NUM] eql .RP_ADDR [DISK]
;        4211                                                  ! IF RETPKT UNIT NUMBER MATCHES CST ENTRY
;        4212          then
;        4213              begin
;        4214              SET_UPAR (.OFFSET);                 ! SET UP UNIT-RELATED DATA
;        4215              return;                             ! DONE
;        4216              end;
;        4217
;        4218  !     PRINTF (DBM19, .RP_ADDR [DISK], .CCTLR);    ! "CAN'T FIND DISK XXX IN CST X"        !JSD REV A
;        4219      end;                                        ! ROUTINE FSET_UPAR

                                .SBTTL  FSET.UPAR MULTI-DRIVE TEST ROUTINES
000000  004137  000000G    FSET.UPAR:
                                JSR     R1,$SAVE4                       ;                               4192
000004  012702  000003              MOV     #3,R2                           ; *,OFFSET                      4208
000010  010201              1$:     MOV     R2,R1                           ; OFFSET,*                      4210
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,*                      4214
000044  004737  000000G             JSR     PC,SET.UPAR
000050  005726                      TST     (SP)+                           ;                               4210
000052  000207                      RTS     PC                              ;                               4213
000054  062702  000005      2$:     ADD     #5,R2                           ; *,OFFSET                      4208
000060  020227  000022              CMP     R2,#22                          ; OFFSET,*
000064  003751                      BLE     1$
000066  000207                      RTS     PC                              ;                               4192

; Routine Size:  28 words,      Routine Base:  $CODE$ + 16176
; Maximum stack depth per invocation:  7 words
```

```
;       4220  routine HARD_ERROR : novalue =
;       4221
;       4222  !+
;       4223  !       THIS ROUTINE IS CALLED BY IO_RETPKT, DIO_RETPKT, AND OTHERS TO INCREMENT THE HARD
;       4224  !       ERROR STATISTIC FIELD FOR THE CURRENT UNIT. IF THE HARD ERROR COUNT
;       4225  !       HAS EXCEEDED THE OPERATOR-SPECIFIED LIMIT, THEN THE UNIT IS DROPPED
;       4226  !       FROM TESTING.
;       4227  !
;       4228  !       IMPLICIT INPUTS:
;       4229  !               L$LUN - CURRENT UNIT NUMBER
;       4230  !               CST_ADDR - ADDRESS OF CURRENT CONTROLLER'S CST
;       4231  !               CUOFF - CST OFFSET FOR CURRENT UNIT
;       4232  !               T_ADDR - ADDRESS OF CURRENT UNIT'S STATISTICS BLOCK (TALLY)
;       4233  !-
;       4234
;       4235      begin
;       4236      T_ADDR [ERR_HARD] = .T_ADDR [ERR_HARD] + 1;                       ! INCREMENT UNIT'S HARD ERROR COUNT
;       4237  if .RP_ADDR [CONID] EQL CID_MSCP
;       4238  THEN
;       4239      selectoneu .ST_CODE of
;       4240          set
;       4241
;       4242          [ST_SUC]:       if .SB_CODE neq 0                           ! SUCCESS WITH NON ZERO SUB CODE
;       4243                          then
;       4244                              begin
;       4245
;       4246                              if .SB_CODE eql 4
;       4247                              then
;       4248                                  begin
;       4249                                  T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1
;       4250                                  end
;       4251                              else
;       4252                                  T_ADDR [ERR_HRD_HST] = .T ADDR [ERR_HRD_HST] + 1;
;       4253
;       4254                              ERRHRD (44, EGH_30, EMS_30);
;       4255                              end;
;       4256
;       4257          [ST_CMD]:       begin                                       ! INVALID COMMAND
;       4258                          T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;       4259                          ERRHRD (31, EGH_30, EMS_30);
;       4260                          end;
;       4261
;       4262          [ST_ABO]:       begin                                       ! COMMMAND ABORTED
;       4263                          T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;       4264                          ERRHRD (32, EGH_30, EMS_30);
;       4265                          end;
;       4266
;       4267          [ST_OFL] :      begin                                       ! OFFLINE
;       4268                          T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;       4269                          ERRDF (18, EGD_18, EMS_18);
;       4270                          DUR [.L$LUN] = DU_DFATAL;                   ! DEVICE FATAL ERROR
;       4271                          DODU (.L$LUN);                              ! DROP UNIT
;       4272                          end;
;       4273
;       4274          [ST_AVL] :      begin
;       4275                          T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
```

```
;        4276                    ERRDF (24, EGD_18, EMS_18);
;        4277                    DUR [.L$LUN] = DU_AV;                           ! DEVICE WENT AVAILABLE STATE
;        4278                    DODU (.L$LUN);                                  ! DROP UNIT
;        4279                    end;
;        4280
;        4281            [ST_MFE]:       begin                                   ! MEDIA FORMAT ERROR
;        4282                    T_ADDR [ERR_HRD_SEK] = .T_ADDR [ERR_HRD_SEK] + 1;
;        4283                    ERRHRD (35, EGH_30, EMS_30);
;        4284                    end;
;        4285
;        4286            [ST_WPT]:       begin                                   ! DEVICE WRITE PROTECTED
;        4287
;        4288                    if .SB_CODE eql 128
;        4289                    then
;        4290                        T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1
;        4291                    else
;        4292                        T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4293
;        4294                    ERRHRD (36, EGH_30, EMS_30);
;        4295                    end;
;        4296
;        4297            [ST_CMP]:       begin                                   ! COMPARE ERROR
;        4298                    T_ADDR [ERR_HRD_DAT] = .T_ADDR [ERR_HRD_DAT] + 1;
;        4299                    ERRHRD (37, EGH_30, EMS_30);
;        4300                    end;
;        4301
;        4302            [ST_DAT]:       begin                                   ! DATA ERROR
;        4303
;        4304                    if .SB_CODE eql 2
;        4305                    then
;        4306                        T_ADDR [ERR_HRD_SEK] = .T_ADDR [ERR_HRD_SEK] + 1
;        4307                    else
;        4308                        T_ADDR [ERR_HRD_DAT] = .T_ADDR [ERR_HRD_DAT] + 1;
;        4309
;        4310                    ERRHRD (38, EGH_30, EMS_30);
;        4311                    end;
;        4312
;        4313            [ST_HST]:       begin                                   ! HOST ACCESS ERROR
;        4314                    T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;        4315                    ERRHRD (39, EGH_30, EMS_30);
;        4316                    end;
;        4317
;        4318            [ST_CNT]:       begin                                   ! CONTROLLER ERROR
;        4319                    T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;        4320                    ERRHRD (40, EGH_30, EMS_30);
;        4321                    end;
;        4322
;        4323            [ST_DRV]:       begin                                   ! DRIVE ERROR
;        4324
;        4325                    if .SB_CODE eql 3
;        4326                    then
;        4327                        T_ADDR [ERR_HRD_SEK] = .T_ADDR [ERR_HRD_SEK] + 1
;        4328                    else
;        4329                        T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD DRV] + 1;
;        4330
;        4331                    ERRHRD (41, EGH_30, EMS_30);
```

```
;       4332                            end;
;       4333
.       4334            [ST_DIA]:       begin                                   ! MESSAGE FROM INTERNAL DIAGNOSTIC
.       4335                            T_ADDR [ERR HRD DRV] = .T_ADDR [ERR HRD DRV] + 1;
.       4336                            ERRHRD (43, EGH 30, EMS 30);
.       4337                            end;
.       4338
.       4339            [otherwise]:    begin                                   ! PRINT STATUS CODE IF NO MATCH
.       4340                            C_ERR_TBL [.CCTLR, C ERR HRD] = .C ERR_TBL [.CCTLR, C ERR HRD] + 1;
.       4341                            ERRHRD (45, EGH_30, EMS 30);
.       4342                            end;
.       4343
.       4344            tes;
.       4345    if .RP_ADDR [CONID] EQL CID_DUP
.       4346    THEN
;       4347
;       4348        selectoneu .RP_ADDR [STSCOD] of
;       4349                SET
.       4350                [%o'0']             : begin                         ! if status code succesful
;       4351                    IF .RP ADDR [ENDCOD] EQLU (OP_GDS + OP_END)     ! IF ENDCODE IS GET DUST STATUS
;       4352                    THEN
;       4353                            IF .RP_ADDR [9,9,1,0] NEQ 1 THEN        ! TEST TO SEE IF CONTROLLER LOCAL PROGRAMS(PG 18 OF
DUP DOC)
;       4354                                BEGIN
;       4355                                ERRHRD (47, EH_12, EMS_30);     ! UNABLE TO LOAD LOCAL CONTROLLER DUP MEDIA(PROGRAMS
)
;       4356                                T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
;       4357                                END;
;       4358                    if .ouppkt [dupmesg] eql 3
;       4359                    then
;       4360                            if (.DUPPKT [DUPTYPE] eql 5)
;       4361                            then
;       4362                                    begin
;       4363                                    DUR [.L$LUN] = DU_DFATAL;
;       4364                                    DODU (.L$LUN);                  ! FATAL DEVICE ERROR DROP UNIT);
.       4365                                                                    ! SET REASON FOR DROPPING UNIT
;       4366                                    selectoneu .DUPPKT [DUPMESG] of
;       4367                                            SET
;       4368                                            [%o'1'] : begin
;       4369                                                    errhrd (48, eh_13, ems_30); ! illegal unit number
.       4370                                                    T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;       4371                                                    end;
.       4372                                            [%o'2'] : begin
;       4373                                                    errhrd (49, eh_13, ems_30); ! illegal relative or physical block #
.       4374                                                    T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
.       4375                                                    end;
;       4376                                            [%o'3'] : begin
;       4377                                                    errhrd (50, eh_13, ems_30); ! device error
.       4378                                                    T_ADDR [ERR_HRD_DRV] = .T_ADDR [ERR_HRD_DRV] + 1;
.       4379                                                    end;
.       4380                                            [%o'4'] : begin
.       4381                                                    errhrd (51, eh_13, ems_30); ! zero lenght message
.       4382                                                    T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR HRD HST] + 1;
.       4383                                                    end;
.       4384                                            tes;
.       4385                                    end;
        4386                            end;
        4387                [%o'1']         : begin
```

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


                        .SBTTL  HARD.ERROR MULTI-DRIVE TEST ROUTINES
000000  004137  000000G  HARD.ERROR:
                        JSR     R1,$SAVE3                       ;       4220
000004  013701  000000G         MOV     T.ADDR,R1               ;       4236
000010  005261  000014          INC     14(R1)
000014  013760  000000G         MOV     RP.ADDR,R0              ;       4237
000020  105760  000003          TSTB    3(R0)
000024  001157                  BNE     12$
000026  013700  000000G         MOV     ST.CODE,R0              ;       4239
000032  001022                  BNE     3$
000034  013702  000000G         MOV     SB.CODE,R2              ;       4242
000040  001574                  BEQ     16$
000042  012703  000062          MOV     #62,R3                  ;       4249
000046  060103                  ADD     R1,R3
000050  020227  000004          CMP     R2,#4                   ;       4246
000054  001002                  BNE     1$
000056  105213                  INCB    (R3)                    ;       4249
000060  000402                  BR      2$                      ;       4246
000062  105263  000001  1$:     INCB    1(R3)                   ;       4252
000066  104456          2$:     TRAP    56                      ;       4254
000070  000054                  .WORD   54
000072  000000G                 .WORD   EGH.30
000074  000000G                 .WORD   EMS.30
000076  000567                  BR      18$                     ;       4239
```

```
000100  020027  000001                  3$:   CMP     R0,#1
000104  001007                                BNE     4$
000106  105261  000063                        INCB    63(R1)                  ;                4258
000112  104456                                TRAP    56                      ;                4259
000114  000037                                .WORD   37
000116  000000G                               .WORD   EGH.30
000120  000000G                               .WORD   EMS.30
000122  000567                                BR      20$                     ;                4239
000124  020027  000002                  4$:   CMP     R0,#2
000130  001007                                BNE     5$
000132  105261  000062                        INCB    62(R1)                  ;                4263
000136  104456                                TRAP    56                      ;                4264
000140  000040                                .WORD   40                                        !
000142  000000G                               .WORD   EGH.30
000144  000000G                               .WORD   EMS.30
000146  000576                                BR      24$                     ;                4239
000150  020027  000003                  5$:   CMP     R0,#3
000154  001015                                BNE     6$
000156  105261  000062                        INCB    62(R1)                  ;                4268
000162  104455                                TRAP    55                      ;                4269
000164  000022                                .WORD   22
000166  000000G                               .WORD   EGD.18
000170  000000G                               .WORD   EMS.18
000172  013700  000000G                       MOV     L$LUN,R0                ;                4270
000176  112760  000005  000000G               MOVB    #5,DUR(R0)
000204  104451                                TRAP    51                      ;                4271
000206  000570                                BR      26$                     ;                4239
000210  020027  000004                  6$:   CMP     R0,#4
000214  001015                                BNE     7$
000216  105261  000062                        INCB    62(R1)                  ;                4275
000222  104455                                TRAP    55                      ;                4276
000224  000030                                .WORD   30
000226  000000G                               .WORD   EGD.18
000230  000000G                               .WORD   EMS.18
000232  013700  000000G                       MOV     L$LUN,R0                ;                4277
000236  112760  000013  000000G               MOVB    #13,DUR(R0)
000244  104451                                TRAP    51                      ;                4278
000246  000562                                BR      28$                     ;                4239
000250  020027  000005                  7$:   CMP     R0,#5
000254  001007                                BNE     8$
000256  105261  000060                        INCB    60(R1)                  ;                4282
000262  104456                                TRAP    56                      ;                4283
000264  000043                                .WORD   43
000266  000000G                               .WORD   EGH.30
000270  000000G                               .WORD   EMS.30
000272  000550                                BR      28$                     ;                4239
000274  020027  000006                  8$:   CMP     R0,#6
000300  001020                                BNE     11$
000302  012702  000062                        MOV     #62,R2                  ;                4290
000306  060102                                ADD     R1,R2
000310  023727  000000G 000200                CMP     $B.CODE,#200            ;                4288
000316  001003                                BNE     9$
000320  105262  000001                        INCB    1(R2)                   ;                4290
000324  000401                                BR      10$                     ;                4288
000326  105212                          9$:   INCB    (R2)                    ;                4292
000330  104456                          10$:  TRAP    56                      ;                4294
```

```
000332  000044                          .WORD   44
000334  000000G                         .WORD   EGH.30
000336  000000G                         .WORD   EMS.30
000340  000525                          BR      28$                             ;                       4239
000342  020027  000007          11$:    CMP     R0,#7
000346  001007                          BNE     13$
000350  105261  000061                  INCB    61(R1)                          ;                       4298
000354  104456                          TRAP    56                              ;                       4299
000356  000045                          .WORD   45
000360  000000G                         .WORD   EGH.30
000362  000000G                         .WORD   EMS.30
000364  000513          12$:    BR      28$                             ;                       4239
000366  020027  000010          13$:    CMP     R0,#10
000372  001020                          BNE     17$
000374  012702  000060                  MOV     #60,R2                          ;                       4306
000400  060102                          ADD     R1,R2
000402  023727  000000G 000002          CMP     $B.CODE,#2                      ;                       4304
000410  001002                          BNE     14$
000412  105212                          INCB    (R2)                            ;                       4306
000414  000402                          BR      15$                             ;                       4304
000416  105262  000001          14$:    INCB    1(R2)                           ;                       4308
000422  104456                  15$:    TRAP    56                              ;                       4310
000424  000046                          .WORD   46
000426  000000G                         .WORD   EGH.30
000430  000000G                         .WORD   EMS.30
000432  000470                  16$:    BR      28$                             ;                       4239
000434  020027  000011          17$:    CMP     R0,#11
000440  001007                          BNE     19$
000442  105261  000063                  INCB    63(R1)                          ;                       4314
000446  104456                          TRAP    56                              ;                       4315
000450  000047                          .WORD   47
000452  000000G                         .WORD   EGH.30
000454  000000G                         .WORD   EMS.30
000456  000456                  18$:    BR      28$                             ;                       4239
000460  020027  000012          19$:    CMP     R0,#12
000464  001007                          BNE     21$
000466  105261  000062                  INCB    62(R1)                          ;                       4319
000472  104456                          TRAP    56                              ;                       4320
000474  000050                          .WORD   50
000476  000000G                         .WORD   EGH.30
000500  000000G                         .WORD   EMS.30
000502  000444                  20$:    BR      28$                             ;                       4239
000504  020027  000013          21$:    CMP     R0,#13
000510  001016                          BNE     25$
000512  023727  000000G 000003          CMP     $B.CODE,#3                      ;                       4325
000520  001003                          BNE     22$
000522  105261  000060                  INCB    60(R1)                          ;                       4327
000526  000402                          BR      23$                             ;                       4325
000530  105261  000062          22$:    INCB    62(R1)                          ;                       4329
000534  104456                  23$:    TRAP    56                              ;                       4331
000536  000051                          .WORD   51
000540  000000G                         .WORD   EGH.30
000542  000000G                         .WORD   EMS.30
000544  000423                  24$:    BR      28$                             ;                       4239
000546  020027  000037          25$:    CMP     R0,#37
000552  001007                          BNE     27$
```

```
000554  105261  000062                          INCB    62(R1)                  ;                       4335
000560  104456                                  TRAP    56                      ;                       4336
000562  000053                                  .WORD   53
000564  000000G                                 .WORD   EGH.30
000566  000000G                                 .WORD   EMS.30
000570  000411                          26$:    BR      28$                     ;                       4239
000572  013700  000000G                 27$:    MOV     CCTLR,R0                ;                       4340
000576  006300                                  ASL     R0
000600  105260  000000G                         INCB    C.ERR.TBL(R0)
000604  104456                                  TRAP    56                      ;                       4341
000606  000055                                  .WORD   55
000610  000000G                                 .WORD   EGH.30
000612  000000G                                 .WORD   EMS.30
000614  013700  000000G                 28$:    MOV     RP.ADDR,R0              ;                       4345
000620  126027  000003  000002                  CMPB    3(R0),#2
000626  001107                                  BNE     33$
000630  116001  000016                          MOVB    16(R0),R1               ;                       4348
000634  042701  177740                          BIC     #177740,R1
000640  001110                                  BNE     34$
000642  126027  000014  000201                  CMPB    14(R0),#201             ;                       4351
000650  001014                                  BNE     29$
000652  032760  001000  000022                  BIT     #1000,22(R0)            ;                       4353
000660  001010                                  BNE     29$
000662  104456                                  TRAP    56                      ;                       4355
000664  000057                                  .WORD   57
000666  000000G                                 .WORD   EH.12
000670  000000G                                 .WORD   EMS.30
000672  013700  000000G                         MOV     T.ADDR,R0               ;                       4356
000676  105260  000062                          INCB    62(R0)
000702  013700  000000G                 29$:    MOV     DUPPKT,R0               ;                       4358
000706  042700  170000                          BIC     #170000,R0
000712  020027  000003                          CMP     R0,#3
000716  001162                                  BNE     43$
000720  013700  000000G                         MOV     DUPPKT,R0               ;                       4360
000724  042700  007777                          BIC     #7777,R0
000730  020027  050000                          CMP     R0,#50000
000734  001153                                  BNE     43$
000736  013700  000000G                         MOV     L$LUN,R0                ;                       4363
000742  112760  000005  000000G                 MOVB    #5,DUR(R0)
000750  104451                                  TRAP    51                      ;                       4364
000752  013700  000000G                         MOV     DUPPKT,R0               ;                       4366
000756  042700  170000                          BIC     #170000,R0
000762  020027  000001                          CMP     R0,#1
000766  001005                                  BNE     30$
000770  104456                                  TRAP    56                      ;                       4369
000772  000060                                  .WORD   60
000774  000000G                                 .WORD   EH.13
000776  000000G                                 .WORD   EMS.30
001000  000513                                  BR      41$                     ;                       4370
001002  020027  000002                  30$:    CMP     R0,#2                   ;                       4366
001006  001005                                  BNE     31$
001010  104456                                  TRAP    56                      ;                       4373
001012  000061                                  .WORD   61
001014  000000G                                 .WORD   EH.13
001016  000000G                                 .WORD   EMS.30
001020  000503                                  BR      41$                     ;                       4374
```

```
001022  020027  000003          31$:    CMP     R0,#3                           ;                       4366
001026  001005                          BNE     32$
001030  104456                          TRAP    56                              ;                       4377
001032  000062                          .WORD   62
001034  000000G                         .WORD   EH.13
001036  000000G                         .WORD   EMS.30
001040  000457                          BR      39$                             ;                       4378
001042  020027  000004          32$:    CMP     R0,#4                           ;                       4366
001046  001106                  33$:    BNE     43$
001050  104456                          TRAP    56                              ;                       4381
001052  000063                          .WORD   63
001054  000000G                         .WORD   EH.13
001056  000000G                         .WORD   EMS.30
001060  000463                          BR      41$                             ;                       4382
001062  020127  000001          34$:    CMP     R1,#1                           ;                       4348
001066  001005                          BNE     35$
001070  104456                          TRAP    56                              ;                       4388
001072  000064                          .WORD   64
001074  000000G                         .WORD   EH.7
001076  000000G                         .WORD   EMS.30
001100  000437                          BR      39$                             ;                       4389
001102  020127  000002          35$:    CMP     R1,#2                           ;                       4348
001106  001005                          BNE     36$
001110  104456                          TRAP    56                              ;                       4392
001112  000065                          .WORD   65
001114  000000G                         .WORD   EH.7
001116  000000G                         .WORD   EMS.30
001120  000427                          BR      39$                             ;                       4393
001122  020127  000003          36$:    CMP     R1,#3                           ;                       4348
001126  001005                          BNE     37$
001130  104456                          TRAP    56                              ;                       4396
001132  000066                          .WORD   66
001134  000000G                         .WORD   EH.7
001136  000000G                         .WORD   EMS.30
001140  000433                          BR      41$                             ;                       4397
001142  020127  000004          37$:    CMP     R1,#4                           ;                       4348
001146  001005                          BNE     38$
001150  104456                          TRAP    56                              ;                       4400
001152  000067                          .WORD   67
001154  000000G                         .WORD   EH.7
001156  000000G                         .WORD   EMS.30
001160  000423                          BR      41$                             ;                       4401
001162  020127  000005          38$:    CMP     R1,#5                           ;                       4348
001166  001011                          BNE     40$
001170  104456                          TRAP    56                              ;                       4404
001172  000070                          .WORD   70
001174  000000G                         .WORD   EH.7
001176  000000G                         .WORD   EMS.30
001200  013700  000000G         39$:    MOV     T.ADDR,R0                       ;                       4405
001204  105260  000062                  INCB    62(R0)
001210  000207                          RTS     PC                              ;                       4348
001212  020127  000006          40$:    CMP     R1,#6
001216  001011                          BNE     42$
001220  104456                          TRAP    56                              ;                       4408
001222  000071                          .WORD   71
001224  000000G                         .WORD   EH.7
```

```
001226  000000G                            .WORD   EMS.50
001230  015700  000000G           41$:     MOV     T.ADDR,R0                          ;         4409
001234  105260  000065                     INCB    65(R0)
001240  000207                             RTS     PC                                 ;         4348
001242  104456                    42$:     TRAP    56                                 ;         4412
001244  000072                             .WORD   72
001246  000000G                            .WORD   EM.8
001250  000000G                            .WORD   EMS.50
001252  015700  000000G                    MOV     CCTLR,R0                           ;         4413
001256  006300                             ASL     R0
001260  105260  000000G                    INCB    C.ERR.TBL(R0)
001264  000207                    43$:     RTS     PC                                 ;         4220

; Routine Size:  347 words,     Routine Base:  $CODE$ + 16266
; Maximum stack depth per invocation:  6 words
```

```
4418  routine UPD_IO_TALLY : novalue =
4419
4420  !+
4421  !       THIS ROUTINE IS CALLED FROM IO_RETPKT FOR ALL I/O TRANSFER RETURN
4422  !       PACKETS WITH "SUCCESS" STATUS CODES. ITS PURPOSE IS TO UPDATE ALL THE
4423  !       APPROPRIATE STATISTICAL FIELDS FOR THE CURRENT UNIT. A CHECK IS ALSO
4424  !       MADE ON THE TOTAL NUMBER OF BYTES TRANSFERRED THUS FAR; IF THE
4425  !       OPERATOR-SPECIFIED LIMIT HAS BEEN REACHED, THEN THE UNIT IS DROPPED.
4426  !
4427  !       IMPLICIT INPUTS:
4428  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
4429  !               T_ADDR - ADDRESS OF THE CURRENT UNIT'S STATISTICS BLOCK (TALLY)
4430  !               CST_ADDR - ADDRESS OF THE CURRENT CONTROLLER'S CST
4431  !               CUOFF - CST OFFSET FOR THE CURRENT UNIT
4432  !               LUN - CURRENT UNIT NUMBER
4433  !-
4434
4435      begin
4436
4437      local
4438          THOUSANDS : word,
4439          MILLIONS : word;                                        ! TOTAL NUMBER OF BYTES TRANSFERRED TO/FROM A UNIT
4440
4441      if .RP_ADDR [ENDCOD] eql (OP_RD or OP_END)                  ! IF ENDCODE IS READ
4442      then
4443          begin
4444          T_ADDR [TOT_READS_LO] = .T_ADDR [TOT_READS_LO] + 1;     ! INCREMENT NO. OF READS AND ADD BYTE COUNT
4445          T_ADDR [BYTES_READ_LO] = .T_ADDR [BYTES_READ_LO] + .RP_ADDR [BCNT_LO];
4446          T_ADDR [TOT_BYT_RED_LO] = .T_ADDR [TOT_BYT_RED_LO] + .RP_ADDR [BCNT_LO];
4447          OVF_CHK (T_ADDR [TOT_READS_LO]);                        ! CHECK FOR FIELD OVERFLOW
4448          OVF_CHK (T_ADDR [BYTES_READ_LO]);                       !
4449          OVF_CHK (T_ADDR [TOT_BYT_RED_LO]);                      !
4450          end
4451      else
4452
4453          if .RP_ADDR [ENDCOD] eql (OP_WRT or OP_END)             ! IF ENDCODE IS WRITE
4454          then
4455              begin
4456              T_ADDR [TOT_WRITES_LO] = .T_ADDR [TOT_WRITES_LO] + 1;       ! INCREMENT NO. OF WRITES AND ADD BYTE COUNT
4457              T_ADDR [BYTES_WRIT_LO] = .T_ADDR [BYTES_WRIT_LO] + .RP_ADDR [BCNT_LO];
4458              T_ADDR [TOT_BYT_WRT_LO] = .T_ADDR [TOT_BYT_WRT_LO] + .RP_ADDR [BCNT_LO];
4459              OVF_CHK (T_ADDR [TOT_WRITES_LO]);                       ! CHECK FOR FIELD OVERFLOW
4460              OVF_CHK (T_ADDR [BYTES_WRIT_LO]);                       !
4461              OVF_CHK (T_ADDR [TOT_BYT_WRT_LO]);                      !
4462              end;
4463
4464      if (.RP_ADDR [ENDCOD] eql (OP_RD or OP_END)) or
4465         (.RP_ADDR [ENDCOD] eql (OP_WRT or OP_END))
4466      then
4467          begin
4468          MILLIONS = .T_ADDR [MBYTES_READ] + .T_ADDR [MBYTES_WRIT];       ! TOTAL BYTES TRANSFERRED
4469          THOUSANDS = .T_ADDR [BYTES_READ_HI] + .T_ADDR [BYTES_WRIT_HI];
4470
4471          if .THOUSANDS gequ 1000
4472          then
4473              begin
```

```
;        4474              MILLIONS = .MILLIONS + 1;                          ! COUNT THE LOWER OVERFLOW TOO!
;        4475              THOUSANDS = .THOUSANDS - 1000;
;        4476              end;
;        4477
;        4478  !
;        4479  ! THIS ADDED BECAUSE IT WILL TAKE FOREVER TO TRANSFER ON THE ORDER OF A MEGABYTE TO A FLOPPY
;        4480  ! BUT IT IS A MUCH MORE REASONABLE MEASURE FOR THE RD51 WINCHESTER.  THE QUESTION NOW REFERS TO
;        4481  ! THE TOTAL DATA TRANSFER TO THE CONTROLLER AND THIS IS PRETTY CLOSE SINCE THE FLOPPIES GET
;        4482  ! ABOUT 1/1000 THE DATA THE HARD DISK(S) GET.
;        4483  !
;        4484
;        4485          if .SWP_XFER eql 0                                     ! IF THERE IS A TRANSFER LIMIT
;        4486          then
;        4487              begin
;        4488
;        4489              if .THOUSANDS gtru 100
;        4490              then
;        4491                  begin
;        4492                  EOP_FLAG = TRUE;                               ! SET END-OF PASS FLAG
;        4493                  DCT_ADDR [IG_INT] = TRUE;                      ! IGNORE FURTHER INTERRUPTS
;        4494                  end;
;        4495
;        4496              end
;        4497          else
;        4498
;        4499              if .MILLIONS gequ .SWP_XFER                        ! IF TRANSFER LIMIT IS REACHED
;        4500              then
;        4501                  begin
;        4502                  EOP_FLAG = TRUE;                               ! SET END-OF-PASS FLAG
;        4503                  DCT_ADDR [IG_INT] = TRUE;                      ! IGNORE FURTHER INTERRUPTS
;        4504                  end;
;        4505
;        4506              end;                                               ! IF UNIT IS STILL ALIVE
;        4507
;        4508      end;                                                       ! ROUTINE UPD_IO_TALLY

                                .SBTTL  UPD.IO.TALLY MULTI-DRIVE TEST ROUTINES
000000  004137  000000G         UPD.IO.TALLY:
                                JSR     R1,$SAVE2                          ;    4418
000004  013701  000000G         MOV     RP.ADDR,R1                         ;    4441
000010  126127  000014  000241  CMPB    14(R1),#241
000016  001027                  BNE     1$
000020  013700  000000G         MOV     T.ADDR,R0                          ;    4444
000024  005260  000016          INC     16(R0)
000030  066110  000020          ADD     20(R1),(R0)                        ;    4445
000034  066160  000020  000032  ADD     20(R1),32(R0)                      ;    4446
000042  012746  000016          MOV     #16,-(SP)                          ;    4447
000046  060016                  ADD     R0,(SP)
000050  004737  000000V         JSR     PC,OVF.CHK
000054  013716  000000G         MOV     T.ADDR,(SP)                        ;    4448
000060  004737  000000V         JSR     PC,OVF.CHK
000064  013716  000000G         MOV     T.ADDR,(SP)                        ;    4449
000070  062716  000032          ADD     #32,(SP)
000074  000435                  BR      2$
000076  126127  000014  000242  1$:     CMPB    14(R1),#242                ;    4453
```

```
000104  001034                                  BNE     3$
000106  013700  000000G                         MOV     T.ADDR,R0                       ;                               4456
000112  005260  000024                          INC     24(R0)
000116  066160  000020  000006                  ADD     20(R1),6(R0)                    ;                               4457
000124  066160  000020  000040                  ADD     20(R1),40(R0)                   ;                               4458
000132  012746  000024                          MOV     #24, (SP)                       ;                               4459
000136  060016                                  ADD     R0,(SP)
000140  004737  000000V                         JSR     PC,OVF.CHK
000144  013716  000000G                         MOV     T.ADDR,(SP)                     ;                               4460
000150  062716  000006                          ADD     #6,(SP)
000154  004737  000000V                         JSR     PC,OVF.CHK
000160  013716  000000G                         MOV     T.ADDR,(SP)                     ;                               4461
000164  062716  000040                          ADD     #40,(SP)
000170  004737  000000V                 2$:     JSR     PC,OVF.CHK
000174  005726                                  TST     (SP)+                           ;                               4455
000176  013700  000000G                 3$:     MOV     RP.ADDR,R0                      ;                               4464
000202  126027  000014  000241                  CMPB    14(R0),#241
000210  001404                                  BEQ     4$
000212  126027  000014  000242                  CMPB    14(R0),#242                     ;                               4465
000220  001037                                  BNE     8$
000222  013700  000000G                 4$:     MOV     T.ADDR,R0                       ;                               4468
000226  016002  000004                          MOV     4(R0),R2                        ; *,MILLIONS
000232  066002  000012                          ADD     12(R0),R2                       ; *,MILLIONS
000236  016001  000002                          MOV     2(R0),R1                        ; *,THOUSANDS                   4469
000242  066001  000010                          ADD     10(R0),R1                       ; *,THOUSANDS
000246  020127  001750                          CMP     R1,#1750                        ; THOUSANDS,*                   4471
000252  103403                                  BLO     5$
000254  005202                                  INC     R2                              ; MILLIONS                      4474
000256  162701  001750                          SUB     #1750,R1                        ; *,THOUSANDS                   4475
000262  013700  000000G                 5$:     MOV     SWP.XFER,R0                     ;                               4485
000266  001004                                  BNE     6$
000270  020127  000144                          CMP     R1,#144                         ; THOUSANDS,*                   4489
000274  101411                                  BLOS    8$
000276  000402                                  BR      7$                              ;                               4492
000300  020200                         6$:     CMP     R2,R0                           ; MILLIONS,*                    4499
000302  103406                                  BLO     8$
000304  112737  000001  000000G         7$:     MOVB    #1,EOP.FLAG                     ;                               4502
000312  052777  040000  000000G                 BIS     #40000,@DCT.ADDR                ;                               4503
000320  000207                         8$:     RTS     PC                              ;                               4418

; Routine Size:  105 words,     Routine Base:  $CODE$ + 17554
; Maximum stack depth per invocation:  5 words
```

```
;        4509  routine OVF_CHK (ADDR) : novalue =
;        4510
;        4511  !+
;        4512  !       THIS ROUTINE IS CALLED FROM UPD_IO_TALLY TO CHECK FOR OVERFLOW IN
;        4513  !       CERTAIN STATISTICAL FIELDS OF THE CURRENT UNIT. SPECIFICALLY, THE
;        4514  !       LOW-ORDER FIELD OF THE NUMBER OF BYTES READ OR WRITTEN IS CHECKED FOR
;        4515  !       EXCEEDING 1000. IF TRUE, THEN THE HIGH-ORDER COUNT IS INCREMENTED. IF
;        4516  !       THAT EXCEEDS 1000, THEN THE MEGABYTE COUNT IN INCREMENTED.
;        4517  !
;        4518  !       INPUTS:
;        4519  !               ADDR - ADDRESS OF THE BYTES_READ_LO OR BYTES_WRIT_LO FIELD FOR
;        4520  !                       THE CURRENT UNIT (SEE STATISTIC TABLE (TALLY) LAYOUT)
;        4521  !
;        4522
;        4523      begin
;        4524
;        4525      while ..ADDR gequ 1000 do                   ! IF LO-ORDER OVERFLOW
;        4526          begin
;        4527          .ADDR = ..ADDR   1000;                  ! SUBTRACT 1000
;        4528          (.ADDR + 2) = .(.ADDR + 2) + 1;         ! INCR HI-ORDER
;        4529          end;
;        4530
;        4531      if .(.ADDR + 2) gequ 1000                   ! IF HI-ORDER OVERFLOW
;        4532      then
;        4533          begin
;        4534          (.ADDR + 2) = .(.ADDR + 2) - 1000;      ! SUBTRACT 1000
;        4535          (.ADDR + 4) = .( ADDR + 4) + 1;         ! INCREMENT MBYTES
;        4536          end;
;        4537
;        4538      end;                                        ! ROUTINE OVF_CHK

                                .SBTTL  OVF.CHK MULTI-DRIVE TEST ROUTINES
000000  010146                OVF.CHK:MOV     R1,-(SP)                        ;                               4509
000002  016600  000004                MOV     4(SP),R0                        ; ADDR,*                        4525
000006  012701  000002                MOV     #2,R1                           ;                               4528
000012  060001                        ADD     R0,R1
000014  021027  001750        1$:     CMP     (R0),#1750                      ;                               4525
000020  103404                        BLO     2$
000022  162710  001750                SUB     #1750,(R0)                      ;                               4527
000026  005211                        INC     (R1)                            ;                               4528
000030  000771                        BR      1$                              ;                               4525
000032  021127  001750        2$:     CMP     (R1),#1750                      ;                               4531
000036  103404                        BLO     3$
000040  162711  001750                SUB     #1750,(R1)                      ;                               4534
000044  005260  000004                INC     4(R0)                           ;                               4535
000050  012601                3$:     MOV     (SP)+,R1                        ;                               4509
000052  000207                        RTS     PC

; Routine Size:  22 words,      Routine Base:  $CODE$ + 20076
; Maximum stack depth per invocation:  2 words
```

```
;          4539  routine HOST_WRT_CHK =
;          4540
;          4541  !+
;          4542  !       THIS ROUTINE IS CALLED FROM IO_RETPKT FOR ALL I/O TRANSFER RETURN
;          4543  !       PACKETS WITH "SUCCESS" STATUS CODES, BUT ONLY IF THE HOST WRITE-COMPARE
;          4544  !       OPTION WAS SELECTED BY THE OPERATOR.
;          4545  !
;          4546  !       IF THE CURRENT RETPKT BEING PROCESSED IS A WRITE FUNCTION, THEN THE
;          4547  !       PACKET INDEX (RP_INDX) IS SAVED IN THE CONTROLLER'S RETURN PACKET SAVE
;          4548  !       AREA (RP_SAVE). OTHERWISE, THE PACKET IS A READ, SO ITS ASSOCIATED
;          4549  !       WRITE PACKET IS REMOVED FROM THE SAVE AREA, AND A BYTE-BY BYTE
;          4550  !       COMPARISON IS PERFORMED ON THE TWO I/O BUFFERS. ANY DIFFERENCES
;          4551  !       ENCOUNTERED RESULTS IN THE DECLARATION OF A HARD ERROR.
;          4552  !
;          4553  !       IMPLICIT INPUTS:
;          4554  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;          4555  !               RP_INDX - INDEX OF THE CURRENT RETURN PACKET
;          4556  !-
;          4557
;          4558      begin
;          4559
;          4560      local
;          4561          BUFF1 : ref block [MAX_XFER * 2, byte], ! I/O BUFFER ADDRESS
;          4562          BUFF2 : ref block [MAX_XFER * 2, byte], ! I/O BUFFER ADDRESS
;          4563          BUFFW,                                  ! I/O BUFFER ADDRESS
;          4564          COUNT : word,                           ! BYTE COUNT
;          4565          FLAG : byte initial (byte (TRUE)),
;          4566          index : signed word;
;          4567
;          4568      if .RP_ADDR [ENDCOD] eql (OP_WRT or OP_END) ! IF WRITE OPERATION
;          4569      then
;          4570          FLAG = FALSE                            ! DON'T CALL SWEEP FROM IO_RETPKT
;          4571      else
;          4572          if (.RP_ADDR [ENDCOD] eql (OP_RD or OP_END)) and
;          4573             ((index = RPS_REM ()) geq 0)         ! IF ASSOCIATED WRITE PACKET IS FOUND ELSE ENDCODE IS READ
;          4574          then
;          4575              begin
;          4576              BUFFW = RETPKT [.index, BUFF_0];    ! ADDR OF ADDR OF WRITE I/O BUFFER
;          4577              BUFF1 = ..BUFFW;                    ! ADDR OF WRITE I/O BUFFER
;          4578              BUFF2 = .RP_ADDR [BUFF_0];          ! ADDR OF READ I/O BUFFER
;          4579              COUNT = .RP_ADDR [BCNT_LO];         ! BYTE COUNT
;          4580
;          4581              incr I from 1 to .COUNT do          ! FOR EACH BYTE IN BUFFERS
;          4582
;          4583                  if .(.BUFF1)<0, 8, 0> eql .(.BUFF2)<0, 8, 0>    ! IF BYTES COMPARE O.K.
;          4584                  then
;          4585                      begin
;          4586                      BUFF1 = .BUFF1 + 1;         ! ADVANCE WRITE BUFFER ADDR
;          4587                      BUFF2 = .BUFF2 + 1;         ! ADVANCE READ BUFFER ADDR
;          4588                      end
;          4589                  else
;          4590                      begin                       ! ELSE - COMPARE ERROR
;          4591                      T_ADDR [ERR_HARD] = .T_ADDR [ERR_HARD] + 1;
;          4592                      T_ADDR [ERR_HRD_HST] = .T_ADDR [ERR_HRD_HST] + 1;
;          4593                      ERRHRD (42, EGH_30, 0);     ! I/O REQUEST FAILED
;          4594                      EMS_CMP (RETPKT + (.index * RP_LEN * 2));
```

```
;          4595
;          4596                    ;if .T_ADDR [ERR HARD] gequ .SWP ERROR
;          4597                    then
;          4598                        begin
;          4599                        DUR [.L$LUN] = DU HERR; ! IF ERROR COUNT EXCEEDED
;          4600                        DODU (.L$LUN);          ! DROP UNIT
;          4601                        end;
;          4602
;          4603                    exitloop;                   ! NO NEED TO CONTINUE
;          4604                    end;                        ! IF COMPARE ERROR
;          4605
;          4606                end;                            ! IF ASSOCIATED WRITE RETPKT WAS FOUND
;          4607
;          4608        return (.FLAG);
;          4609        end;                                    ! ROUTINE HOST_WRT_CHK


                                        .SBTTL  HOST.WRT.CHK MULTI-DRIVE TEST ROUTINES
000000  004137  000000G           HOST.WRT.CHK:
                                        JSR     R1,$SAVES                       ;                       4539
000004  005746                          TST     -(SP)
000006  112705  000001                  MOVB    #1,R5                           ; *.FLAG                4558
000012  013700  000000G                 MOV     RP.ADDR,R0                      ;                       4568
000016  126027  000014  000242          CMPB    14(R0),#242
000024  001002                          BNE     1$
000026  105005                          CLRB    R5                              ; FLAG                  4570
000030  000500                          BR      6$                              ;                       4568
000032  126027  000014  000241  1$:     CMPB    14(R0),#241                     ;                       4572
000040  001074                          BNE     6$
000042  004737  000000V                 JSR     PC,RPS.REM                      ;                       4573
000046  005700                          TST     R0                              ; INDEX
000050  002470                          BLT     6$
000052  010046                          MOV     R0,-(SP)                        ; INDEX,*               4576
000054  012746  000060                  MOV     #60,-(SP)
000060  004737  000000G                 JSR     PC,BL$MUL
000064  010066  000004                  MOV     R0,4(SP)
000070  062700  000024G                 ADD     #RETPKT+24,R0                   ; *.BUFFW
000074  011002                          MOV     (R0),R2                         ; BUFFW,BUFF1           4577
000076  013700  000000G                 MOV     RP.ADDR,R0                      ;                       4578
000102  016003  000024                  MOV     24(R0),R3                       ; *.BUFF2
000106  016004  000020                  MOV     20(R0),R4                       ; *.COUNT               4579
000112  005001                          CLR     R1                              ; I                     4581
000114  000442                          BR      4$
000116  121213          2$:     CMPB    (R2),(R3)                       ; BUFF1,BUFF2           4583
000120  001003                          BNE     3$
000122  005202                          INC     R2                              ; BUFF1                 4586
000124  005203                          INC     R3                              ; BUFF2                 4587
000126  000435                          BR      4$                              ;                       4583
000130  013700  000000G         3$:     MOV     T.ADDR,R0                       ;                       4591
000134  005260  000014                  INC     14(R0)
000140  105260  000063                  INCB    63(R0)                          ;                       4592
000144  104456                          TRAP    56                              ;                       4593
000146  000052                          .WORD   52
000150  000000G                         .WORD   EGH.30
000152  000000                          .WORD   0
000154  016616  000004                  MOV     4(SP),(SP)                      ;                       4594
```

```
000160  062716  000000G                        ADD     #RETPKT,(SP)
000164  004737  000000G                        JSR     PC,EMS.CMP
000170  013700  000000G                        MOV     T.ADDR,R0                       ;                       4596
000174  026037  000014  000000G                CMP     14(R0),SWP.ERROR
000202  103412                                 BLO     5$
000204  013700  000000G                        MOV     L$LUN,R0                        ;                       4599
000210  112760  000004  000000G                MOVB    #4,DUR(R0)
000216  104451                                 TRAP    51                              ;                       4600
000220  000403                                 BR      5$                              ;                       4590
000222  005201                          4$:    INC     R1                              ; I                     4581
000224  020104                                 CMP     R1,R4                           ; I.COUNT
000226  003733                                 BLE     2$
000230  022626                          5$:    CMP     (SP)+,(SP)+                     ;                       4575
000232  005000                          6$:    CLR     R0                              ;                       4558
000234  150500                                 BISB    R5,R0                           ; FLAG,*
000236  005726                                 TST     (SP)+                           ;                       4539
000240  000207                                 RTS     PC

; Routine Size:  81 words,      Routine Base:  $CODE$ + 20152
; Maximum stack depth per invocation:  11 words
```

```
;          4610  routine SWEEP : novalue =
;          4611
;          4612  !+
;          4613  !       THIS ROUTINE IS CALLED FROM IO_RETPKT AND OTHERS TO DEALLOCATE THE
;          4614  !       RESOURCES ASSOCIATED WITH THE CURRENT RETURN PACKET. THIS INCLUDES THE
;          4615  !       PACKET ITSELF AND THE I/O BUFFER. IN ADDITION, IF THE HOST IS
;          4616  !       PERFORMING WRITE-COMPARES, AND IF THE CURRENT RETURN PACKET IS A READ
;          4617  !       FUNCTION, THEN THE CURRENT CONTROLLER'S RP_SAVE AREA IS SEARCHED FOR
;          4618  !       THE ASSOCIATED WRITE RETPKT SO THAT ITS RESOURCES CAN ALSO BE
;          4619  !       DEALLOCATED.
;          4620  !
;          4621  !       IMPLICIT INPUTS:
;          4622  !               RP_ADDR - ADDRESS OF CURRENT RETURN PACKET
;          4623  !               RP_INDX - INDEX OF CURRENT RETURN PACKET
;          4624  !-
;          4625
;          4626      begin
;          4627
;          4628      local
;          4629          index : signed word;
;          4630
;          4631      if (.RP_ADDR [ENDCOD] and OP_MSK) eql OP_RD ! IF READ OPCODE OR ENDCODE
;          4632      then
;          4633
;          4634          if BIT_TST (SWP_FLAGS, SWF_HWC)         ! IF HOST IS DOING WRITE-COMPARES
;          4635          then
;          4636
;          4637              if (index = RPS_REM ()) geq 0       ! IF ASSOCIATED WRITE RETPKT IS FOUND
;          4638              then
;          4639                  begin
;          4640                  PUT_IO_BUFF (RETPKT [.index, BUFF_O]);  ! RETURN WRITE I/O BUFFER TO POOL
;          4641                  PUT_RETPKT (.index);            ! RETURN WRITE PACKET TO POOL
;          4642                  end;
;          4643
;          4644      PUT_IO_BUFF (RP_ADDR [BUFF_O]);             ! RETURN CURRENT I/O BUFFER TO POOL
;          4645      PUT_RETPKT (.RP_INDX);                      ! RETURN CURRENT RETPKT TO POOL
;          4646      end;                                        ! ROUTINE SWEEP


                                        .SBTTL  SWEEP MULTI-DRIVE TEST ROUTINES
000000  010146                  SWEEP:  MOV     R1,-(SP)                        ;                       4610
000002  013700  000000G                 MOV     RP.ADDR,R0                      ;                       4631
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                   ;                       4634
000032  001422                         BEQ     1$
000034  004737  000000V                 JSR     PC,RPS.REM                      ;                       4637
000040  010001                         MOV     R0,R1                           ; *,INDEX
000042  002416                         BLT     1$
000044  010146                         MOV     R1,-(SP)                        ; INDEX,*               4640
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,*                       4641
000072  004737  000000G                        JSR     PC,PUT.RETPKT
000076  022626                                  CMP     (SP)+,(SP)+             ;                               4639
000100  013746  000000G                1$:     MOV     RP.ADDR,-(SP)           ;                               4644
000104  062716  000024                        ADD     #24,(SP)
000110  004737  000000G                        JSR     PC,PUT.IO.BUFF
000114  013716  000000G                        MOV     RP.INDX,(SP)            ;                               4645
000120  004737  000000G                        JSR     PC,PUT.RETPKT
000124  005726                                  TST     (SP)+                   ;                               4626
000126  012601                                  MOV     (SP)+,R1                ;                               4610
000130  000207                                  RTS     PC

; Routine Size:  45 words,      Routine Base:  $CODE$ + 20414
; Maximum stack depth per invocation:  4 words
```

```
;          4647  routine RPS_REM =
;          4648
;          4649  !+
;          4650  !       THIS ROUTINE SEARCHES THE CURRENT CONTROLLER'S RP_SAVE AREA FOR A
;          4651  !       RETURN PACKET WHOSE COMMAND REFERENCE NUMBER (CRN) IS ONE LESS THAN THE
;          4652  !       CRN OF THE CURRENT RETURN PACKET (I.E., SEARCHING FOR THE SAVED WRITE
;          4653  !       OPERATION ASSOCIATED WITH THE CURRENT READ OPERATION). IF FOUND, THE
;          4654  !       RP_SAVE ENTRY IS CLEARED (TO -1) AND THE RETPKT INDEX OF THE WRITE
;          4655  !       OPERATION IS RETURNED TO THE CALLER.
;          4656  !
;          4657  !       IMPLICIT INPUTS:
;          4658  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;          4659  !
;          4660  !       OUTPUTS:
;          4661  !               INDEX (VALUE OF THIS ROUTINE) - INDEX OF THE RETPKT CONTAINING
;          4662  !                       A CRN WHICH IS ONE LESS THAN THE CURRENT
;          4663  !-
;          4664
;          4665      begin
;          4666
;          4667      local
;          4668          index : signed word initial (-1);                          ! ASSUME NOT FOUND
;          4669
;          4670      incr COUNT from 0 to RP_CNT - 1 do                             ! FOR EACH ENTRY IN RP_SAVE
;          4671
;          4672          if (.RP_USE [.COUNT] eql .CCTLR) and                           ! IF THIS IS A VALID RETPKT
INDEX
;          4673             (.RETPKT [.COUNT, ENDCOD] eql (OP_WRT or OP_END))
;          4674          then
;          4675
;          4676              if ((.RETPKT [.COUNT, CRF_LO] eql (.RP_ADDR [CRF_LO] - 1)) and     ! IF CORRECT CRN
;          4677                  (.RETPKT [.COUNT, CRF_HI] eql .RP_ADDR [CRF_HI])) or
;          4678                 ((.RETPKT [.COUNT, CRF_HI] eql (.RP_ADDR [CRF_HI] - 1)) and
;          4679                  (.RETPKT [.COUNT, CRF_LO] eql %o'177777') and
;          4680                  (.RP_ADDR [CRF_LO] eql 0))
;          4681              then
;          4682                  begin
;          4683                  index = .COUNT;                                    ! INDEX TO BE RETURNED
;          4684                  exitloop;                                          ! DONE
;          4685                  end;
;          4686
;          4687      return .index;
;          4688      end;                                                           ! ROUTINE RPS_REM


                                        .SBTTL  RPS.REM MULTI-DRIVE TEST ROUTINES
000000  004137  000000G           RPS.REM:JSR     R1,$SAVE4                       ;                                       4647
000004  012704  177777                    MOV     #-1,R4                          ; *,INDEX                               4665
000010  005003                           CLR     R3                              ; COUNT                                 4670
000012  116300  000000G           1$:     MOVB    RP.USE(R3),R0                   ; *(COUNT),*                            4672
000016  020037  000000G                   CMP     R0,CCTLR
000022  001053                           BNE     4$
000024  010346                           MOV     R3,-(SP)                        ; COUNT,*                               4673
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,*                       4676
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)      ;                               4677
000112  001415                          BEQ     3$
000114  016102  000006          2$:     MOV     6(R1),R2                ;                               4678
000120  005302                          DEC     R2
000122  026002  000006G                 CMP     RETPKT+6(R0),R2
000126  001011                          BNE     4$
000130  026027  000004G 177777          CMP     RETPKT+4(R0),# 1        ;                               4679
000136  001005                          BNE     4$
000140  005761  000004                  TST     4(R1)                   ;                               4680
000144  001002                          BNE     4$
000146  010304                  3$:     MOV     R3,R4                   ; COUNT.INDEX                   4683
000150  000404                          BR      5$                      ;                               4682
000152  005203                  4$:     INC     R3                      ; COUNT                         4670
000154  020327  000003                  CMP     R3,#3                   ; COUNT,*
000160  003714                          BLE     1$
000162  010400                  5$:     MOV     R4,R0                   ; INDEX,*                       4665
000164  000207                          RTS     PC                      ;                               4647

; Routine Size:  59 words,      Routine Base:  $CODE$ + 20546
; Maximum stack depth per invocation:  8 words
```

```
;          4689  routine DR_RETPKT : novalue =
;          4690
;          4691  !+
;          4692  !       THIS ROUTINE IS CALLED BY PROC_RETPKT FOR ALL PACKETS ORIGINATING AT
;          4693  !       THE "DRIVER" PORTION OF THE PROGRAM. THIS INCLUDES PACKETS DESCRIBING
;          4694  !       FATAL DEVICE ERRORS.
;          4695  !
;          4696  !       FOR FATAL DEVICE ERRORS, THIS ROUTINE RELEASES ALL RESOURCES HELD BY
;          4697  !       THE CONTROLLER. THE CONTROLLER IS MARKED OFFLINE IN ITS CST, AND ALL
;          4698  !       UNITS ATTACHED TO THE CONTROLLER ARE DROPPED.
;          4699  !
;          4700  !       IMPLICIT INPUTS:
;          4701  !               RP_INDX - INDEX OF THE CURRENT RETURN PACKET
;          4702  !               RP_ADDR - ADDRESS OF THE CURRENT RETURN PACKET
;          4703  !               CST_ADDR   ADDRESS OF THE CURRENT CONTROLLER'S CST
;          4704  !               CCTLR - CURRENT CONTROLLER NUMBER
;          4705  !-
;          4706
;          4707      begin
;          4708
;          4709      PUTA_BUFF ();                           ! RELEASE ALL I/O BUFFERS HELD BY CONTROLLER
;          4710
;          4711      incr index from 0 to RP_CNT - 1 do      ! FOR EACH ENTRY IN CONTROLLER'S RP_SAVE
;          4712
;          4713          if .RP_USE [.index] eql .CCTLR      ! IF VALID RETPKT INDEX
;          4714          then
;          4715              PUT_RETPKT (.index);            ! RETURN RETPKT TO POOL
;          4716
;          4717      QIO [.CCTLR] = 0;                       ! CLEAR NO. OF OUTSTANDING QIOS
;          4718      CST_ADDR [STATE] = OFFLINE;             ! MARK CST OFFLINE
;          4719      DROP_CTLR (.CCTLR, DU_CFATAL);          ! DROP CONTROLLER'S UNITS
;          4720      PUT_RETPKT (.RP_INDX);                  ! PUT BACK RETPKT
;          4721      end;                                    ! ROUTINE DR_RETPKT
```

```
                                        ;***  DR.RETPKT MULTI DRIVE TEST ROUTINE
000000  010146                  DR.RETPKT:
                                        MOV     R1,-(SP)                ;                               4689
000002  004737  000000G                 JSR     PC,PUTA.BUFF            ;                               4709
000006  005001                          CLR     R1                      ; INDEX                         4711
000010  116100  000000G         10:     MOVB    RP.USE(R1),R0           ; R0=INDEX,0                    4713
000014  020037  000000G                 CMP     R0,CCTLR
000020  001004                          BNE     20
000022  010146                          MOV     R1,-(SP)                ; INDEX,0                       4715
000024  004737  000000G                 JSR     PC,PUT.RETPKT
000030  005726                          TST     (SP)+
000032  005201                  20:     INC     R1                      ; INDEX                         4711
000034  020127  000005                  CMP     R1,#5                   ; INDEX,0
000040  003765                          BLE     10
000042  013701  000000G                 MOV     CCTLR,R1                ;                               4717
000046  105061  000000G                 CLRB    QIO(R1)
000052  013700  000000G                 MOV     CST.ADDR,R0             ;                               4718
000056  042760  100000  000002         BIC     #100000,2(R0)
000064  010146                          MOV     R1,-(SP)                ;                               4719
000066  012746  000006                  MOV     #6,-(SP)
000072  004737  000000G                 JSR     PC,DROP.CTLR
000076  013716  000000G                 MOV     RP.INDX,(SP)            ;                               4720
000102  004737  000000G                 JSR     PC,PUT.RETPKT
000106  022626                          CMP     (SP)+,(SP)+             ;                               4707
000110  012601                          MOV     (SP)+,R1                ;                               4689
000112  000207                          RTS     PC

; Routine Size:  38 words,    Routine Base:  $CODE$ + 20734
; Maximum stack depth per invocation:  4 words
```

```
;         4722  %sbttl  RQRX INTERRUPT SERVICE ROUTINES
;         4723
;         4724  !+
;         4725  !       THERE EXISTS AN RQRX INTERRUPT SERVICE ROUTINE FOR EACH DEVICE
;         4726  !       CONTROLLER. EACH SERVICE ROUTINE BEGINS BY SIMPLY SETTING THE
;         4727  !       APPROPRIATE CONTROLLER NUMBER INTO "ICTLR". ALL SERVICE ROUTINES THEN
;         4728  !       BRANCH TO A COMMON INTERRUPT PROCESSING ROUTINE.
;         4729  !
;         4730
;         4731  BGNSRV (AZINTO);
;         4732  ICTLR = 0;
;         4733  AZINT ();
;         4734  ENDSRV;


                                        .SBTTL  AZINTO RQRX INTERRUPT SERVICE ROUTINES
000000  010046                  AZINTO::MOV     R0,-(SP)                ;                       4731
000002  005037  000156'                CLR     ICTLR                   ;                       4732
000006  004737  000000V                JSR     PC,AZINT                ;                       4733
000012  012600                        MOV     (SP)+,R0                ;                       4731
000014  000002                        RTI

; Routine Size:  7 words,       Routine Base:  $CODE$ + 21050
; Maximum stack depth per invocation:  2 words
```

```
;        4735  routine AZINT : novalue =
;        4736
;        4737  !+
;        4738  !       THIS IS THE COMMON INTERRUPT SERVICE ROUTINE FOR ALL RDRX CONTROLLERS.
;        4739  !       AFTER CALCULATING THE DCT ADDRESS FOR THE INTERRUPTING DEVICE, THIS
;        4740  !       ROUTINE WILL SAVE THE CURRENT CONTENTS OF THE SA REGISTER IN THE DCT.
;        4741  !       THEN, IF THE "IGNORE INTERRUPT" BIT IS SET, NO FURTHER ACTION IS TAKEN.
;        4742  !       OTHERWISE, THE SA VALUE IS CHECKED FOR A FATAL ERROR, AND THE COMMAND
;        4743  !       AND RESPONSE RINGS ARE POLLED.
;        4744  !-
;        4745
;        4746      begin
;        4747      IDCT_ADDR = DCT + (.ICTLR * DCT_LEN * 2);    ! GET DCT ADDRESS
;        4748      ICST_ADDR = CST + (.ICTLR * CST_LEN * 2);    ! GET CST ADDRESS
;        4749      IRDRX_ADDR = .ICST_ADDR [IP_ADDR];           ! GET RDRX ADDRESS
;        4750      ICOM_ADDR = COMM_AREA + (.ICTLR * COMM_LEN * 2);     ! GET COMM_AREA ADDR
;        4751      IDCT_ADDR [SA_SAVE] = .IRDRX_ADDR [RCSA, RC_ALL];    ! SAVE SA REGISTER
;        4752
;        4753      if .IDCT_ADDR [IG_INT]                       ! IGNORE INTERRUPT?
;        4754      then
;        4755          return;                                  ! RETURN IF INTERRUPTS IGNORED
;        4756
;        4757      if BIT_TST (IDCT_ADDR [SA_SAVE], SA_ERR)     ! IF FATAL ERROR
;        4758      then
;        4759          FATAL_ERROR ()
;        4760      else
;        4761          begin
;        4762          POLL_CRING ();                           ! POLL COMMAND RING
;        4763          POLL_RRING ();                           ! POLL RESPONSE RING
;        4764          end;
;        4765
;        4766      end;

                                .SBTTL  AZINT RDRX INTERRUPT SERVICE ROUTINES
000000  010146                AZINT:  MOV     R1,-(SP)                        ;                       4735
000002  005746                        TST     -(SP)
000004  013701  000156'                MOV     ICTLR,R1                        ;                       4747
000010  010146                        MOV     R1,-(SP)
000012  012746  000022                MOV     #22,-(SP)
000016  004737  000000G               JSR     PC,BL$MUL
000022  062700  000000G               ADD     #DCT,R0
000026  010037  000110'               MOV     R0,IDCT.ADDR
000032  010116                        MOV     R1,(SP)                         ;                       4748
000034  012746  000056                MOV     #56,-(SP)
000040  004737  000000G               JSR     PC,BL$MUL
000044  062700  000000G               ADD     #CST,R0
000050  010037  000106'               MOV     R0,ICST.ADDR
000054  011037  000000G               MOV     (R0),IRDRX.ADDR                 ; ICST.ADDR,*           4749
000060  010116                        MOV     R1,(SP)                         ;                       4750
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                    ;                       4751
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                   4753
000132  001016                                  BNE     2$                              ;                               4735
000134  016600  000010                          MOV     10(SP),R0                       ;                               4757
000140  042700  077777                          BIC     #77777,R0
000144  020027  100000                          CMP     R0,# 100000
000150  001003                                  BNE     1$
000152  004737  000000V                         JSR     PC,FATAL.ERROR                  ;                               4759
000156  000404                                  BR      2$                              ;                               4757
000160  004737  000000V             1$:         JSR     PC,POLL.CRING                   ;                               4762
000164  004737  000000V                         JSR     PC,POLL.RRING                   ;                               4763
000170  062706  000012             2$:         ADD     #12,SP                          ;                               4735
000174  012601                                  MOV     (SP)+,R1
000176  000207                                  RTS     PC

; Routine Size:  64 words,      Routine Base:  $CODE$ + 21066
; Maximum stack depth per invocation:  7 words
```

```
;         4767  routine FATAL_ERROR : novalue =
;         4768
;         4769  !+
;         4770  !       THIS ROUTINE IS CALLED BY THE INTERRUPT SERVICE ROUTINE (AZINT) UPON
;         4771  !       DETECTING AN UNRECOVERABLE ERROR THROUGH THE DEVICE'S SA REGISTER.
;         4772  !       ITS PURPOSE IS TO CLEAN UP DEVICE DATA IN THE "DRIVER" PORTION OF
;         4773  !       THE EXERCISER, AND TO INFORM THE "PROGRAM" PORTION OF THE EVENT VIA
;         4774  !       RETURN PACKET.
;         4775  !
;         4776  !       IMPLICIT INPUTS:
;         4777  !               ICTLR   INTERRUPTING CONTROLLER NUMBER
;         4778  !               IDCT_ADDR   ADDRESS OF INTERRUPTING CONTROLLER'S DCT
;         4779  !               ICST_ADDR - ADDRESS OF INTERRUPTING CONTROLLER'S CST
;         4780  !               IRDRX_ADDR   ADDRESS OF INTERRUPTING CONTROLLER'S IP REGISTER
;         4781  !
;         4782
;         4783      begin
;         4784
;         4785      local
;         4786          index : signed word,
;         4787          U_SAVE : word;
;         4788
;         4789      SA_REG = .IDCT_ADDR [SA_SAVE];
;         4790      U_SAVE = .L$LUN;                                ! SAVE PRE-INTERRUPT CURRENT UNIT NUMBER
;         4791      C_ERR_TBL [.ICTLR, C_ERR_HRD] = .C_ERR_TBL [.ICTLR, C_ERR_HRD] + 1;
;         4792      L$LUN = .ICST_ADDR [OF_UN, D_UNIT];             ! SET CURRENT UNIT TO FIRST IN CONTROLLER
;         4793      ERRDF (14, EGD_14, EMS_14);                     ! FATAL CONTROLLER ERROR
;         4794      L$LUN = .U_SAVE;                                ! RESTORE PRE-INTERRUPT CURRENT UNIT
;         4795      DRV_CTLERR (.ICTLR);                            ! CLEAN UP DRIVER DATA FOR CONTROLLER
;         4796
;         4797      if (index = GET_RETPKT (.ICTLR)) lss 0          ! TRY TO GET A RETPKT; IF FAILURE
;         4798      then
;         4799  !       PRINTF (DBM18)                              ! "FATAL_ERROR: RETPKT NOT AVAILABLE"   !JSD REV A
;         4800      else
;         4801          begin                                       ! IF RETPKT WAS ALLOCATED
;         4802          RETPKT [.index, CONID] = CID_DRIVER;        ! SET CONNECTION ID TO "DRIVER"
;         4803          RETPKT [.index, MESTYP] = MT_FATAL;         ! FATAL ERROR
;         4804          RETPKT [.index, CTLR] = .ICTLR;             ! CONTROLLER NUMBER
;         4805          IN_IODQ (.index);                           ! LOAD RETPKT INDEX INTO IODQ
;         4806          end;                                        ! IF RETPKT WAS ALLOCATED
;         4807
;         4808      end;                                            ! ROUTINE FATAL_ERR


                                        .SBTTL  FATAL.ERROR RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G           FATAL.ERROR:
                                        JSR     R1,$SAVE2                       ;                                       4767
000004  013700  000110'                 MOV     IDCT.ADDR,R0                    ;                                       4789
000010  016037  000002  000000G         MOV     2(R0),SA.REG
000016  013701  000000G                 MOV     L$LUN,R1                        ; =.U.SAVE                              4790
000022  013700  000156'                 MOV     ICTLR,R0                        ;                                       4791
000026  006300                          ASL     R0
000030  105260  000000G                 INCB    C.ERR.TBL(R0)
000034  013700  000106'                 MOV     ICST.ADDR,R0                    ;                                       4792
000040  016002  000006                  MOV     6(R0),R2
000044  000302                          SWAB    R2
```

```
000046  042702  177760                          BIC     #177760,R2
000052  010237  000000G                         MOV     R2,L$LUN
000056  104455                                  TRAP    55                      ;                       4793
000060  000016                                  .WORD   16
000062  000000G                                 .WORD   EGD.14
000064  000000G                                 .WORD   EMS.14
000066  010137  000000G                         MOV     R1,L$LUN                ; U.SAVE,*              4794
000072  013746  000156'                         MOV     ICTLR,-(SP)             ;                       4795
000076  004737  000000G                         JSR     PC,DRV.CTLERR
000102  013716  000156                          MOV     ICTLR,(SP)              ;                       4797
000106  004737  000000G                         JSR     PC,GET.RETPKT
000112  010001                                  MOV     R0,R1                   ; *,INDEX
000114  002425                                  BLT     1$
000116  010116                                  MOV     R1,(SP)                 ; INDEX,*               4802
000120  012746  000060                          MOV     #60,-(SP)
000124  004737  000000G                         JSR     PC,BL$MUL
000130  062700  000002G                         ADD     #RETPKT+2,R0
000134  112760  000003  000001                  MOVB    #3,1(R0)
000142  013702  000156'                         MOV     ICTLR,R2                ;                       4804
000146  042702  177760                          BIC     #177760,R2
000152  112710  000060                          MOVB    #60,(R0)
000156  150210                                  BISB    R2,(R0)
000160  010116                                  MOV     R1,(SP)                 ; INDEX,*               4805
000162  004737  000000G                         JSR     PC,IN.IODQ
000166  005726                                  TST     (SP)+                   ;                       4801
000170  005726                          1$:     TST     (SP)+                   ;                       4783
000172  000207                                  RTS     PC                      ;                       4767

; Routine Size:  62 words,      Routine Base:  $CODE$ + 21266
; Maximum stack depth per invocation:  6 words
```

```
;          4809  routine POLL_CRING : novalue =
;          4810
;          4811  !+
;          4812  !       THIS ROUTINE IS CALLED BY THE RDRX INTERRUPT SERVICE ROUTINE (AZINT)
;          4813  !       FOR EACH DEVICE INTERRUPT EXCEPT DURING INITIALIZATION OR FATAL ERROR.
;          4814  !       ITS PURPOSE IS TO SCAN THE DEVICE'S COMMAND RING AND CHECK FOR ANY
;          4815  !       COMMAND SLOTS THAT HAVE BEEN "TAKEN" BY THE CONTROLLER. SUCH SLOTS
;          4816  !       HAVE BEEN RETURNED TO THE HOST, INDICATED BY A ZERO OWNERSHIP BIT. FOR
;          4817  !       EACH SLOT THAT HAS BEEN RETURNED TO THE HOST, THE CRING COUNT IS
;          4818  !       DECREMENTED, AND THE CR_POLL ADDRESS IS ADVANCED TO THE NEXT SLOT IN
;          4819  !       THE COMMAND RING.
;          4820  !
;          4821  !       IMPLICIT INPUTS:
;          4822  !               ICTLR - INTERRUPTING CONTROLLER NUMBER
;          4823  !               IDCT_ADDR - ADDRESS OF INTERRUPTING CONTROLLER'S DCT
;          4824  !               ICOM_ADDR - ADDRESS OF INTERRUPTING CONTROLLER'S COMM AREA
;          4825  !
;          4826
;          4827      begin
;          4828
;          4829      while ((.IDCT_ADDR [CRING_CNT] gtru 0) and                    ! WHILE # OF COMMANDS IN CRING > 0 AND
;          4830          not (BIT_TST ((.IDCT_ADDR [CR_POLL] + 2), ED_OWN))) do    ! CURRENT SLOT IS HOST-OWNED
;          4831          begin
;          4832          IDCT_ADDR [CRING_CNT] = .IDCT_ADDR [CRING_CNT]   1;       ! DECREMENT # CMDS IN CRING
;          4833          IDCT_ADDR [CR_POLL] = .IDCT_ADDR [CR_POLL] + 4;           ! ADVANCE TO NEXT SLOT TO POLL
;          4834
;          4835          if .IDCT_ADDR [CR_POLL] gtra .IDCT ADDR [CR_END]          ! IF BEYOND END OF RING
;          4836          then
;          4837              IDCT_ADDR [CR_POLL] = .IDCT_ADDR [CR_BEG];            ! SET POINTER TO TOP OF CRING
;          4838
;          4839          end;
;          4840
;          4841      ICOM_ADDR [CMD_INT] = 0;                                      ! CLEAR COMMAND INTERRUPT WORD IN RING HEADER
;          4842      end;
```

```
                                        .SBTTL  POLL.CRING RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G        POLL.CRING:
                                        JSR     R1,$SAVE2                       ;                       4809
000004  013701  000110                  MOV     IDCT.ADDR,R1                    ;                       4829
000010  012702  000016                  MOV     #16,R2                          ;                       4833
000014  060102                          ADD     R1,R2
000016  105711                  1$:     TSTB    (R1)                            ;                       4829
000020  001422                          BEQ     2$
000022  016100  000016                  MOV     16(R1),R0                       ;                       4830
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)                            ;                       4832
000046  062712  000004                  ADD     #4,(R2)                         ;                       4833
000052  021261  000012                  CMP     (R2),12(R1)                     ;                       4835
000056  101757                          BLOS    1$
000060  016112  000010                  MOV     10(R1),(R2)                     ;                       4837
000064  000754                          BR      1$                              ;                       4829
000066  013700  000104'         2$:     MOV     ICOM.ADDR,R0                    ;                       4841
000072  005060  000004                  CLR     4(R0)
000076  000207                          RTS     PC                              ;                       4809

; Routine Size:  32 words,      Routine Base:  $CODE$ + 21462
; Maximum stack depth per invocation:  4 words
```

```
;        4843  routine POLL_RRING : novalue =
;        4844
;        4845  !+
;        4846  !       THIS ROUTINE IS CALLED BY THE RDRX INTERRUPT SERVICE ROUTINE (AZINT)
;        4847  !       FOR EACH DEVICE INTERRUPT EXCEPT DURING INITIALIZATION OR FATAL ERROR.
;        4848  !       ITS PURPOSE IS TO SCAN THE DEVICE'S RESPONSE RING AND CHECK FOR ANY
;        4849  !       SLOTS WHICH HAVE BEEN RETURNED TO THE HOST (OWNERSHIP BIT = 0). FOR
;        4850  !       EACH SUCH SLOT, THE ASSOCIATED MESSAGE IS PROCESSED BASED ON ITS
;        4851  !       CONNECTION ID (MSCP OR DUP). AFTER PROCESSING, THE MESSAGE PACKET
;        4852  !       IS RE-INITIALIZED AND RETURNED TO THE CONTROLLER (OWNERSHIP BIT SET
;        4853  !       TO 1).
;        4854  !
;        4855  !       IMPLICIT INPUTS:
;        4856  !               ICTLR - NUMBER OF INTERRUPTING CONTROLLER
;        4857  !               IDCT_ADDR   ADDRESS OF INTERRUPTING CONTROLLER'S DCT
;        4858  !-
;        4859
;        4860      begin
;        4861
;        4862      while not (BIT_TST ((.IDCT_ADDR [RR_POLL] + 2), ED_OWN)) do ! WHILE O = 0
;        4863          begin
;        4864          IPKT_ADDR = ..IDCT_ADDR [RR_POLL] - 10; ! ADDRESS OF RESPONSE PACKET
;        4865          CREDIT_BAL = .CREDIT_BAL + .IPKT_ADDR [CREDITS];
;        4866
;        4867          selectoneu .IPKT_ADDR [CONNID] of
;        4868              set
;        4869
;        4870              [CID_MSCP] :        MSCP_RSP ();
;        4871              [CID_DUP]  :        DUP_RSP ();
;        4872
;        4873  !           [otherwise] :       PRINTF (DBM20, .IPKT_ADDR [CONNID], .IRDRX_ADDR);       !JSD REV A
;        4874                                                  ! "BAD CONN ID = XXXXX RECEIVED FROM XXXXXX"
;        4875              tes;
;        4876
;        4877          IPKT_ADDR [MSGLEN] = MSG_LEN + 2;               ! RE-INIT PKT FIELDS; MESSAGE LENGTH
;        4878          IDCT_ADDR [RR_POLL] = .IDCT_ADDR [RR_POLL] + 2; ! ADVANCE TO HI ORDER WORD OF RING SLOT
;        4879          .IDCT_ADDR [RR_POLL] = .IPKT_ADDR [PKT_HI];     ! RETURN SLOT TO CONTROLLER
;        4880          .IDCT_ADDR [RR_POLL] = ..IDCT_ADDR [RR_POLL] or ED_OWN or ED_FLAG; ! OWNERSHIP TOO
;        4881          IDCT_ADDR [RR_POLL] = .IDCT_ADDR [RR_POLL] + 2; ! ADVANCE TO NEXT RRING SLOT
;        4882
;        4883          if .IDCT_ADDR [RR_POLL] gtra .IDCT_ADDR [RR_END]! IF BEYOND END OF RING
;        4884          then
;        4885              IDCT_ADDR [RR_POLL] = .IDCT_ADDR [RR_BEG];  ! CYCLE TO TOP OF RING
;        4886
;        4887          end;                                            ! WHILE LOOP
;        4888
;        4889      ICOM_ADDR [RSP_INT] = 0;                            ! CLEAR RESPONSE INTERRUPT WORD IN RING HEADER
;        4890      end;


                                        .SBTTL  POLL.RRING RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G          POLL.RRING:
                                        JSR     R1,$SAVE2                       ;                       4843
000004  013702  000110'                 MOV     IDCT.ADDR,R2                    ;                       4862
000010  062702  000014                  ADD     #14,R2
000014  011200                  1$:     MOV     (R2),R0
```

```
000016  016000  000002                        MOV     2(R0),R0
000022  042700  077777                        BIC     #77777,R0
000026  020027  100000                        CMP     R0,#-100000
000032  001467                                BEQ     4$
000034  017237  000000  000000G               MOV     @0(R2),IPKT.ADDR        ;                       4864
000042  162737  000012  000000G               SUB     #12,IPKT.ADDR
000050  013700  000000G                       MOV     IPKT.ADDR,R0            ;                       4865
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  116000  000011                        MOVB    11(R0),R0               ;                       4867
000100  042700  177400                        BIC     #177400,R0
000104  001003                                BNE     2$
000106  004737  000000V                       JSR     PC,MSCP.RSP             ;                       4870
000112  000405                                BR      3$                      ;                       4867
000114  020027  000002              2$:       CMP     R0,#2
000120  001002                                BNE     3$
000122  004737  000000V                       JSR     PC,DUP.RSP              ;                       4871
000126  013700  000000G              3$:       MOV     IPKT.ADDR,R0            ;                       4877
000132  012760  000074  000006               MOV     #74,6(R0)
000140  013701  000110'                       MOV     IDCT.ADDR,R1            ;                       4878
000144  010102                                MOV     R1,R2
000146  062702  000014                        ADD     #14,R2
000152  062712  000002                        ADD     #2,(R2)
000156  016072  000002  000000               MOV     2(R0),@0(R2)            ;                       4879
000164  052772  140000  000000               BIS     #-40000,@0(R2)          ;                       4880
000172  062712  000002                        ADD     #2,(R2)                 ;                       4881
000176  021261  000006                        CMP     (R2),6(R1)              ;                       4883
000202  101704                                BLOS    1$
000204  016112  000004                        MOV     4(R1),(R2)              ;                       4885
000210  000701                                BR      1$                      ;                       4862
000212  013700  000104'              4$:       MOV     ICOM.ADDR,R0            ;                       4889
000216  005060  000006                        CLR     6(R0)
000222  000207                                RTS     PC                      ;                       4843

. Routine Size:  74 words,      Routine Base:  $CODE$ + 21562
. Maximum stack depth per invocation:  4 words
```

```
;        4891  !+
;        4892  ROUTINE DUP_RSP : NOVALUE =
;        4893
;        4894  !+
;        4895  !       THIS ROUTINE IS CALLED BY POLL_RRING FOR EACH DUP RESPONSE
;        4896  !       ITS GENERAL PURPOSE IS TO ACT ON A DATAGRAM OR SEQUENTIAL MESSAGE.
.        4897  !       IF THE MESSAGE TYPE IS SEQUENTIAL, THE ROUTINE COPIES THE
.        4898  !       CONTENTS OF THE MESSAGE ENVELOPE INTO A RETURN PACKET SO THAT THE
.        4899  !       ENVELOPE CAN BE RETURNED TO THE CONTROLLER.
.        4900  !
.        4901  !       IMPLICIT INPUTS:
.        4902  !               ICTLR   INTERRUPTING CONTROLLER NUMBER
.        4903  !               IPKT_ADDR   ADDRESS OF MSCP ENVELOPE CONTAINING RESPONSE
.        4904  !-
;        4905      begin
.        4906
.        4907      local
.        4908          R_INDEX : signed word,
;        4909          SRC_ADDR,
;        4910          DST_ADDR,
;        4911          R_ADDR : ref block [RP_LEN, word] field (RP_FIELDS);
.        4912      !PRINTX (DER34);
.        4913
;        4914      incr COUNT from 0 to PKT_CNT - 1 do
;        4915
;        4916          if (.MSCP_PKT [.COUNT, CRN_LO] eql .IPKT_ADDR [CRN_LO]) and     ! IF THIS IS THE ASSOC CMD
;        4917             (.MSCP_PKT [.COUNT, CRN_HI] eql .IPKT_ADDR [CRN_HI]) and
;        4918             (.MSCP_PKT [.COUNT, PKT_LO] neqa .IPKT_ADDR [PKT_LO]) and
;        4919             ((.MSCP_PKT [.COUNT, OPCODE] and OP_END) neq OP_END) and
;        4920             (.MSCP_PKT [.COUNT, MSGTYP] eql MT_SEQ) and
;        4921             (.MSCP_PKT [.COUNT, CONNID] eql CID_DUP) and
;        4922             ((.IPKT_ADDR [OPCODE] and OP_END) eql OP_END)
;        4923          then
;        4924              begin
;        4925              P_INDEX = .COUNT;                      ! SET PKT NUMBER
.        4926              exitloop;
;        4927              end;
;        4928
;        4929      if .P_INDEX lss 0                              ! IF COMMAND NOT FOUND
.        4930      then
.        4931          begin
;        4932  !       PRINTF (DBM108, .IPKT_ADDR [CRN_LO]);      ! UNKNOWN COMMAND REF. NUMBER   !JSD REV A
.        4933          return;
.        4934          end;
.        4935
.        4936      if (R_INDEX = GET_RETPKT (.ICTLR)) lss 0       ! IF RETPKT IS NOT AVAILABLE
;        4937      then
.        4938  !       PRINTF (DBM112)                            ! "DUP-RSP: RETPKT NOT AVAILABLE"       !JSD REV A
.        4939      else
.        4940          begin
.        4941          SRC_ADDR = .IPKT_ADDR + 6;                 ! SET UP COPY (SKIP OVER PKT DESC)
.        4942          R_ADDR = DST_ADDR = RETPKT + (.R_INDEX * RP_LEN * 2);    ! START OF ALLOCATED RETPKT
.        4943
.        4944          incr COUNT from 1 to RP_LEN do
.        4945              begin
.        4946              .DST_ADDR = ..SRC_ADDR;                ! COPY 1 WORD
```

```
;        4947              DST_ADDR = .DST_ADDR + 2;          ! ADVANCE DESTINATION ADDR
;        4948              SRC_ADDR = .SRC_ADDR + 2;          ! ADVANCE SOURCE ADDR
;        4949              end;                               ! COPY LOOP
;        4950
;        4951          R_ADDR [CTLR] = .ICTLR;                ! LOAD CONTROLLER NUMBER INTO PACKET
;        4952          IN_IODQ (.R_INDEX);                    ! PUT RETPKT INDEX INTO IODQ
;        4953          end;                                   ! IF RETPKT WAS ALLOCATED
;        4954
;        4955      if .P_INDEX geq 0                          ! IF ASSOC CMD PKT WAS FOUND
;        4956      then
;        4957          PUT_PKT (.P_INDEX);                    ! RETURN COMMAND PACKET TO POOL
;        4958
;        4959      end;                                       ! ROUTINE DUP_RSP

                                        .SBTTL  DUP.RSP RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G            DUP.RSP:JSR     R1,$SAVE4                       ;                       4892
000004  013701  000000G                    MOV     IPKT.ADDR,R1                    ;                       4916
000010  005002                            CLR     R2                              ; COUNT                 4914
000012  010246                    1$:     MOV     R2,-(SP)                        ; COUNT,*               4916
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)          ;                       4917
000044  001024                            BNE     2$
000046  026011  000000G                    CMP     MSCP.PKT(R0),(R1)               ;                       4918
000052  001421                            BEQ     2$
000054  105760  000022G                    TSTB    MSCP.PKT+22(R0)                 ;                       4919
000060  100416                            BMI     2$
000062  132760  000360  000010G           BITB    #360,MSCP.PKT+10(R0)            ;                       4920
000070  001013                            BNE     2$
000072  126027  000011G 000002            CMPB    MSCP.PKT+11(R0),#2              ;                       4921
000100  001006                            BNE     2$
000102  105761  000022                    TSTB    22(R1)                          ;                       4922
000106  100003                            BPL     2$
000110  010237  000000G                    MOV     R2,P.INDEX                      ; COUNT,*               4925
000114  000406                            BR      3$                              ;                       4924
000116  005202                    2$:     INC     R2                              ; COUNT                 4914
000120  020227  000013                    CMP     R2,#13                          ; COUNT,*
000124  003732                            BLE     1$
000126  005737  000000G                    TST     P.INDEX                         ;                       4929
000132  002455                    3$:     BLT     6$                              ;                       4931
000134  013746  000156'                    MOV     ICTLR,-(SP)                     ;                       4936
000140  004737  000000G                    JSR     PC,GET.RETPKT
000144  010004                            MOV     R0,R4                           ; *,R.INDEX
000146  005726                            TST     (SP)+
000150  005704                            TST     R4                              ; R.INDEX
000152  002436                            BLT     5$
000154  013702  000000G                    MOV     IPKT.ADDR,R2                    ; *,SRC.ADDR            4941
000160  062702  000006                    ADD     #6,R2                           ; *,SRC.ADDR
000164  010446                            MOV     R4,-(SP)                        ; R.INDEX,*             4942
000166  012746  000060                    MOV     #60,-(SP)
000172  004737  000000G                    JSR     PC,BL$MUL
000176  062700  000000G                    ADD     #RETPKT,R0
```

```
000202  010003                          MOV     R0,R3                   ; *,DST.ADDR
000204  012701  000030                  MOV     #30,R1                  ; *,COUNT                       4944
000210  012223                  4$:     MOV     (R2)+,(R3)+             ; SRC.ADDR,DST.ADDR             4946
000212  005301                          DEC     R1                      ; COUNT                         4944
000214  001375                          BNE     4$
000216  013701  000156'                  MOV     ICTLR,R1                ;                               4951
000222  042701  177760                  BIC     #177760,R1
000226  142760  000017  000002          BICB    #17,2(R0)               ; *,*(R.ADDR)
000234  150160  000002                  BISB    R1,2(R0)                ; *,*(R.ADDR)
000240  010416                          MOV     R4,(SP)                 ; R.INDEX,*                     4952
000242  004737  G00000G                  JSR     PC,IN.IODQ
000246  022626                          CMP     (SP)+,(SP)+             ;                               4940
000250  013700  000000G           5$:    MOV     P.INDEX,R0              ;                               4955
000254  002404                          BLT     6$
000256  010046                          MOV     R0,-(SP)                ;                               4957
000260  004737  000000G                  JSR     PC,PUT.PKT
000264  005726                          TST     (SP)+
000266  000207                  6$:     RTS     PC                      ;                               4892

; Routine Size:  92 words,      Routine Base:  $CODE$ + 22006
; Maximum stack depth per invocation:  8 words
```

```
;    routine MSCP_RSP : novalue =
;
;    !+
;    !       THIS ROUTINE IS CALLED BY POLL_RRING FOR EACH RESPONSE MESSAGE
;    !       WHICH HAS A CONNECTION ID INDICATING A DISK MSCP ORIGINATOR
;    !       (I.E., ALL EXCEPT DUP RESPONSES). ITS PURPOSE IS TO PASS
;    !       CONTROL TO THE APPROPRIATE ROUTINE BASED ON THE MESSAGE TYPE
;    !       FIELD (SEQUENTIAL, DATAGRAM, OR CREDIT NOTIFICATION).
;    !
;    !       IMPLICIT INPUTS:
;    !               IPKT_ADDR    ADDRESS OF MSCP PACKET CONTAINING RESPONSE
;    !                            MESSAGE
;    !
;
;        selectoneu .IPKT_ADDR [MSGTYP] of
;            set
;
;            [MT_SEQ] :        SEQUEN ();
;
;            [MT_DG] :         DATAGM ();
;
;    !       [otherwise] .     PRINTF (DBM21, .IPKT_ADDR [MSGTYP]);    ! "MESSAGE TYPE XX RECEIVED"     !JSD REV A
;            tes;

                                                  .SBTTL  MSCP.RSP RDRX INTERRUPT SERVICE ROUTINES
000000  013700  000000G                  MSCP.RSP:
                                                  MOV     IPKT.ADDR,R0                    ;          4974
000004  116000  000010                            MOVB    10(R0),R0
000010  006200                                    ASR     R0
000012  006200                                    ASR     R0
000014  006200                                    ASR     R0
000016  006200                                    ASR     R0
000020  042700  177760                            BIC     #177760,R0
000024  001003                                    BNE     1$
000026  004737  000000V                           JSR     PC,SEQUEN                       ;          4977
000032  000207                                    RTS     PC                              ;          4974
000034  020027  000001                   1$:      CMP     R0,#1
000040  001002                                    BNE     2$
000042  004737  000000V                           JSR     PC,DATAGM                       ;          4979
000046  000207                           2$:      RTS     PC                              ;          4960

; Routine Size:  20 words,      Routine Base:  $CODE$ + 22276
; Maximum stack depth per invocation:  1 word
```

```
routine SEQUEN : novalue =

!+
!       THIS ROUTINE IS CALLED BY MSCP_RSP FOR EACH DISK MSCP RESPONSE MESSAGE
!       WITH THE "SEQUENTIAL" MESSAGE TYPE. ITS GENERAL PURPOSE IS TO COPY THE
!       CONTENTS OF THE MESSAGE PACKET INTO A RETURN PACKET SO THAT THE
!       PACKET CAN BE RETURNED TO THE CONTROLLER.   IN ADDITION,
!       IF THE COMMAND WAS AN I/O TRANSFER (READ, WRITE, OR ACCESS), THEN SOME
!       FIELDS OF THE COMMAND PACKET ARE COPIED INTO THE RETURN PACKET.
!
!       IMPLICIT INPUTS:
!               ICTLR   INTERRUPTING CONTROLLER NUMBER
!               IPKT_ADDR - ADDRESS OF MSCP PACKET CONTAINING RESPONSE
!


    begin

    local
        R_INDEX : signed word,
        SRC_ADDR,
        DST_ADDR,
        R_ADDR : ref block [RP_LEN, word] field (RP_FIELDS),
        TEMP_UNIT,
        SFT_ERR_PRINTED : byte initial (byte (FALSE));

    incr COUNT from 0 to PKT_CNT   1 do

        if (.MSCP_PKT [.COUNT, CRN_LO] eql .IPKT_ADDR [CRN_LO]) and      ! IF THIS IS THE ASSOC CMD
            (.MSCP_PKT [.COUNT, CRN_HI] eql .IPKT_ADDR [CRN_HI]) and
            (.MSCP_PKT [.COUNT, PKT_LO] neqa .IPKT_ADDR [PKT_LO]) and
            ((.MSCP_PKT [.COUNT, OPCODE] and OP_END) neq OP_END) and
            (.MSCP_PKT [.COUNT, MSGTYP] eql MT_SEQ) and
            ((.IPKT_ADDR [OPCODE] and OP_END) eql OP_END) and
            (.PKT_USE [.count] eql .ICTLR)                               ! don't want old packets from other controll
er
        then
            begin
            P_INDEX = .COUNT;                     ! SET PKT NUMBER
            exitloop;
            end;

    if .P_INDEX lss 0                             ! IF COMMAND NOT FOUND
    then
        begin
!       PRINTF (DBM108, .IPKT_ADDR [CRN_LO]);    ! UNKNOWN COMMAND REF. NUMBER   !JSD REV A
        return;
        end;

    if (R_INDEX = GET_RETPKT (.ICTLR)) lss 0     ! IF RETPKT IS NOT AVAILABLE
    then
!       PRINTF (DBM22)                            ! "SEQUEN: RETPKT NOT AVAILABLE"
    else
        begin
        SRC_ADDR = .IPKT_ADDR + 6;                ! SET UP COPY (SKIP OVER PKT DESC)
        R_ADDR = DST_ADDR = RETPKT + (.R_INDEX * RP_LEN * 2);   ! START OF ALLOCATED RETPKT

        incr COUNT from 1 to RP_LEN do
```

```
;          5039              begin
;          5040              .DST_ADDR = ..SRC_ADDR;                 ! COPY 1 WORD
;          5041              DST_ADDR = .DST_ADDR + 2;               ! ADVANCE DESTINATION ADDR
;          5042              SRC_ADDR = .SRC_ADDR + 2;               ! ADVANCE SOURCE ADDR
;          5043
;          5044              if .IPKT_ADDR [OPCODE] eql (OP_ONL or OP_END)       ! IF THIS IS THE ONLINE END MESSAGE
;          5045              then
;          5046
;          5047                  if .COUNT eql 10                    ! SKIP OVER RE    cD WORDS
;          5048                  then
;          5049                      SRC_ADDR = .SRC_ADDR + 4;       !       IN ONLINE END   MESSAGE
;          5050
;          5051              end;                                    ! COPY LOOP
;          5052
;          5053          R_ADDR [CTLR] = .ICTLR;                     ! LOAD CONTROLLER NUMBER INTO PACKET
;          5054
;          5055          if .P_INDEX geq 0                           ! IF ASSOC. CMD PKT WAS FOUND
;          5056          then
;          5057
;          5058              if (.IPKT_ADDR [OPCODE] eql (OP_RD or OP_END)) or   ! IF END MESSAGE IS
;          5059                 (.IPKT_ADDR [OPCODE] eql (OP_WRT or OP_END)) or  !      READ, WRITE, OR
;          5060                 (.IPKT_ADDR [OPCODE] eql (OP_ACC or OP_END))     !      ACCESS
;          5061              then
;          5062                  begin
;          5063                  R_ADDR [CMDMOD] = .MSCP_PKT [.P_INDEX, MODIFY]; ! COPY
;          5064                  R_ADDR [CBCNT_LO] = .MSCP_PKT [.P_INDEX, BC_LO];! RELEVANT
;          5065                  R_ADDR [CBCNT_HI] = .MSCP_PKT [.P_INDEX, BC_HI];! FIELDS
;          5066                  R_ADDR [LBN_LO] = .MSCP_PKT [.P_INDEX, LBN_L];  ! FROM
;          5067                  R_ADDR [LBN_HI] = .MSCP_PKT [.P_INDEX, LBN_H];  ! COMMAND
;          5068                  R_ADDR [BUFF_0] = .MSCP_PKT [.P_INDEX, BUF_0];  ! PACKET
;          5069                  R_ADDR [BUFF_1] = .MSCP_PKT [.P_INDEX, BUF_1];  ! TO RETPKT
;          5070                  end;                                            ! IF ENDCODE WAS READ, WRITE, OR ACCESS
;          5071
;          5072                  IN_IODQ (.R_INDEX);                             ! PUT RETPKT INDEX INTO IODQ
;          5073                  end;                                            ! IF RETPKT WAS ALLOCATED
;          5074
;          5075          if (.IPKT_ADDR [STATUS_CODE] neq  ST_SUC) or
;          5076             (.IPKT_ADDR [STATUS_SUBCODE] neq 0)
;          5077          then
;          5078              LAST_PKT [.ICTLR, LAST_HRD_ERR] = HRD_OCCURED           ! SAVE ERROR CONDITION
;          5079          else
;          5080              LAST_PKT [.ICTLR, LAST_HRD_ERR] = HRD_NOT_OCCURED;      !
;          5081
;          5082          LAST_PKT [.ICTLR, LAST_CRN_LO] = .IPKT_ADDR [CRN_LO];       ! SAVE COMMAND REFERENCE NUMBER
;          5083          LAST_PKT [.ICTLR, LAST_CRN_HI] = .IPKT_ADDR [CRN_HI];       !
;          5084
;          5085          incr index from 0 to EP_CNT - 1 do                          ! IF CORRESPONDING REF NUM HAD AN ERROR LOG
;          5086
;          5087              if (.ELOG_PKT [.index, EL_CNTR] eql .ICTLR) and
;          5088                 (.ELOG_PKT [.index, EL_CRN_LO] eql .IPKT_ADDR [CRN_LO]) and
;          5089                 (.ELOG_PKT [.index, EL_CRN_HI] eql .IPKT_ADDR [CRN_HI]) and
;          5090                 (.ELOG_PKT [.index, EL_CONTENTS] eql FULL)
;          5091              then
;          5092                  begin
;          5093
;          5094                  if .LAST_PKT [.ICTLR, LAST_HRD_ERR] eql HRD_NOT_OCCURED     ! IF SOFT ERROR OCCURED
```

```
;          5095                then
;          5096
;          5097                    if .ELOG_PKT [.index, EL_FORMAT] lequ 4
;          5098                    then
;          5099                        begin
;          5100                        SOFT_ERROR (.index);                        ! UPATE SOFT ERROR COUNT
;          5101                        TEMP_UNIT = .L$LUN;                         ! SAVE UNIT NUMBER AS KNOWN TO DRS
;          5102
;          5103                        incr OFFSET from (0 * OF_UN) to (3 * UNIT_SIZE * OF_UN) by UNIT_SIZE do
;          5104
;          5105                            if (.ICST_ADDR [.OFFSET, D_DISK_NUM] eql .ELOG_PKT [.index, EL_DK_NUM]) and
;          5106                               (.ICST_ADDR [.OFFSET, D_PRES] eql PRESENT)
;          5107                            then
;          5108                                begin
;          5109                                L$LUN = .ICST_ADDR [.OFFSET, D_UNIT];       ! CORECT UNIT NUMBER FOR ERROR MESSAGE
;          5110                                exitloop;
;          5111                                end;
;          5112
;          5113                            case .ELOG_PKT [.index, EL_FORMAT] from 0 to 4 of
;          5114                                set
;          5115
;          5116                                [0]:    ERRSOFT (60, 0, 0);             ! CONTROLLER ERROR
;          5117
;          5118                                [1]:    ERRSOFT (61, 0, 0);             ! HOST MEMORY ACCESS ERROR
;          5119
;          5120                                [2]:    ERRSOFT (62, 0, 0);             ! DISK TRANSFER ERROR
;          5121
;          5122                                [3]:    ERRSOFT (63, 0, 0);             ! SDI ERROR
;          5123
;          5124                                [4]:    ERRSOFT (64, 0, 0);             ! SMALL DISK ERROR
;          5125                                tes;
;          5126
;          5127                                L$LUN = .TEMP_UNIT;                     ! RESTORE UNIT NUMBER
;          5128                                SFT_ERR_PRINTED = TRUE;                 ! SOFT ERROR PRINTOUT OCCURED
;          5129                                end
;          5130
;          5131                            else
;          5132  !                             PRINTF (DBM109, .ELOG_PKT [.index, EL_FORMAT]);     ! UNKNOWN ERROR-LOG FORMAT      !JSD REV A
;          5133
;          5134                    if not (.SFT_ERR_PRINTED)
;          5135                    then
;          5136                        PRINTB (CRLF);                              ! EXTRA CARRIAGE-RETURN/LINE FEED
;          5137
;          5138                    EMS_EL (.index);                                ! PRINT ERROR-LOG CONTENTS
;          5139                    end;
;          5140
;          5141                if .P_INDEX geq 0                                   ! IF ASSOC CMD PKT WAS FOUND
;          5142                then
;          5143                    PUT_PKT (.P_INDEX);                             ! RETURN COMMAND PACKET TO POOL
;          5144
;          5145                end;                                                ! ROUTINE DISK_RSP

                                        .SBTTL  SEQUEN RDRX INTERRUPT SERVICE ROUTINES
000000  004137  000000G           SEQUEN: JSR     R1,$SAVE5                       ;                                   4983
000004  105046                          CLRB    (SP)                            ; SFT.ERR.PRINTED                   4998
```

```
000006  013701  000000G                    MOV     IPKT.ADDR,R1                    ;                               5010
000012  005002                              CLR     R2                              ; COUNT                         5008
000014  010246                      1$:     MOV     R2, (SP)                        ; COUNT,*                       5010
000016  012746  000104                      MOV     #104,-(SP)
000022  004737  000000G                    JSR     PC,BL$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)          ;                               5011
000046  001025                              BNE     2$
000050  026011  000000G                    CMP     MSCP.PKT(R0),(R1)               ;                               5012
000054  001422                              BEQ     2$
000056  105760  000022G                    TSTB    MSCP.PKT+22(R0)                 ;                               5013
000062  100417                              BMI     2$
000064  132760  000360  000010G            BITB    #360,MSCP.PKT+10(R0)            ;                               5014
000072  001013                              BNE     2$
000074  105761  000022                      TSTB    22(R1)                          ;                               5015
000100  100010                              BPL     2$
000102  116200  000000G                    MOVB    PKT.USE(R2),R0                  ; *(COUNT),*                    5016
000106  020037  000156'                    CMP     R0,ICTLR
000112  001003                              BNE     2$
000114  010237  000000G                    MOV     R2,P.INDEX                      ; COUNT,*                       5019
000120  000406                              BR      3$                              ;                               5018
000122  005202                      2$:     INC     R2                              ; COUNT                         5008
000124  020227  000013                      CMP     R2,#13                          ; COUNT,*
000130  003731                              BLE     1$
000132  005737  000000G                    TST     P.INDEX                         ;                               5023
000136  002002                      3$:     BGE     4$                              ;                               5025
000140  000137  023516'                    JMP     28$
000144  013746  000156'              4$:     MOV     ICTLR,-(SP)                     ;                               5030
000150  004737  000000G                    JSR     PC,GET.RETPKT
000154  010005                              MOV     R0,R5                           ; *,R.INDEX
000156  005726                              TST     (SP)+
000160  005705                              TST     R5                              ; R.INDEX
000162  002526                              BLT     9$
000164  013704  000000G                    MOV     IPKT.ADDR,R4                    ; *,SRC.ADDR                    5035
000170  062704  000006                      ADD     #6,R4                           ; *,SRC.ADDR
000174  010546                              MOV     R5,-(SP)                        ; R.INDEX,*                     5036
000176  012746  000060                      MOV     #60,-(SP)
000202  004737  000000G                    JSR     PC,BL$MUL
000206  062700  000000G                    ADD     #RETPKT,R0
000212  010003                              MOV     R0,R3                           ; *,DST.ADDR
000214  010002                              MOV     R0,R2                           ; *,R.ADDR
000216  013701  000000G                    MOV     IPKT.ADDR,R1                    ;                               5044
000222  012700  000001                      MOV     #1,R0                           ; *,COUNT                       5038
000226  012423                      5$:     MOV     (R4)+,(R3)+                     ; SRC.ADDR,DST.ADDR             5040
000230  126127  000022  000211              CMPB    22(R1),#211                                                     5044
000236  001005                              BNE     6$
000240  020027  000012                      CMP     R0,#12                          ; COUNT,*                       5047
000244  001002                              BNE     6$
000246  062704  000004                      ADD     #4,R4                           ; *,SRC.ADDR                    5049
000252  005200                      6$:     INC     R0                              ; COUNT                         5038
000254  020027  000030                      CMP     R0,#30                          ; COUNT,*
000260  003762                              BLE     5$
000262  013700  000156'                    MOV     ICTLR,R0                        ;                               5053
000266  042700  177760                      BIC     #177760,R0
```

```
000272  142762  000017  000002                  BICB    #17,2(R2)                       ; *,*(R.ADDR)
000300  150062  000002                          BISB    R0,2(R2)                        ; *,*(R.ADDR)
000304  013704  000000G                         MOV     P.INDEX,R4                      ;                                       5055
000310  002447                                  BLT     8$
000312  005000                                  CLR     R0                              ;                                       5058
000314  156100  000022                          BISB    22(R1),R0
000320  020027  000241                          CMP     R0,#241
000324  001406                                  BEQ     7$
000326  020027  000242                          CMP     R0,#242                         ;                                       5059
000332  001403                                  BEQ     7$
000334  020027  000220                          CMP     R0,#220                         ;                                       5060
000340  001033                                  BNE     8$
000342  C10416                          7$:     MOV     R4,(SP)                         ;                                       5063
000344  012746  000104                          MOV     #104,-(SP)
000350  004737  000000G                         JSR     PC,BL$MUL
000354  016062  000024G 000012                  MOV     MSCP.PKT+24(R0),12(R2)          ; *,*(R.ADDR)
000362  016062  000026G 000044                  MOV     MSCP.PKT+26(R0),44(R2)          ; *,*(R.ADDR)                           5064
000370  016062  000030G 000046                  MOV     MSCP.PKT+30(R0),46(R2)          ; *,*(R.ADDR)                           5065
000376  016062  000046G 000050                  MOV     MSCP.PKT+46(R0),50(R2)          ; *,*(R.ADDR)                           5066
000404  016062  000050G 000052                  MOV     MSCP.PKT+50(R0),52(R2)          ; *,*(R.ADDR)                           5067
000412  016062  000032G 000024                  MOV     MSCP.PKT+32(R0),24(R2)          ; *,*(R.ADDR)                           5068
000420  016062  000034G 000026                  MOV     MSCP.PKT+34(R0),26(R2)          ; *,*(R.ADDR)                           5069
000426  005726                                  TST     (SP)+                           ;                                       5062
000430  010516                          8$:     MOV     R5,(SP)                         ; R.INDEX,*                             5072
000432  004737  000000G                         JSR     PC IN.IODQ
000436  022626                                  CMP     (SP)+,(SP)+                     ;                                       5034
000440  013746  000156'                  9$:     MOV     ICTLR,-(SP)                     ;                                       5078
000444  012746  000006                          MOV     #6,-(SP)
000450  004737  000000G                         JSR     PC,BL$MUL
000454  013702  000000G                         MOV     IPKT.ADDR,R2                    ;                                       5075
000460  012703  000024                          MOV     #24,R3
000464  060203                                  ADD     R2,R3
000466  132713  000037                          BITB    #37,(R3)
000472  001003                                  BNE     10$
000474  032713  177740                          BIT     #177740,(R3)                    ;                                       5076
000500  001404                                  BEQ     11$
000502  012760  000001  000170'          10$:    MOV     #1,LAST.PKT(R0)                 ;                                       5078
000510  000402                                  BR      12$                             ;                                       5075
000512  005060  000170'                  11$:    CLR     LAST.PKT(R0)                    ;                                       5080
000516  016260  000012  000172'          12$:    MOV     12(R2),LAST.PKT+2(R0)           ;                                       5082
000524  016260  000014  000174'                  MOV     14(R2),LAST.PKT+4(R0)           ;                                       5083
000532  005003                                  CLR     R3                              ; INDEX                                 5085
000534  010316                          13$:    MOV     R3,(SP)                         ; INDEX,*                               5087
000536  012746  000102                          MOV     #102,-(SP)
000542  004737  000000G                         JSR     PC,BL$MUL
000546  010005                                  MOV     R0,R5
000550  005726                                  TST     (SP)+
000552  005000                                  CLR     R0
000554  156500  000000G                         BISB    ELOG.PKT(R5),R0
000560  020037  000156'                          CMP     R0,ICTLR
000564  001156                                  BNE     26$
000566  013700  000000G                         MOV     IPKT.ADDR,R0                    ;                                       5088
000572  026560  000006G 000012                  CMP     ELOG.PKT+6(R5),12(R0)
000600  001150                                  BNE     26$
000602  026560  000010G 000014                  CMP     ELOG.PKT+10(R5),14(R0)          ;                                       5089
000610  001144                                  BNE     26$
```

```
000612  126527  000001G 000001          CMPB    ELOG.PKT+1(R5),#1               ;                               5090
000620  001140                          BNE     26$
000622  013716  000156'                 MOV     ICTLR,(SP)                      ;                               5094
000626  012746  000006                  MOV     #6,-(SP)
000632  004737  000000G                 JSR     PC,BL$MUL
000636  005726                          TST     (SP)+
000640  005760  000170'                 TST     LAST.PKT(R0)
000644  001123                          BNE     25$
000646  126527  000016G 000004          CMPB    ELOG.PKT+16(R5),#4              ;                               5097
000654  101104                          BHI     24$
000656  010316                          MOV     R3,(SP)                         ; INDEX,*                       5100
000660  004737  000000V                 JSR     PC,SOFT.ERROR
000664  013702  000000G                 MOV     L$LUN,R2                        ; *,TEMP.UNIT                   5101
000670  012700  000006                  MOV     #6,R0                           ; *,OFFSET                      5103
000674  010004                  14$:    MOV     R0,R4                           ; OFFSET,*                      5105
000676  063704  000106'                 ADD     ICST.ADDR,R4
000702  016546  000012G                 MOV     ELOG.PKT+12(R5),-(SP)
000706  111401                          MOVB    (R4),R1
000710  042701  177774                  BIC     #177774,R1
000714  020126                          CMP     R1,(SP)+
000716  001012                          BNE     15$
000720  032714  040000                  BIT     #40000,(R4)                     ;                               5106
000724  001407                          BEQ     15$
000726  011401                          MOV     (R4),R1                         ;                               5109
000730  000301                          SWAB    R1
000732  042701  177760                  BIC     #177760,R1
000736  010137  000000G                 MOV     R1,L$LUN
000742  000405                          BR      16$                             ;                               5108
000744  062700  000012          15$:    ADD     #12,R0                          ; *,OFFSET                      5103
000750  020027  000044                  CMP     R0,#44                          ; OFFSET,*
000754  003747                          BLE     14$
000756  005000                  16$:    CLR     R0                              ;                               5113
000760  156500  000016G                 BISB    ELOG.PKT+16(R5),R0
000764  006300                          ASL     R0
000766  066007  000000'                 ADD     P.AAA(R0),PC                    ; Case dispatch
000772  104457                  18$:    TRAP    57                              ;                               5116
000774  000074                          .WORD   74
000776  000000                          .WORD   0
001000  000000                          .WORD   0
001002  000423                          BR      23$                             ;                               5113
001004  104457                  19$:    TRAP    57                              ;                               5118
001006  000075                          .WORD   75
001010  000000                          .WORD   0
001012  000000                          .WORD   0
001014  000416                          BR      23$                             ;                               5113
001016  104457                  20$:    TRAP    57                              ;                               5120
001020  000076                          .WORD   76
001022  000000                          .WORD   0
001024  000000                          .WORD   0
001026  000411                          BR      23$                             ;                               5113
001030  104457                  21$:    TRAP    57                              ;                               5122
001032  000077                          .WORD   77
001034  000000                          .WORD   0
001036  000000                          .WORD   0
001040  000404                          BR      23$                             ;                               5113
001042  104457                  22$:    TRAP    57                              ;                               5124
```

```
001044  000100                                  .WORD   100
001046  000000                                  .WORD   0
001050  000000                                  .WORD   0
001052  010237  000000G                 23$:    MOV     R2,L$LUN                ; TEMP.UNIT,*                   5127
001056  112766  000001  000004                  MOVB    #1,4(SP)                ; *,SFT.ERR.PRINTED             5128
001064  000413                                  BR      25$                     ;                               5097
001066  032766  000001  000004          24$:    BIT     #1,4(SP)                ; *,SFT.ERR.PRINTED             5134
001074  001007                                  BNE     25$
001076  012716  000000G                         MOV     #CRLF,(SP)              ;                               5136
001102  012746  000001                          MOV     #1, (SP)
001106  010600                                  MOV     SP,R0                   ; SP,*
001110  104414                                  TRAP    14
001112  005726                                  TST     (SP)+
001114  010316                          25$:    MOV     R3,(SP)                 ; INDEX,*                       5138
001116  004737  000000G                         JSR     PC,EMS.EL
001122  005203                          26$:    INC     R3                      ; INDEX                         5085
001124  020327  000013                          CMP     R3,#13                  ; INDEX,*
001130  003601                                  BLE     13$
001132  013700  000000G                         MOV     P.INDEX,R0              ;                               5141
001136  002403                                  BLT     27$
001140  010016                                  MOV     R0,(SP)                 ;                               5143
001142  004737  000000G                         JSR     PC,PUT.PKT
001146  022626                          27$:    CMP     (SP)+,(SP)+             ;                               4998
001150  005726                          28$:    TST     (SP)+                   ;                               4983
001152  000207                                  RTS     PC

; Routine Size:  310 words,     Routine Base:  $CODE$ + 22346
; Maximum stack depth per invocation:  12 words


000000                                          .PSECT  $PLIT$,  RO ,  D

                                P.AAA:                                          ; CASE Table for SEQUEN+0766    5113
000000  000000                  17$:    .WORD   0                               ; [18$]
000002  000012                          .WORD   12                              ; [19$]
000004  000024                          .WORD   24                              ; [20$]
000006  000036                          .WORD   36                              ; [21$]
000010  000050                          .WORD   50                              ; [22$]
```

```
;          5146  routine DATAGM : novalue =
;          5147
;          5148  !+
;          5149  !       THIS ROUTINE HANDLES ALL DATAGRAM (ERROR LOG) MESSAGES RECEIVED FROM
;          5150  !       THE RDRX
;          5151  !
;          5152  !       IMPLICIT INPUTS:
;          5153  !               IPKT_ADDR   ADDRESS OF MSCP PACKET CONTAINING ERROR LOG
;          5154  !                           MESSAGE
;          5155  !               ICST_ADDR - ADDRESS OF THE INTERRUPTING CONTROLLER'S CST
;          5156  !-
;          5157
;          5158      begin
;          5159
;          5160      local
;          5161          index : signed word initial (-1),
;          5162          SAVE_ADDR : ref block [EP_LEN, word] field (EP FIELDS),
;          5163          SRC_ADDR,
;          5164          DST_ADDR,
;          5165          TEMP_UNIT,
;          5166          SFT_ERR_PRINTED : byte initial (byte (FALSE));
;          5167
;          5168  !
;          5169  ! FIND AN EMPTY SLOT IN THE ERROR-LOG PACKET SAVE AREA
;          5170  !
;          5171
;          5172      incr COUNT from 0 to EP_CNT - 1 do
;          5173
;          5174          if .ELOG_PKT [.COUNT, EL_CONTENTS] eql EMPTY    ! IF EMPTY SLOT FOUND
;          5175          then
;          5176              begin
;          5177              index = .COUNT;                             ! SAVE INDEX INTO THE SAVE AREA
;          5178              exitloop;
;          5179              end;
;          5180
;          5181  !
;          5182  ! IF AN EMPTY SLOT FOUND, SAVE THE PACKET CONTENTS
;          5183  !
;          5184
;          5185      if .index geq 0
;          5186      then
;          5187          begin
;          5188          SAVE_ADDR = ELOG_PKT + (.index * EP_LEN * 2);   ! ADDRESS OF THE SAVE AREA
;          5189          SAVE_ADDR [EL_CONTENTS ] = FULL;                ! MARK IT FULL
;          5190          SAVE_ADDR [EL_CNTR] = .ICTLR;                   ! OWNERSHIP
;          5191          SRC_ADDR = .IPKT_ADDR + 6;                      ! SETUP COPY ADDRESSES
;          5192          DST_ADDR = .SAVE_ADDR + 2;                      !
;          5193
;          5194          incr COUNT from 1 to ((.IPKT_ADDR [MSGLEN] + 1) / 2) + 2 do
;          5195              begin
;          5196              .DST_ADDR = ..SRC_ADDR;                     ! COPY A WORD
;          5197              SRC_ADDR = .SRC_ADDR + 2;                   ! UPDATE ADDRESS POINTERS
;          5198              DST_ADDR = .DST_ADDR + 2;                   !
;          5199              end;
;          5200
;          5201          end
```

```
;          5202      else
;          5203          begin
;          5204  !       PRINTF (DBM110);                        ! IF EMPTY SLOT NOT FOUND, PRINT MESSAGE        !JSD REV A
;          5205          return;
;          5206          end;
;          5207
;          5208  !
;          5209  ! CHECK IF THE CORRESPONDING RESPONSE HAS ALREADY BEEN RECEIVED
;          5210  !
;          5211
;          5212      if (.SAVE_ADDR [EL_CRN_LO] eql .LAST_PKT [.ICTLR, LAST_CRN_LO]) and
;          5213         (.SAVE_ADDR [EL_CRN_HI] eql .LAST_PKT [.ICTLR, LAST_CRN_HI])
;          5214      then
;          5215          begin
;          5216
;          5217          if .LAST_PKT [.ICTLR, LAST_HRD_ERR] eql HRD_NOT_OCCURED ! IF SOFT ERROR HAD OCCURED
;          5218          then
;          5219
;          5220              if .SAVE_ADDR [EL_FORMAT] lequ 4
;          5221              then
;          5222                  begin
;          5223                  SOFT_ERROR (.index);                ! UPDATE SOFT ERROR COUNT
;          5224                  TEMP_UNIT = .L$LUN;                 ! SAVE UNIT NUMBER AS KNOWN TO DRS
;          5225
;          5226                  incr OFFSET from (0 * OF_UN) to (3 * UNIT_SIZE * OF_UN) by UNIT_SIZE do
;          5227
;          5228                      if (.ICST_ADDR [.OFFSET, D_DISK_NUM] eql .SAVE_ADDR [EL_DK_NUM]) and
;          5229                         (.ICST_ADDR [.OFFSET, D_PRES] eql PRESENT)
;          5230                      then
;          5231                          begin
;          5232                          L$LUN = .ICST_ADDR [.OFFSET, D_UNIT];   ! CORRECT UNIT NUMBER FOR ERROR MESSAGE
;          5233                          exitloop;
;          5234                          end;
;          5235
;          5236                  case .SAVE_ADDR [EL_FORMAT] from 0 to 4 of
;          5237                      set
;          5238
;          5239                      [0] :       ERRSOFT (60, 0, 0);         ! CONTROLLER ERROR
;          5240
;          5241                      [1] :       ERRSOFT (61, 0, 0);         ! HOST MEMORY ACCESS ERROR
;          5242
;          5243                      [2] :       ERRSOFT (62, 0, 0);         ! DISK TRANSFER ERROR
;          5244
;          5245                      [3] :       ERRSOFT (63, 0, 0);         ! SDI ERROR
;          5246
;          5247                      [4] :       ERRSOFT (64, 0, 0);         ! SMALL DISK ERROR
;          5248                      tes;
;          5249
;          5250                  L$LUN = .TEMP_UNIT;                     ! RESTORE UNIT NUMBER
;          5251                  SFT_ERR_PRINTED = TRUE;                 ! SOFT ERROR PRINTOUT OCCURED
;          5252                  end
;          5253
;          5254              else
;          5255  !               PRINTF (DBM109, .SAVE_ADDR [EL_FORMAT]);    ! ERROR LOG FORMAT UNKNOWN      !JSD REV A
;          5256
;          5257          if not (.SFT_ERR_PRINTED)
```

```
;          5258          then
;          5259              PRINTB (CRLF);                    ! EXTRA CARRIEGE RETURN/LINE-FEED
;          5260
;          5261          EMS EL (.index);                      ! PRINT PACKET CONTENTS
;          5262          end;                                  ! CORRESPONDING RESPONSE RECEIVED
;          5263
;          5264      end;


                                        .SBTTL  DATAGM RDRX INTERRUPT SERVICE ROUTINES
023522                                  .PSECT  $CODE$,  RO

000000  004137  000000G        DATAGM: JSR     R1,$SAVE5                ;                                5146
000004  012705  177777                MOV     #-1,R5                   ; *,INDEX                        5158
000010  105046                        CLRB    (SP)                     ; SFT.ERR.PRINTED
000012  005001                        CLR     R1                       ; COUNT                          5172
000014  010146                1$:     MOV     R1,-(SP)                 ; COUNT,*                        5174
000016  012746  000102                MOV     #102,-(SP)
000022  004737  000000G               JSR     PC,BL$MUL
000026  022626                        CMP     (SP)+,(SP)+
000030  105760  000001G               TSTB    ELOG.PKT+1(R0)
000034  001002                        BNE     2$
000036  010105                        MOV     R1,R5                    ; COUNT,INDEX                    5177
000040  000405                        BR      3$                       ;                                5176
000042  005201                2$:     INC     R1                       ; COUNT                          5172
000044  020127  000013                CMP     R1,#13                   ; COUNT,*
000050  003761                        BLE     1$
000052  005705                        TST     R5                       ; INDEX                          5185
000054  002002                3$:     BGE     4$
000056  000137  024236'               JMP     19$
000062  010546                4$:     MOV     R5,-(SP)                 ; INDEX,*                        5188
000064  012746  000102                MOV     #102,-(SP)
000070  004737  000000G               JSR     PC,BL$MUL
000074  062700  000000G               ADD     #ELOG.PKT,R0
000100  010004                        MOV     R0,R4                    ; *,SAVE.ADDR
000102  111764  000001                MOVB    (PC),1(R4)               ; *,*(SAVE.ADDR)                 5189
000106  113714  000156'               MOVB    ICTLR,(R4)               ; *,SAVE.ADDR                    5190
000112  013700  000000G               MOV     IPKT.ADDR,R0             ;                                5191
000116  012702  000006                MOV     #6,R2                    ; *,SRC.ADDR
000122  060002                        ADD     R0,R2                    ; *,SRC.ADDR
000124  012703  000002                MOV     #2,R3                    ; *,DST.ADDR                     5192
000130  060403                        ADD     R4,R3                    ; SAVE.ADDR,DST.ADDR
000132  016016  000006                MOV     6(R0),(SP)               ;                                5194
000136  005216                        INC     (SP)
000140  012746  000002                MOV     #2,-(SP)
000144  004737  000000G               JSR     PC,BL$DIV
000150  062700  000002                ADD     #2,R0
000154  005001                        CLR     R1                       ; COUNT
000156  000401                        BR      6$
000160  012223                5$:     MOV     (R2)+,(R3)+              ; SRC.ADDR,DST.ADDR              5196
000162  005201                6$:     INC     R1                       ; COUNT                          5194
000164  020100                        CMP     R1,R0                    ; COUNT,*
000166  003774                        BLE     5$
000170  022626                        CMP     (SP)+,(SP)+              ;                                5187
000172  013716  000156'               MOV     ICTLR,(SP)               ;                                5212
000176  012746  000006                MOV     #6,-(SP)
```

```
000202  004737  000000G                        JSR     PC,BL$MUL
000206  022626                                  CMP     (SP)+,(SP)+
000210  026460  000006  000172                CMP     6(R4),LAST.PKT+2(R0)        ; *(SAVE.ADDR),*
000216  001136                                  BNE     19$
000220  026460  000010  000174                CMP     10(R4),LAST.PKT+4(R0)       ; *(SAVE.ADDR),*        5213
000226  001132                                  BNE     19$
000230  005760  000170'                        TST     LAST.PKT(R0)                ;                       5217
000234  001123                                  BNE     18$
000236  126427  000016  000004                CMPB    16(R4),#4                   ; *(SAVE.ADDR),*        5220
000244  101105                                  BHI     17$
000246  010546                                  MOV     R5, (SP)                    ; INDEX,*               5223
000250  004737  000000V                        JSR     PC,SOFT.ERROR
000254  013702  000000G                        MOV     L$LUN,R2                    ; *,TEMP.UNIT           5224
000260  012700  000006                        MOV     #6,R0                       ; *,OFFSET              5226
000264  010003                        7$:     MOV     R0,R3                       ; OFFSET,*              5228
000266  063703  000106'                        ADD     ICST.ADDR,R3
000272  016446  000012                        MOV     12(R4),-(SP)                ; *(SAVE.ADDR),*
000276  111301                                  MOVB    (R3),R1
000300  042701  177774                        BIC     #177774,R1
000304  020126                                  CMP     R1,(SP)+
000306  001012                                  BNE     8$
000310  032713  040000                        BIT     #40000,(R3)                 ;                       5229
000314  001407                                  BEQ     8$
000316  011301                                  MOV     (R3),R1                     ;                       5232
000320  000301                                  SWAB    R1
000322  042701  177760                        BIC     #177760,R1
000326  010137  000000G                        MOV     R1,L$LUN
000332  000405                                  BR      9$                          ;                       5231
000334  062700  000012                8$:     ADD     #12,R0                      ; *,OFFSET              5226
000340  020027  000044                        CMP     R0,#44                      ; OFFSET,*
000344  003747                                  BLE     7$
000346  005000                        9$:     CLR     R0                          ;                       5236
000350  156400  000016                        BISB    16(R4),R0                   ; *(SAVE.ADDR),*
000354  006300                                  ASL     R0
000356  066007  000012'                        ADD     P.AAB(P0),PC                ; Case dispatch
000362  104457                        11$:    TRAP    57                          ;                       5239
000364  000074                                  .WORD   74
000366  000000                                  .WORD   0
000370  000000                                  .WORD   0
000372  000423                                  BR      16$                         ;                       5236
000374  104457                        12$:    TRAP    57                          ;                       5241
000376  000075                                  .WORD   75
000400  000000                                  .WORD   0
000402  000000                                  .WORD   0
000404  000416                                  BR      16$                         ;                       5236
000406  104457                        13$:    TRAP    57                          ;                       5243
000410  000076                                  .WORD   76
000412  000000                                  .WORD   0
000414  000000                                  .WORD   0
000416  000411                                  BR      16$                         ;                       5236
000420  104457                        14$:    TRAP    57                          ;                       5245
000422  000077                                  .WORD   77
000424  000000                                  .WORD   0
000426  000000                                  .WORD   0
000430  000404                                  BR      16$                         ;                       5236
000432  104457                        15$:    TRAP    57                          ;                       5247
```

```
000434  000100                                  .WORD   100
000436  000000                                  .WORD   0
000440  000000                                  .WORD   0
000442  010237  000000G                 16$:    MOV     R2,L$LUN                ; TEMP.UNIT,*                   5250
000446  112766  000001  000002                  MOVB    #1,2(SP)                ; *,SFT.ERR.PRINTED             5251
000454  005726                                  TST     (SP)+                   ;                               5222
000456  000412                                  BR      18$                     ;                               5220
000460  032716  000001                 17$:    BIT     #1,(SP)                 ; *,SFT.ERR.PRINTED             5257
000464  001007                                  BNE     18$
000466  012746  000000G                         MOV     #CRLF,-(SP)             ;                               5259
000472  012746  000001                         MOV     #1,-(SP)
000476  010600                                  MOV     SP,R0                   ; SP,*
000500  104414                                  TRAP    14
000502  022626                                  CMP     (SP)+,(SP)+
000504  010546                 18$:    MOV     R5,-(SP)                ; INDEX,*                       5261
000506  004737  000000G                         JSR     PC,EMS.EL
000512  005726                                  TST     (SP)+                   ;                               5215
000514  005726                 19$:    TST     (SP)+                   ;                               5146
000516  000207                                  RTS     PC

; Routine Size:  168 words,    Routine Base:  $CODE$ + 23522
; Maximum stack depth per invocation:  11 words


000012                                  .PSECT  $PLIT$,  RO ,  D

                        P.AAB:                                          ; CASE Table for DATAGM+0356            5236
000012  000000          10$:    .WORD   0                               ; [11$]
000014  000012                  .WORD   12                              ; [12$]
000016  000024                  .WORD   24                              ; [13$]
000020  000036                  .WORD   36                              ; [14$]
000022  000050                  .WORD   50                              ; [15$]
```

```
;          5265  routine SOFT_ERROR (INDEX) : novalue =
;          5266
;          5267  !+
;          5268  !       THIS ROUTINE UPDATES THE SOFT ERROR COUNT IN THE TALLY TABLE FOR EACH
;          5269  !       ERROR LOG MESSAGE RECEIVED
;          5270  !
;          5271  !       IMPLICIT INPUTS:
;          5272  !               ICST_ADDR - ADDRESS OF THE INTERRUPTING CONTROLLER'S CST
;          5273  !
;          5274
;          5275      begin
;          5276
;          5277      local
;          5278          FOUND: byte initial (byte (FALSE)),
;          5279          SOFT_OCCURED : byte initial (byte (FALSE)),
;          5280          UNIT: word,
;          5281          ERROR_CODE : byte,
;          5282          ERROR_SUB : word,
;          5283          TALLY_ADDR : ref block [TALLY_LEN, word] field (T_FIELDS),
;          5284          ELOG_ADDR : ref block [EP_LEN, word] field (EP_FIELDS);
;          5285
;          5286      ELOG_ADDR = ELOG_PKT + (.index * EP_LEN * 2);                          ! ADDRESS OF ERROR L
OG PACKET
;          5287      ERROR_CODE = .ELOG_ADDR [EL_CODE];                                     ! ERROR CODE
;          5288      ERROR_SUB = .ELOG_ADDR [EL_SUBCODE];                                   ! ERROR SUBCODE
;          5289
;          5290      incr OFFSET from (0 * OF_UN) to (3 * UNIT_SIZE * OF_UN) by UNIT_SIZE do
;          5291
;          5292          if (.ICST_ADDR [.OFFSET, D_PRES] eql PRESENT) and                  ! MAP DISK NUMBER TO
 UNIT NUMBER
;          5293             (.ICST_ADDR [.OFFSET, D_DISK_NUM] eql .ELOG_ADDR [EL_DK_NUM])
;          5294          then
;          5295              begin
;          5296              FOUND = TRUE;
;          5297              UNIT = .ICST_ADDR [.OFFSET, D_UNIT];                           ! UNIT NUMBER OF DIS
K
;          5298              exitloop;
;          5299              end;
;          5300
;          5301  !   if (.ELOG_ADDR [EL_SUCCESS]) or
;          5302  !      (.ELOG_ADDR [EL_CONTINUE])
;          5303  !   then
;          5304          SOFT_OCCURED = TRUE;                                           ! SOFT ERROR FLAG
;          5305
;          5306      if .FOUND                                                          ! IF UNIT NUMBER FOU
ND
;          5307      then
;          5308          begin
;          5309          TALLY_ADDR = TALLY + (.UNIT * TALLY_LEN * 2);                  ! ADDRESS OF TALLY T
ABLE
;          5310
;          5311          if .SOFT_OCCURED                                               ! FOR SOFT ERRORRS
;          5312          then
;          5313              selectoneu .ERROR_CODE of
;          5314                  set
;          5315
;          5316                  [ST_MFE]:       TALLY_ADDR [ERR_SFT_SEK] = .TALLY_ADDR [ERR_SFT_SEK] + 1;       ! SOFT   MEDIA FORMA
T
;          5317
;          5318                  [ST_DAT]:       if .ERROR_SUB eql 2                            ! SOFT   DATA
;          5319                                  then
;          5320                                      TALLY_ADDR [ERR_SFT_SEK] = .TALLY_ADDR [ERR_SFT_DAT] + 1
```

```
;  5321                                 else
;  5322                                     TALLY_ADDR [ERR_SFT_DAT] = .TALLY_ADDR [ERR_SFT_DAT] + 1;
;  5323
;  5324                 [ST_MST]:       TALLY_ADDR [ERR_SFT_MST] = .TALLY_ADDR [ERR_SFT_MST] + 1;       ! SOFT - HOST ACCESS
;  5325
;  5326                 [ST_CNT]:       C_ERR_TBL [.ICTLR, C_ERR_SFT] = .C_ERR_TBL [.ICTLR, C_ERR_SFT] + 1;
;  5327                                                                                                 ! SOFT - CONTROLLER
;  5328
;  5329                 [ST_DRV]:       if .ERROR_SUB eql 3                                             ! SOFT - DRIVE
;  5330                                 then
;  5331                                     TALLY_ADDR [ERR_SFT_SEK] = .TALLY_ADDR [ERR_SFT_SEK] + 1
;  5332                                 else
;  5333                                     TALLY_ADDR [ERR_SFT_DRV] = .TALLY_ADDR [ERR_SFT_DRV] + 1;
;  5334                 tes
;  5335             else
;  5336
;  5337                 if (.ELOG_ADDR [EL_CRN_LO] eql 0) and
;  5338                    (.ELOG_ADDR [EL_CRN_HI] eql 0)
;  5339                 then
;  5340                     selectoneu .ERROR_CODE of
;  5341                         set
;  5342
;  5343                         [ST_MFE]:   TALLY_ADDR [ERR_HRD_SEK] = .TALLY_ADDR [ERR_HRD_SEK] + 1;       ! HARD - MEDIA FORMA
;  5344
;  5345                         [ST_DAT]:   if .ERROR_SUB eql 2                                             ! HARD - DATA
;  5346                                     then
;  5347                                         TALLY_ADDR [ERR_HRD_SEK] = .TALLY_ADDR [ERR_HRD_SEK] + 1
;  5348                                     else
;  5349                                         TALLY_ADDR [ERR_HRD_DAT] = .TALLY_ADDR [ERR_HRD_DAT] + 1;
;  5350
;  5351                         [ST_HST]:   TALLY_ADDR [ERR_HRD_HST] = .TALLY_ADDR [ERR_HRD_HST] + 1;       ! HARD - HOST ACCESS
;  5352
;  5353                         [ST_CNT]:   C_ERR_TBL [.ICTLR, C_ERR_HRD] = .C_ERR_TBL [.ICTLR, C_ERR_HRD] + 1;
;  5354                                                                                                     ! HARD - CONTROLLER
;  5355
;  5356                         [ST_DRV]:   if .ERROR_SUB eql 3                                             ! HARD - DRIVE
;  5357                                     then
;  5358                                         TALLY_ADDR [ERR_HRD_SEK] = .TALLY_ADDR [ERR_HRD_SEK] + 1
;  5359                                     else
;  5360                                         TALLY_ADDR [ERR_HRD_DRV] = .TALLY_ADDR [ERR_HRD_DRV] + 1;
;  5361                         tes;
;  5362
;  5363                 end
;  5364
;  5365             else                                                                                    ! UNIT NOT FOUND
;  5366
;  5367                 if .SOFT_OCCURED
;  5368                 then
;  5369                     C_ERR_TBL [.ICTLR, C_ERR_SFT] = .C_ERR_TBL [.ICTLR, C_ERR_SFT] + 1
;  5370                 else
;  5371                     C_ERR_TBL [.ICTLR, C_ERR_HRD] = .C_ERR_TBL [.ICTLR, C_ERR_HRD] + 1;
;  5372
;  5373             end;                                                    ! ROUTINE SOFT_ERROR

                                    .SBTTL  SOFT.ERROR RDRX INTERRUPT SERVICE ROUTINES
```

```
02424                                          .PSECT  $CODE$,  RO

000000  004137  000000G              SOFT.ERROR:
                                             JSR     R1,$SAVE5                ;                                          5 65
000004  024646                               CMP     (SP), (SP)
000006  105001                               CLRB    R1                       ; FOUND                                    5275
000010  105066  000002                       CLRB    2(SP)                    ; SOFT.OCCURED
000014  016646  000022                       MOV     22(SP),-(SP)             ; INDEX,*                                  5286
000020  012746  000102                       MOV     #102, (SP)
000024  004737  000000G                      JSR     PC,BL$MUL
000030  062700  000000G                      ADD     #ELOG.PKT,RO
000034  010004                               MOV     RO,R4                    ; *,ELOG.ADDR
000036  116400  000020                       MOVB    20(R4),RO                ; *(ELOG.ADDR),*                           5287
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                   5288
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                                 5290
000100  010005              1$:            MOV     RO,R5                    ; OFFSET,*                                 5292
000102  063705  000106'                     ADD     ICST.ADDR,R5
000106  032715  040000                       BIT     #40000,(R5)
000112  001420                               BEQ     2$
000114  016446  000012                       MOV     12(R4),-(SP)             ; *(ELOG.ADDR),*                           5293
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                                  5296
000136  011546                               MOV     (R5),-(SP)               ;                                          5297
000140  000316                               SWAB    (SP)
000142  042716  177760                       BIC     #177760,(SP)
000146  012666  000004                       MOV     (SP)+,4(SP)              ; *,UNIT
000152  000405                               BR      3$                       ;                                          5295
000154  062700  000012              2$:      ADD     #12,RO                   ; *,OFFSET                                 5290
000160  020027  000044                       CMP     RO,#44                   ; OFFSET,*
000164  003745                               BLE     1$
000166  112766  000001  000006      3$:      MOVB    #1,6(SP)                 ; *,SOFT.OCCURED                           5304
000174  006001                               ROR     R1                       ; FOUND                                    5306
000176  1031[illegible]                       BCC     17$
000200  0166.6  000004                       MOV     4(SP),(SP)               ; UNIT,*                                   5309
000204  012746  000070                       MOV     #70,-(SP)
000210  004737  000000G                      JSR     PC,BL$MUL
000214  062700  000000G                      ADD     #TALLY,RO
000220  032766  000001  000010              BIT     #1,10(SP)                ; *,SOFT.OCCURED                           5311
000226  001455                               BEQ     9$
000230  120227  000005                       CMPB    R2,#5                    ; ERROR.CODE,*                             5313
000234  001444                               BEQ     7$                       ;                                          5316
000236  120227  000010                       CMPB    R2,#10                   ; ERROR.CODE,*                             5313
000242  001014                               BNE     4$
000244  012701  000064                       MOV     #64,R1                   ;                                          5320
```

```
000250  060001                          ADD     R0,R1                   ; TALLY.ADDR,*
000252  020327  000002                   CMP     R3,#2                   ; ERROR.SUB,*                   5318
000256  001065                          BNE     10$
000260  005004                          CLR     R4                      ;                               5320
000262  156104  000001                   BISB    1(R1),R4
000266  005204                          INC     R4
000270  110411                          MOVB    R4,(R1)
000272  000514                          BR      16$                     ;                               5318
000274  120227  000011          4$:     CMPB    R2,#11                  ; ERROR.CODE,*                  5313
000300  001003                          BNE     5$
000302  105260  000067                   INCB    67(R0)                  ; *(TALLY.ADDR)                 5324
000306  000506                          BR      16$                     ;                               5313
000310  120227  000012          5$:     CMPB    R2,#12                  ; ERROR.CODE,*
000314  001006                          BNE     6$
000316  013701  000156'                  MOV     ICTLR,R1                ;                               5326
000322  006301                          ASL     R1
000324  105261  000001G                  INCB    C.ERR.TBL+1(R1)
000330  000475                          BR      16$                     ;                               5313
000332  120227  000013          6$:     CMPB    R2,#13                  ; ERROR.CODE,*
000336  001072                          BNE     16$
000340  020327  000003                   CMP     R3,#3                   ; ERROR.SUB,*                   5329
000344  001003                          BNE     8$
000346  105260  000064          7$:     INCB    64(R0)                  ; *(TALLY.ADDR)                 5331
000352  000464                          BR      16$                     ;                               5329
000354  105260  000066          8$:     INCB    66(R0)                  ; *(TALLY.ADDR)                 5333
000360  000461                          BR      16$                     ;                               5313
000362  005764  000006          9$:     TST     6(R4)                   ; *(ELOG.ADDR)                  5337
000366  001056                          BNE     16$
000370  005764  000010                   TST     10(R4)                  ; *(ELOG.ADDR)                  5338
000374  001053                          BNE     16$
000376  120227  000005                   CMPB    R2,#5                   ; ERROR.CODE,*                  5340
000402  001443                          BEQ     14$                     ;                               5343
000404  120227  000010                   CMPB    R2,#10                  ; ERROR.CODE,*                  5340
000410  001013                          BNE     11$
000412  012701  000060                   MOV     #60,R1                  ;                               5347
000416  060001                          ADD     R0,R1                   ; TALLY.ADDR,*
000420  020327  000002                   CMP     R3,#2                   ; ERROR.SUB,*                   5345
000424  001002                          BNE     10$
000426  105211                          INCB    (R1)                    ;                               5347
000430  000435                          BR      16$                     ;                               5345
000432  105261  000001          10$:    INCB    1(R1)                   ;                               5349
000436  000432                          BR      16$                     ;                               5340
000440  120227  000011          11$:    CMPB    R2,#11                  ; ERROR.CODE,*
000444  001003                          BNE     12$
000446  105260  000063                   INCB    63(R0)                  ; *(TALLY.ADDR)                 5351
000452  000424                          BR      16$                     ;                               5340
000454  120227  000012          12$:    CMPB    R2,#12                  ; ERROR.CODE,*
000460  001006                          BNE     13$
000462  013701  000156'                  MOV     ICTLR,R1                ;                               5353
000466  006301                          ASL     R1
000470  105261  000000G                  INCB    C.ERR.TBL(R1)
000474  000413                          BR      16$                     ;                               5340
000476  120227  000013          13$:    CMPB    R2,#13                  ; ERROR.CODE,*
000502  001010                          BNE     16$
000504  020327  000003                   CMP     R3,#3                   ; ERROR.SUB,*                   5356
000510  001003                          BNE     15$
```

```
000512  105260  000060                  14$:    INCB    60(R0)                  ; *(TALLY.ADDR)                         5358
000516  000402                          16$:    BR      16$                     ;                                       5356
000520  105260  000062                  15$:    INCB    62(R0)                  ; *(TALLY.ADDR)                         5360
000524  005726                          16$:    TST     (SP)+                   ;                                       5308
000526  000415                          19$:    BR      19$                     ;                                       5306
000530  013700  000156'                17$:    MOV     ICTLR,R0                ;                                       5369
000534  006300                          ASL     R0
000536  062700  000000G                 ADD     #C.ERR.TBL,R0
000542  032766  000001  000006          BIT     #1,6(SP)                ; *,SOFT.OCCURED                        5367
000550  001403                          BEQ     18$
000552  105260  000001                  INCB    1(R0)                   ;                                       5369
000556  000401                          BR      19$                     ;                                       5367
000560  105210                          18$:    INCB    (R0)                    ;                               5371
000562  062706  000010                  19$:    ADD     #10,SP                  ;                               5265
000566  000207                          RTS     PC

; Routine Size:  188 words,    Routine Base:  $CODE$ + 24242
; Maximum stack depth per invocation:  12 words

;         5374
;         5375  end
;         5376
;         5377  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
;         $CODE$                         5389       RO ,  I  ,  LCL,  REL,  CON
;         $PLIT$                           10       RO ,  D  ,  LCL,  REL,  CON


;                                   LIBRARY STATISTICS
;
;                                           -------- Symbols --------     Blocks
;       File                                Total    Loaded   Percent       Read
;
;  SPIDER$USERS:[DOUCETTE.FALCON]CNRQAA.L16;7
;                                             457       360        78        119

;                                       COMMAND QUALIFIERS

;       BLISS /PDP11 CNRQAA.BL2/LIST=CNRQAA.LI2/OBJECT=CNRQAA.O82/SOURCE=PAGE:56
```

```
         5378  module NRQAM4 (
         5379
         5380  %title 'RD/RX EXERCISER'
         5381                  ident = 'V01.2',
         5382                  addressing mode (absolute),
         5383                  environment (noeis)
         5384                  ) =
         5385
         5386  begin
         5387
         5388  %ebttl 'LASTAD AND SETUP'
         5389
         5390  library 'CNRQAA.L16';
         5391
         5392  require 'BLSMAC.REQ';                          ! DIAGNOSTIC SUPERVISOR LIBRARY
         6881
         6882  LASTAD
         6883
         6884  BGNSETUP (3)
         6885
       P 6886          BGNPTAB
       P 6887                  INIT_IP_ADDR, INIT_INTR_VECT, INIT_BR_LEVEL, %o'100034', 0, RD_MAX_LBN
       P 6888                          ! IP, VECTOR, BR, DISK ADDR, START BLOCK, END BLOCK
         6889          ENDPTAB
         6890
       P 6891          BGNPTAB
       P 6892                  INIT_IP_ADDR, INIT_INTR_VECT, INIT_BR_LEVEL, %o'100001', 0, RX_MAX_LBN
       P 6893                          ! IP, VECTOR, BR, DISK ADDR, START BLOCK, END BLOCK
         6894          ENDPTAB
         6895
       P 6896          BGNPTAB
       P 6897                  INIT_IP_ADDR, INIT_INTR_VECT, INIT_BR_LEVEL, %o'100002', 0, RX_MAX_LBN
       P 6898                          ! IP, VECTOR, BR, DISK ADDR, START BLOCK, END BLOCK
         6899          ENDPTAB
         6900
         6901  ENDSETUP


                                                .TITLE  NRQAM4 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$LAST+24
000006  000006                                .WORD   6                               ; Plit count word
000010  176150                        P.AAB:  .WORD   -1630
000012  000154                                .WORD   154
000014  000004                                .WORD   4
000016  100034                                .WORD   -77744
000020  000000                                .WORD   0
000022  052137                                .WORD   52137
000024  000054'                        P.AAC:  .WORD   L$LAST+50
000026  000006                                .WORD   6                               ; Plit count word
```

```
000030  176150                          P.AAD:  .WORD   1630
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  176150                          P.AAF:  .WORD   1630
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                              ;                                       6880
; Routine Size:  1 word,        Routine Base:  $XYZ$ + 0066
; Maximum stack depth per invocation:  0 words

;         6902  end
;         6903
;         6904  eludom
;                                       PSECT SUMMARY
;
;       Psect Name                      Words     Attributes
;        $XYZ$                            28       RO ,  I  ,  LCL,  REL,  CON

;                                   LIBRARY STATISTICS
;
;                                            -------- Symbols -------      Blocks
;       File                                 Total    Loaded   Percent       Read
;
;  SPIDER$USERS:[DOUCETTE.FALCON]CNRQAA.L16;7
;                                             457         7         1         17

;                                       COMMAND QUALIFIERS
;       BLISS /PDP11 CNRQAA.BL2/LIST=CNRQAA.LI2/OBJECT=CNRQAA.OB2/SOURCE=PAGE:56

; Size:         5390 code + 380 data words
; Run Time:       03:11.7
; Elapsed Time:   10:10.4
; Memory Used:  411 pages
; Compilation Complete
```

```
0001  !                       REVISION HISTORY
0002  !
0003  !
0004  !       The following changes were made to CZRQAB in producing CNRQAA for the
0005  !       FALCON-PLUS project (SBC 11/21+).  Release date December 19, 1984.
0006  !       Changes made by James S. Doucette.  All changes are marked by
0007  !       "!JSD REV A'.
0008  !
0009  !       1. Lowered the general operating priority of the program from level 7
0010  !              to level 6 to allow the BREAK key to interrupt and invoke
0011  !              ODT.
0012  !
0013  !       2. Set the ODT BREAK vector (location 140) to the starting address
0014  !              of FALCON's ODT ROM (170000-octal).
0015  !
0016  !       3. Due to space limitations, removed all references to DBM's (debug
0017  !              messages).
0018  !
0019  !       4. Changed the default IP address from 172150 to 176150.
0020  !
0021  !---- ---- ---------------------------------------- ----------------------
0022  !
0023  !*****************************************************************************
0024  !                                                                            *
0025  !                       L I T E R A L S                                      *
0026  !                                                                            *
0027  !*****************************************************************************
0028  !
0029  LITERAL
0030  !
0031  !****** ODT TRAP VECTOR LOCATION
0032  !
0033  !       O_TVEC          = %o'14',               !JSD REV A
0034          O_TVEC          = %o'140',              !JSD REV A
0035          O_BRK           = %o'170000',           !JSD REV A - ODT START ADDRESS
0036  !
0037  !****** HARDWARE ADDRESSES ETC.
0038  !
0039          INIT_INTR_VECT  = %o'154',              ! VECTOR ADDRESS
0040  !       INIT_IP_ADDR    = %o'172150',           ! IP REGISTER ADDRESS   !JSD REV A
0041          INIT_IP_ADDR    = %o'176150',           ! IP REGISTER ADDRESS   !JSD REV A
0042          INIT_BR_LEVEL   = %o'4',                ! BUS REQUEST LEVEL
0043  !
0044  !****** HARDWARE LIMITS
0045  !
0046   !      dupround        = 10,                   ! number of dbn's read per dup exerciser pass.
0047
0048          MAX_CTLR        = 1,                    ! MAXIMUM NUMBER OF LCP CONTROLLERS ALLOWED
0049          MAX_DRIVES      = 4,
0050          MAX_UNITS       = MAX_CTLR * max_drives, ! MAXIMUM NUMBER OF UNITS TO TEST
0051
0052          RD_MAX_TRACK    = 1200,                 ! MAXIMUM NUMBER OF TRACKS FOR RD51
0053          RD_SEC_PER_TRK  = 18,                   ! NUMBER OF HOST SECTORS PER TRACK FOR RD51
0054          RD_MAX_SECT     = RD_SEC_PER_TRK - 1,   ! LARGEST SECTOR NUMBER IN RD51
0055          RD_MAX_LBN      = RD_MAX_TRACK * RD_SEC_PER_TRK - 1, !MAX LBN FOR RD
0056
```

```
0057          RX_MAX_TRACK    = 80,                   ! MAXIMUM NUMBER OF TRACKS FOR RX50
0058          RX_SEC_PER_TRK  = 10,                   ! NUMBER OF HOST SECTORS PER TRACK FOR RX50
0059          RX_MAX_SECT     = RX_SEC_PER_TRK - 1,   ! LARGEST SECTOR NUMBER IN RX50
0060          RX_MAX_LBN      = RX_MAX_TRACK * RX_SEC_PER_TRK - 1, !MAX LBN FOR RX
0061
0062          MAX_TRACK       = RD_MAX_TRACK,         ! MAX_TRACK IS THE MAX OF RD_- AND RX_MAX_TRACK
0063                                                  ! IT TURNS OUT TO BE RD_... BECAUSE RD IS A WINCHESTER
0064                                                  ! AS OPPOSED TO A FLOPPY, AND HAS MORE TRACKS
0065          MAX_SECT        = RD_SEC_PER_TRK,       ! MAXIMUM NUMBER OF SECTORS
0066
0067                                                  ! NUMBER IS INCREMENTAL DIFFERENTIAL FOR ADDITIONAL RX UNITS
0068          RX_SEQ_CNT      = (RX_MAX_LBN + 1) / 10, ! NUMBER OF OPERATIONS TO DO TO RX50 IN A ROW FOR SEQUENTIAL MODE
0069          RD_SEQ_CNT      = (RD_MAX_LBN + 1) / 10, ! NUMBER OF OPERATIONS TO DO TO RD51 IN A ROW FOR SEQUENTIAL MODE
0070
0071          BYTES_PER_SECT  = 512,                  ! BYTES PER SECTOR (AT PRESENT SAME FOR RDs AND RXs)
0072          MAX_XFER_SIZE   = 2 * BYTES_PER_SECT,   ! ARTBITRARY MAX SIZE OF EACH DISK I/O
0073
0074    ! NOTE BOTH OF THESE NUMBERS ARE NOW ARBITRARILY CHOSEN AS THE NUMBER OF LBNS CONTAINED PER UNIT/10 .
0075
0076    !
0077    !****** RING SIZES
0078    !
0079          CR_LOG          = 2,                    ! LOG2 LENGTH OF COMMAND RING
0080          RR_LOG          = 2,                    ! LOG2 LENGTH OF RESPONSE RING
0081          CRING_LEN       = 1 ^ CR_LOG,           ! COMMAND RING LENGTH
0082          RRING_LEN       = 1 ^ RR_LOG,           ! RESPONSE RING LENGTH
0083    !
0084    !****** TABLE AND OTHER STRUCTURE SIZES
0085    !
0086          HWPT_LEN        = 6,                    ! SIZE (WORDS) OF HW P-TABLE
0087          COMM_LEN        = (RRING_LEN * 2) + (CRING_LEN * 2) + 4,
0088                                                  ! SIZE (WORDS) OF COMMUNICATION AREA FOR ONE CONTROLLER
0089          LAST_PKT_LEN    = 3,                    ! LENGTH OF BUFFER TO SAVE INFO. ABOUT LAST RESPONSE PACKET
0090    !****** OFFSETS!^
0091          OF_UN           = 3,                    ! WORD OFFSET FROM START OF CST TO FIRST UNIT
0092
0093          UNIT_SIZE       = 5,                    ! SIZE OF CST UNIT ENTRY
0094          CST_LEN         = OF_UN + 4 * UNIT_SIZE,        ! SIZE (WORDS) OF A CONTROLLER STATUS TABLE
0095
0096          TALLY_CLEAR     = 7,                    ! SIZE (WORDS) OF STATISTICS TABLE CLEARED ON EVERY PASS!^
0097          TALLY_TOTALS    = 21,                   ! SIZE (WORDS) OF STATISTICS TABLE FOR STORING TOTALS!^
0098          TALLY_LEN       = TALLY_CLEAR + TALLY_TOTALS,           ! SIZE (WORDS) OF A STATISTICS TABLE
0099          C_ERR_LEN       = 1,                    ! SIZE (WORDS) OF CONTROLLER ERROR TABLE
0100          RP_LEN          = 24,                   ! SIZE (WORDS) OF A RETURN PACKET
0101          IODQ_LEN        = MAX_CTLR * (RRING_LEN / 2) + 2,       ! NUMBER OF ENTRIES IN I/O DONE QUEUE (IODQ)
0102          MSG_LEN         = 30,                   ! SIZE (WORDS) OF AN MSCP MESSAGE (TEXT PORTION)
0103          pkt_LEN         = MSG_LEN + 4,          ! SIZE (WORDS) OF AN MSCP ENVELOPE
0104          DCT_LEN         = 9,                    ! SIZE (WORDS) OF A DRIVER CONTROLLER TABLE
0105          RDM_LEN         = 16,                   ! SIZE (WORDS) OF THE RANDOM NUMBER TABLE
0106          RP_CNT          = MAX_CTLR * RRING_LEN, ! NUMBER OF RETURN PACKETS IN POOL
0107          MAX_UDP_CNT     = 16,                   ! MAX SIZE OF USER DATA PATTERN
0108          MAX_BUF_CNT     = (CRING_LEN * 2) * MAX_CTLR,
0109                                                  ! MAXIMUM NUMBER OF I/O BUFFERS (SIZE OF BUFF_ADDR AND BUFF_OWN)
0110          pkt_CNT         = ((CRING_LEN * 2) + RRING_LEN) * MAX_CTLR,
0111                                                  ! NUMBER OF MSCP ENVELOPES IN POOL
0112          OUTC_CNT        = CRING_LEN * 2,        ! NUMBER OF ENTRIES IN A CONTROLLER'S OUTSTANDING CMD LIST
```

```
;       0113          DP_CNT          = 21,                   ! NUMBER OF PRE-DEFINED DATA PATTERNS
;       0114          EP_CNT          = MAX_CTLR * RRING_LEN * 3,
;       0115                                                  ! NUMBER OF ERROR LOG PACKET SAVE BUFFERS
;       0116          EP_LEN          = PKT_LEN  2 + 1,       ! LENGTH OF EACH ERROR-LOG SAVE BUFFER
;       0117
;       0118  !
;       0119  !****** SW P-TABLE FLAGS (SWP_FLAGS)
;       0120  !
;       0121          SWF_TRC         = %o'1',                ! DIAGNOSTIC TRACE
;       0122          SWF_RDM         = %o'2',                ! RANDOM SEEK MODE
;       0123          SWF_CRC         = %o'4',                ! READ-COMPARE AT CONTROLLER
;       0124          SWF_DCC         = %o'10',               ! DRIVE COMPLEMENT COMPLETE
;       0125          SWF_CWC         = %o'20',               ! WRITE-COMPARE AT CONTROLLER
;       0126          SWF_HWC         = %o'40',               ! WRITE-COMPARE AT HOST
;       0127          SWF_UDP         = %o'100',              ! USER-DEFINED DATA PATTERN
;       0128          SWF_CST         = %o'200',              ! CLEAR STATISTICAL TABLES
;       0129          SWF_DIA         = %o'400',              ! DIAGNOSTIC PACKAGE, WHEN THIS IS SELECTED
;       0130                                                  ! ALL INTERRUPTS ARE WAITED FOR, E.G. ONLY
;       0131                                                  ! ONE MSCP PACKET IS OUTSTANDING AT A TIME
;       0132          SWF_SEQ         = %o'1000',             ! RANDOM OR FIXED SEQUENTIAL STEPPING
;       0133  !       SWF_DUP         = %o'2000',             ! RUN DUP DIAGNOSTIC
;       0134          SWF_FER         = %o'4000',             ! REWRITE BLOCKS WHEN "FORCED ERROR" BIT DETECTED
;       0135          SWF_HOE         = %o'10000',            ! TREAT ALL ERRORS ALIKE FOR 'HOE' DRS FLAG
;       0136
;       0137  !
;       0138  ! DUP FLAGS FOR DUP EXERCISER (DUP_FLAGS)
;       0139  !
;       0140          SWF_DUP         = %o'1',                ! RUN DUP DIAGNOSTIC
;       0141          SWP_DINT        = %o'2',                ! DUP CAUSED REININITIALIZATION
;       0142  !
;       0143  !****** ENTRY_REASON VALUES
;       0144  !               (HOW PROGRAM WAS INVOKED)
;       0145  !
;       0146          START           = 1,                    ! START
;       0147          RESTART         = 2,                    ! RESTART
;       0148          CONT            = 3,                    ! CONTINUE
;       0149          PWR_FAIL        = 4,                    ! POWER FAIL
;       0150          NEW_PASS        = 5,                    ! NEW PASS
;       0151  !
;       0152  !****** DROP UNIT REASONS
;       0153  !               (LOADED INTO DUR VECTOR)
;       0154  !
;       0155          DU_USER         = 0,                    ! USER COMMAND
;       0156          DU_CONF         = 1,                    ! CONFIGURATION ERROR
;       0157          DU_INIT         = 2,                    ! INITIALIZATION ERROR
;       0158          DU_XFER         = 3,                    ! TRANSFER LIMIT REACHED
;       0159          DU_HERR         = 4,                    ! HARD ERROR LIMIT REACHED
;       0160          DU_DFATAL       = 5,                    ! UNRECOVERABLE DEVICE ERROR
;       0161          DU_CFATAL       = 6,                    ! UNRECOVERABLE CONTROLLER ERROR
;       0162          DU_ONLINE       = 7,                    ! ONLINE FAILED
;       0163          DU_ACCESS       = 8,                    ! ACCESS TO LAST TRACK FAILED
;       0164          DU_PROTECT      = 9,                    ! WRITE PROTECT CONFLICT
;       0165          DU_TIME         = 10,                   ! COMMAND TIME OUT
;       0166          DU_AV           = 11,                   ! DEVICE WENT TO AVAILABLE STATE
;       0167  !
;       0168  !****** MISCELLANEOUS LITERALS
```

```
;       0169  !
;       0170          INI_ATT         = 2,                    ! NUMBER OF HW INIT ATTEMPTS BEFORE FAILURE IS ASSUMED
;       0171          WR_RING         = ((%o'200') or (CR_LOG ! 3) or (RR_LOG)),
;       0172                                                  ! WR-BIT-AND-RING-LENGTH BYTE (USED IN STEP 1 WRITE AND STEP 2 READ)
;       0173          QIO_PER_CTLR    = CRING_LEN * 2,        ! MAXIMUM NUMBER OF OUTSTANDING QIOS PER CONTROLLER
;       0174          MAX_XFER        = 256,                  ! MAXIMUM SIZE (WORDS) OF AN I/O TRANSFER
;       0175          RX50_BIT        = %o'0',                ! BIT IN HARDWARE TABLES MARKING AN RX50
;       0176          RD51_BIT        = %o'4',                ! BIT IN HARDWARE TABLES MARKING AN RD51
;       0177          RX_50           = 0,                    ! NUMBER FOR RX_50 WHEN SHIFTED RIGHT
;       0178          RD_51           = 1,                    ! NUMBER FOR RX_50 WHEN SHIFTED RIGHT
;       0179  !
;       0180  !****** MSCP ENVELOPE DESCRIPTOR
;       0181  !
;       0182          ED_OWN          = %o'100000',           ! OWNERSHIP BIT
;       0183          ED_FLAG         = %o'040000',           ! FLAG BIT
;       0184  !
;       0185  !****** MSCP COMMAND PACKET OPCODES
;       0186  !
;       0187          OP_MSK          = %o'177',              ! OPCODE MASK
;       0188          OP_END          = %o'200',              ! ENDCODE DESIGNATOR
;       0189          OP_ACC          = %o'20',               ! ACCESS COMMAND
;       0190          OP_ONL          = %o'11',               ! ONLINE COMMAND
;       0191          OP_RD           = %o'41',               ! READ COMMAND
;       0192          OP_SCC          = %o'4',                ! SET CONTROLLER CHARACTERISTICS COMMAND
;       0193          OP_WRT          = %o'42',               ! WRITE COMMAND
;       0194          OP_GDS          = %o'1',                ! GET DUST STATUS!†
;       0195          OP_ESP          = %o'2',                ! EXECUTE SUPPLIED PROGRAM
;       0196          OP_ELP          = %o'3',                ! EXECUTE LOCAL PROGRAM
;       0197          OP_SDD          = %o'4',                ! SEND DATA
;       0198          OP_RCD          = %o'5',                ! RECEIVE DATA
;       0199          OP_ABP          = %o'6',                ! ABORT PROGRAM
;       0200
;       0201  !
;       0202  ! PACKET SIZES
;       0203  !
;       0204          SZ_ACC = %decimal '32',                 !ACCESS
;       0205          SZ_ONL = %decimal '36',                 !ON LINE COMMAND
;       0206          SZ_RD  = %decimal '32',                 !READ
;       0207          SZ_SCC = %decimal '32',                 !SET CONTROLLER CHARACTERISTICS
;       0208          SZ_WRT = %decimal '32',                 !WRITE
;       0209          SZ_GEN = %decimal '32',                 !GENERAL PACKET SIZE
;       0210          SZ_REC = %DECIMAL '28',
;       0211          SZ_SEN = %DECIMAL '28',
;       0212          SZ_ELP = %DECIMAL '18',
;       0213          SZ_ABT = %DECIMAL '12',
;       0214          SZ_GDS = %DECIMAL '12',
;       0215  !
;       0216  !****** MSCP COMMAND MODIFIERS
;       0217  !
;       0218          MD_CMP          = %o'40000',            ! COMPARE
;       0219  !
;       0220  !****** CONNECTION ID VALUES (MSCP_pkt, RETPKT)
;       0221  !               (SERVE AS SOURCES AND DESTINATIONS OF MSCP MESSAGES)
;       0222  !
;       0223          CID_MSCP        = 0,                    ! DISK MSCP
;       0224          CID_TAPE        = 1,                    ! TAPE MSCP
```

```
0225            CID_DUP         = 2,                    ! DIAGNOSTIC AND UTILITIES PROTOCOL
0226            CID_DRIVER      = 3,                    ! EXERCISER "DRIVER"
0227    !
0228    !****** MESSAGE TYPE VALUES
0229    !
0230            MT_SEQ          = 0,                    ! SEQUENTIAL (FROM PORT)
0231            MT_DG           = 1,                    ! DATAGRAM (FROM PORT)
0232            MT_CRD          = 2,                    ! CREDIT NOTIFICATION (FROM PORT)
0233            MT_FATAL        = 3,                    ! FATAL DEVICE ERROR (FROM "DRIVER")
0234            MT_TIMEOUT      = 4,                    ! COMMAND TIMEOUT (FROM "DRIVER")
0235            MT_BLKDATX      = 5,                    ! BLOCK DATA TRANSFER (DUP FROM PORT)!+
0236    !
0237    !****** CONTROLLER FLAGS
0238    !               (IN SET CONTROLLER CHARACTERISTICS COMMAND AND RESPONSE)
0239    !
0240            CF_ATN          = %o'000200',           ! ENABLE ATTENTION MESSAGES
0241            CF_MSC          = %o'000100',           ! ENABLE MISCELLANEOUS ERROR LOG MESSAGES
0242            CF_OTH          = %o'000040',           ! ENABLE OTHER HOST'S ERROR LOG MESSAGES
0243            CF_THS          = %o'000020',           ! ENABLE THIS HOST'S ERROR LOG MESSAGES
0244    !       CF_MASK         = CF_ATN + CF_MSC + CF_THS,
0245            CF_MASK         = CF_MSC or CF_THS,     ! RELEVENT BITS IN CTLR FLAGS WORD
0246    !
0247    !****** UNIT FLAGS
0248    !               (IN ONLINE COMMAND AND RESPONSE)
0249    !
0250            UF_WPH          = %o'020000',           ! WRITE PROTECT (HARDWARE)
0251            UF_WPS          = %o'010000',           ! WRITE PROTECT (softWARE)
0252    !
0253    !****** STATUS / EVENT CODE DEFINITIONS
0254    !
0255            ST_SUC          = %o'0',                ! SUCCESS
0256            ST_CMD          = %o'1',                ! INVALID COMMAND
0257            ST_ABO          = %o'2',                ! COMMAND ABORTED
0258            ST_OFL          = %o'3',                ! UNIT OFFLINE
0259            ST_AVL          = %o'4',                ! DRIVE AVAILABLE
0260            ST_MFE          = %o'5',                ! MEDIA FORMAT ERROR
0261            ST_WPT          = %o'6',                ! WRITE PROTECTED
0262            ST_CMP          = %o'7',                ! COMPARE ERROR
0263            ST_DAT          = %o'10',               ! DATA ERROR
0264            ST_HST          = %o'11',               ! HOST BUFFER ACCESS ERROR
0265            ST_CNT          = %o'12',               ! CONTROLLER ERROR
0266            ST_DRV          = %o'13',               ! DRIVE ERROR
0267            ST_DIA          = %o'37',               ! MESSAGE FROM INTERNAL DIAGNOSTICS
0268    !
0269    !****** END MESSAGE FLAGS
0270    !
0271            EF_BBR          = %o'200',              ! BAD BLOCK REPORTED
0272            EF_0            = %o'200',              ! BAD BLOCK REPORTED
0273            EF_1            = %o'100',              !
0274            EF_2            = %o'40',               !
0275            EF_3            = %o'20',               !
0276            EF_4            = %o'200',              !
0277            EF_5            = %o'100',              !
0278            EF_6            = %o'40',               !
0279            EF_7            = %o'20',               !
0280            EF_8            = %o'100000',           !
```

```
0281          EF_9            = %o'2',            !
0282          EF_10           = %o'1',            !
0283          EF_11           = %o'1',            !
0284          EF_12           = %o'2',            !
0285          EF_13           = %o'100000',       !
0286          EF_14           = %o'40000',        !
0287          EF_15           = %o'200',          !
0288          EF_16           = %o'4000',         !
0289          EF_17           = %o'2000',         !
0290          EF_18           = %o'100',          !
0291          EF_19           = %o'20000',        !
0292          EF_20           = %o'10000',        !
0293          EF_21           = %o'4',
0294    !
0295    !****** RDRX LITERALS
0296    !
0297          RCIP            = 0,                ! IP REGISTER
0298          RCSA            = 1,                ! SA REGISTER
0299    !
0300    !     COMMON SA REGISTER BIT DEFINITIONS
0301    !
0302          SA_S1           = %o'004000',       ! STEP 1 STATUS BIT
0303          SA_S2           = %o'010000',       !    :  2
0304          SA_S3           = %o'020000',       !    :  3
0305          SA_S4           = %o'040000',       !    V  4
0306          SA_ERR          = %o'100000',       ! ERROR INDICATOR
0307          SA_INT          = %o'000200',       ! INTERRUPT ENABLE DURING INITIALIZATION
0308          SA_GO           = %o'000001',       ! GO BIT TO START FIRMWARE
0309    !
0310    !     INITIALIZATION STEP READ MASKS
0311    !
0312          S1_MASK         = %o'176000',       ! STEP 1 READ BITS
0313          S2_MASK         = %o'174377',       !    :  2
0314          S3_MASK         = %o'174377',       !    :  3
0315          S4_MASK         = %o'174000',       !    V  4
0316    !
0317    !****** COMMAND TYPES
0318    !
0319          IMM_CMD         = 0,                ! IMMEDIATE COMMAND
0320          SEQ_CMD         = 1,                ! SEQUENTIAL COMMAND
0321          NON_SEQ_CMD     = 2,                ! NON-SEQUENTIAL COMMAND
0322    !
0323    !****** ERROR-LOG FORMAT TYPES
0324    !
0325          FORMAT_CNTR     = %o'0',            ! CONTROLLER ERROR
0326          FORMAT_HOST     = %o'1',            ! HOST MEMORY ACCESS ERROR
0327          FORMAT_XFER     = %o'2',            ! DISK TRANSFER ERROR
0328          FORMAT_SDI      = %o'3',            ! 'STANDARD DISK INTECONNECT  ERROR
0329          FORMAT_SDE      = %o'4',            ! SMALL DISK ERROR
0330    !
0331    !****** ERROR-LOG BLOCK NUMBER INFORMATION
0332    !
0333          TYPE_LBN        = %o'0000',         ! LOGICAL BLOCK NUMBER
0334          TYPE_RBN        = %o'0110',         ! REPLACEMENT BLOCK NUMBER
0335
0336    !
```

```
0357  !****** LITERAL FOR READABILITY
0358  !
0359          YES             = 1,
0360          NO              = 0,
0361          TRUE            = 1,
0362          FALSE           = 0,
0363          SUCCESS         = 1,
0364          FAILURE         = 0,
0365          FOUND           = 1,
0366          NOT_FOUND       = 0,
0367          ACTIVE          = 1,
0368          IDLE            = 0,
0369          PRESENT         = 1,        !DISK IS PRESENT IN CONTROLLER
0370          NOT_PRESENT     = 0,        !DISK IS NOT PRESENT IN CONTROLLER
0371          UNPROTECTED     = 1,        !DISK HAS UNPROTECTED CUSTOMER LBN'S
0372          PROTECTED       = 0,        !DISK HAS PROTECTED CUSTOMER LBN'S
0373          ONLINE          = 1,
0374          OFFLINE         = 0,
0375          FULL            = 1,        ! ERROR-LOG SAVE PACKET FILLED
0376          EMPTY           = 0,        ! ERROR-LOG SAVE PACKET PRINTED
0377          HRD_OCCURED     = 1,        ! HARD ERROR DETECTED IN RESPONSE PACKET
0378          HRD_NOT_OCCURED = 0,        ! HARD ERROR NOT DETECTED
0379          ALL_ONES        = %o'177777' ;
```

```
!************************************************************************
!                                                                       *
!                        F I E L D S                                    *
!                                                                       *
!************************************************************************
!
FIELD
!
!****** HARDWARE P TABLE FIELDS
!
HWP_FIELDS =
        set
        HWP_IP_ADDR     = [0, 0, 16, 0],        ! IP ADDRESS
        HWP_VECTOR      = [1, 0, 16, 0],        ! VECTOR ADDRESS
        HWP_BR_LEVEL    = [2, 0, 16, 0],        ! BUS REQUEST LEVEL
        HWP_DISK        = [3, 0, 16, 0],        ! DISK (ALL FIELDS)
        HWP_disk_num    = [3, 0, 2, 0],         ! DISK UNIT NUMBER
        HWP_DISK_TYPE   = [3, 2, 1, 0],         ! DISK TYPE
        HWP_DUPEX       = [3, 3, 1, 0],         ! RUN DUP EXERCISER
        HWP_DUPWT       = [3, 4, 1, 0],         ! WRITE FLAG FOR DUP
        HWP_DISK_CP     = [3, 15, 1, 0],        ! PROTECT CUSTOMER DATA BIT
        HWP_BEG_TRK     = [4, 0, 16, 0],        ! BEGINNING TRACK
        HWP_END_TRK     = [5, 0, 16, 0]         ! ENDING TRACK
        tes,
!
!****** COMMUNICATION AREA HEADER FIELDS
!
CUM FIELDS =
        set
        ADAP_CH         = [1, 8, 8, 0],         ! ADAPTER CHANNEL NUMBER FOR PURGES
        CMD_INT         = [2, 0, 16, 0],        ! COMMAND RING INTERRUPT
        RSP_INT         = [3, 0, 16, 0]         ! RESPONSE RING INTERRUPT
        tes,
!
!****** DUP BUFFER FIELD
!
DP FIELDS =
        SET
        DUPBF0          = [0, 0, 16, 0],
        DUPBF1          = [1, 0, 16, 0],
        DUPBF2          = [2, 0, 16, 0],
        DUPTYPE         = [0, 12, 4, 0],
        DUPMESG         = [0, 0, 12, 0]
        TES,
!
!****** CONTROLLER STATUS TABLE (CST) FIELDS
!
Cst_FIELDS =
        set
        IP_ADDR         = [0, 0, 16, 0],        ! IP ADDRESS
        VEC_ADDR        = [1, 0, 9, 0],         ! VECTOR ADDRESS

        STATE           = [1, 15, 1, 0],        ! CONTROLLER STATUS
        BR_LEV          = [2, 0, 8, 0],         ! BUS REQUEST LEVEL
        U_CNT           = [2, 8, 8, 0],         ! NUMBER OF UNITS (DISKS) FOR THIS CONTROLLER
!
```

```
0416          DO_ALL          = [3, 0, 16, 0],    ! DISK 0 (ALL FIELDS)
0417          DO_disk_num     = [3, 0, 2, 0],     ! DISK UNIT NUMBER
0418          DO_TYPE         = [3, 2, 1, 0],     ! DISK TYPE
0419          DO_UNIT         = [3, 8, 5, 0],     ! DISK 0 UNIT NUMBER (DRS UNIT)
0420          DO_FATAL        = [3, 12, 1, 0],    ! DISK 0 FATAL ERROR BIT
0421          DO_STAT         = [3, 13, 1, 0],    ! DISK 0 STATUS BIT
0422          DO_PRES         = [3, 14, 1, 0],    ! DISK 0 PRESENT BIT
0423          DO_PROT         = [3, 15, 1, 0],    ! DISK 0 PROTECT CUSTOMER DATA BIT
0424          DO_BEG          = [4, 0, 16, 0],    ! DISK 0 BEGIN TRACK
0425          DO_END          = [5, 0, 16, 0],    ! DISK 0 END TRACK
0426          DO_DBN          = [6, 0, 8, 0],     ! DISK 0 RELATIVE DBN!!
0427          DO_ACTIVE       = [6, 13, 1, 0],    ! DISK 0 ACTIVE STATE
0428          DO_DUPERROR     = [6, 14, 1, 0],          ! DUP ERROR FLAG!!
0429          DONODUPMED      = [6, 15, 1, 0],          ! NO LOCAL MEDIA OR DUP PROGRAMS FLAG!!
0430          UO_COUNT        = [7, 0, 16, 0],          ! DISK 0 RELATIVE MSCP FUNC COUNTER
0431    !
0432          D1_ALL          = [8, 0, 16, 0],    ! DISK 1 (ALL FIELDS)
0433          D1_disk_num     = [8, 0, 2, 0],     ! DISK UNIT NUMBER
0434          D1_TYPE         = [8, 2, 1, 0],     ! DISK TYPE
0435          D1_UNIT         = [8, 8, 5, 0],     ! DISK 1 UNIT NUMBER (DRS UNIT)
0436          D1_FATAL        = [8, 12, 1, 0],    ! DISK 1 FATAL ERROR BIT
0437          D1_STAT         = [8, 13, 1, 0],    ! DISK 1 STATUS BIT
0438          D1_PRES         = [8, 14, 1, 0],    ! DISK 1 PRESENT BIT
0439          D1_PROT         = [8, 15, 1, 0],    ! DISK 1 PROTECT CUSTOMER DATA BIT
0440          D1_BEG          = [9, 0, 16, 0],    ! DISK 1 BEGIN TRACK
0441          D1_END          = [10, 0, 16, 0],   ! DISK 1 END TRACK
0442          D1_DBN          = [11, 0, 8, 0],    ! DISK 1 RELATIVE DBN
0443          D1_ACTIVE       = [11, 13, 1, 0],   ! DISK 1 ACTIVE STATE
0444          D1_DUPERROR     = [11, 14, 1, 0],         ! DUP ERROR FLAG!!
0445          D1NODUPMED      = [11, 15, 1, 0],         ! NO LOCAL MEDIA OR DUP PROGRAMS FLAG!!
0446          D1_COUNT        = [12, 0, 16, 0],   ! DISK 1 RELATIVE MSCP FUNC COUNTER
0447    !
0448          D2_ALL          = [13, 0, 16, 0],   ! DISK 2 (ALL FIELDS)
0449          D2_disk_num     = [13, 0, 2, 0],    ! DISK UNIT NUMBER
0450          D2_TYPE         = [13, 2, 1, 0],    ! DISK TYPE
0451          D2_UNIT         = [13, 8, 5, 0],    ! DISK 2 UNIT NUMBER (DRS UNIT)
0452          D2_FATAL        = [13, 12, 1, 0],   ! DISK 2 FATAL ERROR BIT
0453          D2_STAT         = [13, 13, 1, 0],   ! DISK 2 STATUS BIT
0454          D2_PRES         = [13, 14, 1, 0],   ! DISK 2 PRESENT BIT
0455          D2_PROT         = [13, 15, 1, 0],   ! DISK 2 PROTECT CUSTOMER DATA BIT
0456          D2_BEG          = [14, 0, 16, 0],   ! DISK 2 BEGIN TRACK
0457          D2_END          = [15, 0, 16, 0],   ! DISK 2 END TRACK
0458          D2_DBN          = [16, 0, 8, 0],    ! DISK 2 RELATIVE DBN
0459          D2_ACTIVE       = [16, 13, 1, 0],   ! DISK 2 ACTIVE STATE
0460          D2_DUPERROR     = [16, 14, 1, 0],         ! DUP ERROR FLAG!!
0461          D2NODUPMED      = [16, 15, 1, 0],         ! NO LOCAL MEDIA OR DUP PROGRAMS FLAG!!
0462          D2_COUNT        = [17, 0, 16, 0],   ! DISK 2 RELATIVE MSCP FUNC COUNTER
0463    !
0464          D3_ALL          = [18, 0, 16, 0],   ! DISK 3 (ALL FIELDS)
0465          D3_disk_num     = [18, 0, 2, 0],    ! DISK UNIT NUMBER
0466          D3_TYPE         = [18, 2, 1, 0],    ! DISK TYPE
0467          D3_UNIT         = [18, 8, 5, 0],    ! DISK 3 UNIT NUMBER (DRS UNIT)
0468          D3_FATAL        = [18, 12, 1, 0],   ! DISK 3 FATAL ERROR BIT
0469          D3_STAT         = [18, 13, 1, 0],   ! DISK 3 STATUS BIT
0470          D3_PRES         = [18, 14, 1, 0],   ! DISK 3 PRESENT BIT
0471          D3_PROT         = [18, 15, 1, 0],   ! DISK 3 PROTECT CUSTOMER DATA BIT
```

```
0472          D3_BEG          = [19, 0, 16, 0],       ! DISK 3 BEGIN TRACK
0473          D3_END          = [20, 0, 16, 0],       ! DISK 3 END TRACK
0474          D3_DBN          = [21, 0, 8, 0],        ! DISK 3 RELATIVE DBN
0475          D3_ACTIVE       = [21, 13, 1, 0],       ! DISK 3 ACTIVE STATE
0476          D3_DUPERROR     = [21, 14, 1, 0],               ! DUP ERROR FLAG!!
0477          D3NODUPMED      = [21, 15, 1, 0],               ! NO LOCAL MEDIA OR DUP PROGRAMS FLAG!!
0478          D3_COUNT        = [22, 0, 16, 0]                ! DISK 3 RELATIVE MSCP FUNC COUNTER
0479          tes.
0480  !
0481  !****** MSCP PACKET FIELDS
0482  !               (NOTE: BASE ADDRESS OF PACKET REFERENCES THE PACKET'S OWN
0483  !               BUFFER DESCRIPTOR, RATHER THAN THE MESSAGE BODY (TEXT + 0).
0484  !               SEE DOCUMENTATION FOR LAYOUT OF MSCP PACKETS.)
0485  !
0486  PKT_FIELDS =
0487          set
0488  !
0489  !     HEADER FIELDS
0490  !
0491          PKT_LO          = [0, 0, 16, 0],        ! PACKET DESCRIPTOR (LO ORDER)
0492          PKT_HI          = [1, 0, 16, 0],        ! PACKET DESCRIPTOR (HI ORDER   ALL FIELDS)
0493          PKT_U           = [1, 0, 2, 0],         ! PACKET DESCRIPTOR (HI ORDER UNIBUS BITS)
0494          PKT_Q           = [1, 2, 4, 0],         ! PACKET DESCRIPTOR (HI ORDER Q-BUS BITS)
0495          PKT_F           = [1, 14, 1, 0],        ! PACKET DESCRIPTOR FLAG BIT
0496          PKT_O           = [1, 15, 1, 0],        ! PACKET DESCRIPTOR OWNERSHIP BIT
0497          CMD_TYPE        = [2, 0, 16, 0],        ! COMMAND TYPE
0498          MSGLEN          = [3, 0, 16, 0],        ! MESSAGE LENGTH
0499          CREDITS         = [4, 0, 4, 0],         ! CREDITS
0500          MSGTYP          = [4, 4, 4, 0],         ! MESSAGE TYPE
0501          CONNID          = [4, 8, 8, 0],         ! CONNECTION ID
0502  !
0503  !     GENERIC COMMAND PACKET AND END PACKET HEADER FILEDS
0504  !
0505          CRN_LO          = [5, 0, 16, 0],        ! COMMAND REF NUMBER (LO ORDER)
0506          CRN_HI          = [6, 0, 16, 0],        ! COMMAND REF NUMBER (HI ORDER)
0507          DK_NUM          = [7, 0, 16, 0],        ! DISK ADDRESS (RD/RX DISK NUMBER)
0508          OPCODE          = [9, 0, 8, 0],         ! OPCODE AND ENDCODE
0509          MODIFY          = [10, 0, 16, 0],       ! COMMAND MODIFIERS
0510          STATUS_CODE     = [10, 0, 5, 0],        ! STATUS (PART OF RESPONSE PACKET)
0511          STATUS_SUBCODE  = [10, 5, 11, 0],       ! SUBCODE (PART OF RESPONSE PACKET)
0512
0513  !
0514  !     READ, WRITE, AND ACCESS COMMAND FIELDS (FOR COMMAND AND END PACKETS)
0515  !
0516          L1              = [11, 0, 8, 0],        ! LETTER NUMBER 1 ! FOR EXECUTE LOCAL PROGRAM COMMAND 0
0517          L2              = [11, 8, 8, 0],        ! LETTER NUMBER 2
0518          L3              = [12, 0, 8, 0],        ! LETTER NUMBER 3
0519          L4              = [12, 8, 8, 0],        ! LETTER NUMBER 4
0520          L5              = [13, 0, 8, 0],        ! LETTER NUMBER 5
0521          L6              = [13, 8, 8, 0],        ! LETTER NUMBER 6
0522
0523          BC_LO           = [11, 0, 16, 0],       ! BYTE COUNT (LO ORDER)
0524          BC_HI           = [12, 0, 16, 0],       ! BYTE COUNT (HI ORDER)
0525          BUF_0           = [13, 0, 16, 0],       ! I/O BUFFER DESCRIPTOR
0526          BUF_1           = [14, 0, 16, 0],       !       :
0527          BUF_2           = [15, 0, 16, 0],       !       :
```

```
0528          BUF_3           = [16, 0, 16, 0],       !       :
0529          BUF_4           = [17, 0, 16, 0],       !       :
0530          BUF_5           = [18, 0, 16, 0],       !       V
0531          LBN_L           = [19, 0, 16, 0],       ! LOGICAL BLOCK NUMBER (LO ORDER)
0532          LBN_H           = [20, 0, 16, 0],       ! LOGICAL BLOCK NUMBER (HI ORDER)
0533  !
0534  !     SET CONTROLLER CHARACTERISTICS COMMAND FIELDS
0535  !
0536          C_FLAGS         = [12, 0, 16, 0],       ! CONTROLLER FLAGS
0537  !
0538  !     ONLINE COMMAND FIELDS
0539  !
0540          U_FLAGS         = [12, 0, 16, 0],       ! UNIT FLAGS
0541          DDPAR           = [19, 0, 16, 0]        ! DEVICE DEPENDENT PARAMETERS
0542          tes,
0543  !
0544  !****** RETURN PACKET (RETPKT) FIELDS
0545  !               (SIMILAR, BUT NOT IDENTICAL, TO MSCP ENVELOPE FIELDS)
0546  !
0547  RP_FIELDS =
0548          set
0549  !
0550  !     COMMON TO ALL RETURN PACKETS FROM DISK MSCP
0551  !
0552          MESLEN          = [0, 0, 16, 0],        ! MESSAGE LENGTH
0553          CTLR            = [1, 0, 4, 0],         ! CONTROLLER NUMBER (CREDITS OVERWRITTEN)
0554          MESTYP          = [1, 4, 4, 0],         ! MESSAGE TYPE
0555          CONID           = [1, 8, 8, 0],         ! CONNECTION ID
0556          CRF_LO          = [2, 0, 16, 0],        ! COMMAND REFERENCE NUMBER (LO ORDER)
0557          CRF_HI          = [3, 0, 16, 0],        ! COMMAND REFERENCE NUMBER (HI ORDER)
0558          DISK            = [4, 0, 16, 0],        ! DISK ADDRESS (RD/RX UNIT NUMBER)
0559          CMDMOD          = [5, 0, 16, 0],        ! COMMAND MODIFIERS
0560          ENDCOD          = [6, 0, 8, 0],         ! END CODE
0561          FLAGS           = [6, 8, 8, 0],         ! FLAGS
0562          STATUS          = [7, 0, 16, 0],        ! STATUS AND SUB-CODE
0563          STSCOD          = [7, 0, 5, 0],         ! STATUS CODE
0564          SUBCOD          = [7, 5, 11, 0],        ! SUB-CODE
0565  !
0566  !     READ, WRITE, AND ACCESS COMMAND RETURN PACKETS
0567  !
0568          BCNT_LO         = [8, 0, 16, 0],        ! BYTE COUNT (LO ORDER)
0569          BCNT_HI         = [9, 0, 16, 0],        ! BYTE COUNT (HI ORDER)
0570          BUFF_0          = [10, 0, 16, 0],       ! I/O BUFFER DESCRIPTOR (WORD 0)
0571          BUFF_1          = [11, 0, 16, 0],       ! I/O BUFFER DESCRIPTOR (WORD 1)
0572          BUFF_2          = [12, 0, 16, 0],       ! I/O BUFFER DESCRIPTOR (WORD 2)
0573          BUFF_3          = [13, 0, 16, 0],       ! I/O BUFFER DESCRIPTOR (WORD 3)
0574          BUFF_4          = [14, 0, 16, 0],       ! I/O BUFFER DESCRIPTOR (WORD 4)
0575          BUFF_5          = [15, 0, 16, 0],       ! I/O BUFFER DESCRIPTOR (WORD 5)
0576          BBLK_LO         = [16, 0, 16, 0],       ! FIRST BAD BLOCK (LO ORDER)
0577          BBLK_HI         = [17, 0, 16, 0],       ! FIRST BAD BLOCK (HI ORDER)
0578          CBCNT_LO        = [18, 0, 16, 0],       ! BYTE COUNT FROM CMD PACKET (LO ORDER)
0579          CBCNT_HI        = [19, 0, 16, 0],       ! BYTE COUNT FROM CMD PACKET (HI ORDER)
0580          LBN_LO          = [20, 0, 16, 0],       ! LOGICAL BLOCK NUMBER (LO ORDER)
0581          LBN_HI          = [21, 0, 16, 0],       ! LOGICAL BLOCK NUMBER (HI ORDER)
0582  !
0583  !     SET CONTROLLER CHARACTERISTICS RETURN PACKET
```

```
!
        C_FLGS          = [9, 0, 16, 0],        ! CONTROLLER FLAGS
        C_TIME          = [10, 0, 16, 0],       ! CONTROLLER TIMEOUT
!
!   UNIT ONLINE RETURN PACKET
!
        U_FLGS          = [9, 0, 16, 0],        ! UNIT FLAGS
        USIZ_LO         = [18, 0, 16, 0],       ! UNIT SIZE (LO ORDER)
        USIZ_HI         = [19, 0, 16, 0]        ! UNIT SIZE (HI ORDER)
        tes,
!
!
!****** STATISTICS TABLE (TALLY) FIELDS
!
T_FIELDS =
        set
        BYTES_READ_LO   = [0, 0, 16, 0],        ! NUMBER OF BYTES READ (LO ORDER)
        BYTES_READ_HI   = [1, 0, 16, 0],        ! NUMBER OF BYTES READ (HI ORDER)
        MBYTES_READ     = [2, 0, 16, 0],        ! MEGABYTES READ
        BYTES_WRIT_LO   = [3, 0, 16, 0],        ! NUMBER OF BYTES WRITTEN (LO ORDER)
        BYTES_WRIT_HI   = [4, 0, 16, 0],        ! NUMBER OF BYTES WRITTEN (HI ORDER)
        MBYTES_WRIT     = [5, 0, 16, 0],        ! MEGABYTES WRITTEN
        ERR_HARD        = [6, 0, 16, 0],        ! NUMBER OF HARD ERRORS
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
        TOT_READS_LO    = [7, 0, 16, 0],        ! TOTAL NUMBER OF READS (LO ORDER)
        TOT_READS_HI    = [8, 0, 16, 0],        ! TOTAL NUMBER OF READS (HI ORDER)

        TOT_WRITES_LO   = [10, 0, 16, 0],       ! TOTAL NUMBER OF WRITES (LO ORDER)
        TOT_WRITES_HI   = [11, 0, 16, 0],       ! TOTAL NUMBER OF WRITES (HI ORDER)

        TOT_BYT_RED_LO  = [13, 0, 16, 0],       ! TOTAL BYTES READ (LO ORDER)
        TOT_BYT_RED_HI  = [14, 0, 16, 0],       ! TOTAL BYTES READ (HI ORDER)
        MTOT_BYT_RED    = [15, 0, 16, 0],       ! TOTAL MEGABYTES READ
        TOT_BYT_WRT_LO  = [16, 0, 16, 0],       ! TOTAL BYTES WRITTEN (LO ORDER)
        TOT_BYT_WRT_HI  = [17, 0, 16, 0],       ! TOTAL BYTES WRITTEN (HI ORDER)
        MTOT_BYT_WRT    = [18, 0, 16, 0],       ! TOTAL MEGABYTES WRITTEN
        T_BLK_WT        = [20, 0, 16, 0],       !
        T_DBN_WT        = [21, 0, 16, 0],       !
        T_BLK_RD        = [22, 0, 16, 0],       !
        T_DBN_RD        = [23, 0, 16, 0],       !
        ERR_HRD_SEK     = [24, 0, 8, 0],        ! TOTAL HARD ERRORS - SEEK
        ERR_HRD_DAT     = [24, 8, 8, 0],        ! TOTAL HARD ERRORS - DATA
        ERR_HRD_DRV     = [25, 0, 8, 0],        ! TOTAL HARD ERRORS - DRIVE
        ERR_HRD_HST     = [25, 8, 8, 0],        ! TOTAL HARD ERRORS - HOST
        ERR_SFT_SEK     = [26, 0, 8, 0],        ! TOTAL SOFT ERRORS - SEEK
        ERR_SFT_DAT     = [26, 8, 8, 0],        ! TOTAL SOFT ERRORS - DATA
        ERR_SFT_DRV     = [27, 0, 8, 0],        ! TOTAL SOFT ERRORS - DRIVE
        ERR_SFT_HST     = [27, 8, 8, 0]         ! TOTAL SOFT ERRORS - HOST
        tes,
!
!****** CONTROLLER ERROR TALLY FIELDS
!
C_ERR_FIELDS =
        set
        C_ERR_HRD       = [0, 0, 8, 0],         ! HARD ERRORRS
        C_ERR_SFT       = [0, 8, 8, 0]          ! SOFT ERRORS
```

```
;       0640          tes,
;       0641
;       0642  !
;       0643  !****** DRIVER CONTROLLER TABLE (DCT) FIELDS
;       0644  !
;       0645  DCT_FIELDS =
;       0646          set
;       0647          WORD0           = [0, 0, 16, 0],        ! ALL FIELDS IN WORD 0
;       0648          CRING_CNT       = [0, 0, 8, 0],         ! NUMBER OF SLOTS IN CRING NOT YET RETURNED TO HOST
;       0649          IG_INT          = [0, 14, 1, 0],        ! IGNORE INTERRUPT BIT
;       0650          STAT            = [0, 15, 1, 0],        ! ONLINE / OFFLINE STATUS
;       0651          SA_SAVE         = [1, 0, 16, 0],        ! SA REGISTER SAVE WORD
;       0652          RR_BEG          = [2, 0, 16, 0],        ! FIXED ADDRESSES OF START AND
;       0653          RR_END          = [3, 0, 16, 0],        !       END OF EACH RING
;       0654          CR_BEG          = [4, 0, 16, 0],        !               :
;       0655          CR_END          = [5, 0, 16, 0],        !               V
;       0656          RR_POLL         = [6, 0, 16, 0],        ! ADDR OF NEXT RRING SLOT TO BE POLLED
;       0657          CR_POLL         = [7, 0, 16, 0],        ! ADDR OF NEXT CRING SLOT TO BE POLLED
;       0658          CR_NEXT         = [8, 0, 16, 0]         ! ADDR OF NEXT AVAIL CRING SLOT
;       0659          tes,
;       0660  !
;       0661  !****** ERROR LOG PACKET SAVE AREA FIELDS
;       0662  !
;       0663  EP_FIELDS =
;       0664          set
;       0665          EL_CNTR         = [0, 0, 8, 0],         ! CONTROLLER NUMBER
;       0666          EL_CONTENTS     = [0, 8, 8, 0],         ! FLAG INDICATES IF PACKET CONTENTS ALREADY PRINTED
;       0667          EL_MSGLEN       = [1, 0, 16, 0],        ! PACKET LENGTH
;       0668          EL_CRN_LO       = [3, 0, 16, 0],        ! COMMAND REFERENCE NUMBER
;       0669          EL_CRN_HI       = [4, 0, 16, 0],        !
;       0670          EL_DK_NUM       = [5, 0, 16, 0],        ! DISK ADDRESS (RD/RX DISK NUMBER)
;       0671          EL_FORMAT       = [7, 0, 8, 0],         ! FORMAT
;       0672          EL_CONTINUE     = [7, 14, 1, 0],        ! CONTINUE FLAG
;       0673          EL_SUCCESS      = [7, 15, 1, 0],        ! SUCCESS FLAG
;       0674          EL_CODE         = [8, 0, 5, 0],         ! ERROR CODE
;       0675          EL_SUBCODE      = [8, 5, 11, 0],        ! SUB CODE
;       0676          EL_BLOCK        = [23, 0, 16, 0],       ! BLOCK NUMBER
;       0677          EL_BLOCK_TYPE   = [24, 12, 4, 0]        ! TYPE OF BLOCK NUMBER INFO RETURNED
;       0678          tes,
;       0679  !
;       0680  !****** INFORMATION ABOUT LAST RESPONSE PACKET
;       0681  !
;       0682  LAST_PKT_FIELDS =
;       0683          set
;       0684          LAST_HRD_ERR    = [0, 0, 16, 0],        ! FLAG INDICATES IF HARD ERROR OCCURED
;       0685          LAST_CRN_LO     = [1, 0, 16, 0],        ! COMMAND REFERENCE NUMBER
;       0686          LAST_CRN_HI     = [2, 0, 16, 0]         !
;       0687          tes,
;       0688  !
;       0689  !****** RDRX REGISTER FIELDS
;       0690  !
;       0691  RC_REG =
;       0692          set
;       0693          RC_ALL          = [0, 16, 0]            ! DEFINE ALL BITS
;       0694          tes;
```

```
;       0695  !**************************************************************************
;       0696  !                                                                         *
;       0697  !                          M A C R O S                                    *
;       0698  !                                                                         *
;       0699  !**************************************************************************
;       0700  !
;       0701  macro
;       0702  !
;       0703  !****** CST FIELDS (WORDS 3 - 6)
;       0704  !
;       0705          D_ALL           = 0, 16, 0%,            ! ALL FIELDS
;       0706          D_DISK_NUM      = 0, 2, 0%,             ! DISK ADDRESS
;       0707          D_TYPE          = 2, 1, 0%,             ! DISK TYPE
;       0708          D_UNIT          = 8, 4, 0%,             ! DISK UNIT NUMBER (DRS UNIT)
;       0709          D_FATAL         = 12, 1, 0%,            ! FATAL ERROR BIT
;       0710          D_STAT          = 13, 1, 0%,            ! DISK STATUS BIT
;       0711          D_PRES          = 14, 1, 0%,            ! DISK PRESENT BIT
;       0712          D_PROT          = 15, 1, 0%,            ! DISK PROTECTION BIT
;       0713          D_BEG           = 0, 16, 0%,            ! BEGINNNING TRACK
;       0714          D_END           = 0, 16, 0%,            ! ENDING TRACK
;       0715          D_DBN           = 0, 8, 0%,             ! RELATIVE DBN!†
;       0716          D_COUNT         = 0, 16, 0%,            ! MSCP FUNC COUNTER
;       0717          DUPWRITE        = 12, 1, 0%,            ! DUP WPITE FLAG
;       0718          D_ACTIVE        = 13, 1, 0%,            ! ACTIVE STATE
;       0719          DUPERROR        = 14, 1, 0%,            ! DUP ERROR FLAG!†
;       0720          NODUPMEDIA      = 15, 1, 0%,            ! CONTROLLER LOCAL DUP MEDIA
;       0721
;       0722  !
;       0723  !****** BIT TEST
;       0724  !                 (CAUTION: THE FIRST ARGUMENT IS THE ADDRESS AND NOT THE CONTENTS)
;       0725  !
;     M 0726  BIT_TST (ADDR, EXPECTED) =
;     M 0727      (if (.ADDR and EXPECTED) eql EXPECTED
;     M 0728      then
;     M 0729          TRUE
;     M 0730      else
;       0731          FALSE )%,
;       0732  !
;       0733  !****** RDRX WRITE
;       0734  !
;     M 0735  WRT_RDRX (0, FIELDNAM, IMAGE) =
;     M 0736      begin
;     M 0737      local
;     M 0738          RC_REG;
;     M 0739      RC_REG <%fieldexpand (FIELDNAM)> = IMAGE;
;     M 0740      (.RDRX_ADDR + (%upval * 0)) = .RC_REG;
;       0741      end%;
```

```
;       0742  !*****************************************************************************
;       0743  !                                                                            *
;       0744  !                        S T R U C T U R E S                                 *
;       0745  !                                                                            *
;       0746  !*****************************************************************************
;       0747  !
;       0748  !****** NIBBLE (4-BIT) VECTOR STRUCTURE
;       0749  !
;       0750  !structure
;       0751  !    NIBVECTOR [I; N] =
;       0752  !        [(N + 1) / 2]
;       0753  !        (NIBVECTOR + I / 2) <(I + 2) and 4, 4>;
;       0754  !
;       0755  !****** RDRX ACCESS ALGORITHM
;       0756  !
;       0757  structure
;       0758      RDRX [O, P, S, E] =
;       0759          begin
;       0760          local
;       0761              RC_REG;
;       0762          RC_REG = .(RDRX + %upval * 0) <0, %bpval, 0>;
;       0763          RC_REG
;       0764          end
;       0765          <P, S, E>;

;                                   COMMAND QUALIFIERS
;       BLISS /PDP11 CNRQAA.REQ/LIST=CNRQAA.LIS/LIBRARY=CNRQAA.L16/SOURCE=PAGE:56

; Run Time:          00:06.7
; Elapsed Time:      00.11.4
; Memory Used:  62 pages
; L brary Precompilation Complete
```

```
CNRQAA.EXE   Memory allocation map  IKB M40.02      Page 1
                                                                                EQ 04:
                    15 DEC-83   10:47

Partition name : DUMMY
Identification : V01.0
Task  UIC      : [300,20]
Task attributes: -HD
Total address windows: 1.
Task  image  size  : 16608. words
Task address limits: 002000 102653
R W disk blk limits: 000002 000102 000101 00065.

*** Root segment: CNRQAA


R/W mem  limits: 002000 102653 100654 33196.
Disk blk limits: 000002 000102 000101 00065.


Memory allocation synopsis:

Section                                          Title  Ident  File
  - -                                            -----   ----  -- -
. BLK.:(RW,I,LCL,REL,CON)      002000 000000 00000.
$CODE$:(RO,I,LCL,REL,CON)      002000 071546 29542.
                               002000 023756 10222. NRQAM1 V01.0  CNRQAA.OB1;8
                               025756 020112 08266. NRQAM2 V01.2  CNRQAA.OB1;8
                               046070 025032 10778. NRQAM3 V01.0  CNRQAA.OB2;6
                               073122 000316 00206. B16MUL 2.8    CNRQAA.OLB;1
                               073440 000106 00070. B16SAV 2.4    CNRQAA.OLB;1
$FFF$ :(RO,I,LCL,REL,CON)      073546 005236 02718.
                               073546 005236 02718. NRQAM1 V01.0  CNRQAA.OB1;8
$GGG$ :(RO,I,LCL,REL,CON)      101004 001256 00686.
                               101004 001256 00686. NRQAM3 V01.0  CNRQAA.OB2;6
$OWN$ :(RW,D,LCL,REL,CON)      102262 000224 00148.
                               102262 000224 00148. NRQAM2 V01.2  CNRQAA.OB1;8
$PLIT$:(RO,D,LCL,REL,CON)      102506 000054 00044.
                               102506 000030 00024. NRQAM2 V01.2  CNRQAA.OB1;8
                               102536 000024 00020. NRQAM3 V01.0  CNRQAA.OB2;6
$XYZ$ :(RO,I,LCL,REL,CON)      102562 000070 00056.
                               102562 000070 00056. NRQAM4 V01.2  CNRQAA.OB2;6


Global symbols:

ADR     000020     BIT08  000400     BIT5    000040     BUFF.0 100702-R  CST     073546-R  DF.3    022010-R  DU.RSN 004616-R
ASTERI  025640-R   BIT09  001000     BIT6    000100     BYTS.P 100746-R  CST.AD  073624-R  DF.4    022050-R  D$PCNT 002122-R
AZINTO  067140-R   BIT1   000002     BIT7    000200     CCTLR  100726-R  CTLR.C  100734-R  DF.5    022076-R  EBD.10 007236-R
BIT0    000001     BIT10  002000     BIT8    000400     CDISK  100730-R  CUOFF   100732-R  DF.6    022122-R  EBD.12 007276-R
BIT00   000001     BIT11  004000     BIT9    001000     CER.01 014054-R  C.ERR.  075222-R  DF.7    022172-R  EBD.13 007344-R
BIT01   000002     BIT12  010000     BL$DIV  073346-R   CER.02 014120-R  DASH    025632-R  DROP.C  034264-R  EBD.14 007376-R
BIT02   000004     BIT13  020000     BL$LAS  102562-R   CMD.TI 100762-R  DCT     073626-R  DRV.CT  034372-R  EBD.18 007436-R
BIT03   000010     BIT14  040000     BL$MOD  073360-R   CNTR.E 025042-R  DCT.AD  073650-R  DUPPKT  073656-R  EBD.19 007472-R
BIT04   000020     BIT15  100000     BL$MUL  073122-R   CREDIT 101000-R  DFPTBL  025650-R  DUPROU  025704-R  EBH.30 021160-R
BIT05   000040     BIT2   000004     BL$SHF  073372-R   CRLF   025626-R  DF.0    021644-R  DUP.FL  100724-R  EBH.44 021176-R
BIT06   000100     BIT3   000010     BOE     000400     CRN.HI 100770-R  DF.1    021700-R  DUR     100736-R  EBH.45 021300 R
BIT07   000200     BIT4   000020     BUFF.A  100662-R   CRN.LO 100766-R  DF.2    021734-R  DU.MSG  004052-R  EBH.46 021332 R
```

CNRQAA 15 DEC 83 10:47

```
EBH.47 021362-R  EMS.10 045550-R  F.19   021016-R  HWQ5   002220 R  L$LOAD 002100-R  PUTA.B 034100-R  SC.IOP 015034 R
EBH.48 021444-R  EMS.12 045612 R  F.2    017746 R  HWQ6   002310-R  L$LUN  002074-R  PUT.IO 034034 R  SC.ISH 015340 R
EBH.49 021532-R  EMS.13 045650-R  F.20   021056 R  HWQ7   002326-R  L$MREV 002050 R  PUT.PK 033476-R  SC.IS2 015420-R
EBNEX1 013606-R  EMS.14 045712 R  F.21   021130-R  HWQ8   002406-R  L$NAME 002000 R  PUT.RE 033744-R  SC.NXM 016702 R
EBNEX3 013760-R  EMS.18 045754-R  F.3    020002-R  HWQ9   002460-R  L$NDHR 026210-R  P.INDE 100772-R  SC.ODA 016630-R
EBS.01 007174 R  EMS.21 046016-R  F.4    020034-R  IBE    010000    L$NDHW 025664-R  QIO    100742-R  SC.ODB 016660-R
EB.ADD 013544 R  EMS.22 046034-R  F.5    020074-R  IDU    000040    L$NDSF 026436 R  RDRX.A 073652-R  SC.ONL 014632 R
EB.COM 013360-R  EMS.30 046052-R  F.6    020152-R  IER    020000    L$NDSW 025746-R  RDRX.E 025562-R  SC.PAR 016736-R
EB.DCT 013304-R  ENTRY. 100722-R  F.7    020226-R  IN.IOD 034176-R  L$PRIO 002042-R  RETPKT 076722-R  SC.POE 017366-R
EB.NEX 013662 R  EOP.FL 100723-R  F.8    020302-R  IODQ   100712-R  L$PROT 025750-R  RPT1   004766-R  SC.PSP 017630 R
EB.PKT 013442-R  ERRBLK 002134-R  F.9    020360 R  IODQ.I 100716-R  L$PRT  002112 R  RPT10  005472-R  SC.RCT 015742-R
EB.RAL 013504-R  ERRMSG 002132-R  GET.IO 033760-R  IODQ.O 100720-R  L$REPP 002062-R  RPT11  005524-R  SC.RDY 017422-R
EF.CON 000036    ERRNBR 002130-R  GET.PK 033112-R  IPKT.A 076704-R  L$REV  002010-R  RPT12  005612-R  SC.RSP 017532 R
EF.NEW 000035    ERRTYP 002126-R  GET.RE 033636-R  IRDRX. 073654-R  L$RPT  027574-R  RPT13  005660-R  SC.SDI 014534 R
EF.PWR 000034    ERR.CO 011012-R  GP$DIS 026324-R  ISR    000100    L$SFTL 026212-R  RPT14  005760-R  SC.SDS 017044-R
EF.RES 000037    EVL    000004    GP$1   026012-R  IXE    004000    L$SOFT 026214-R  RPT15  006056-R  SC.SON 014654 R
EF.STA 000040    EX.ABP 014524-R  GP$10  026140-R  LOE    040000    L$SPC  002056-R  RPT16  006156-R  SC.SRI 017300-R
EGD.10 006352-R  EX.ACC 014322-R  GP$11  026152-R  LOT    000010    L$SPCP 002020-R  RPT2   005052-R  SC.SRT 017206-R
EGD.11 006412-R  EX.BDR 011414-R  GP$12  026170-R  L$ACP  002110-R  L$SPTP 002024-R  RPT3   005116-R  SC.SUR 017600-R
EGD.12 006436-R  EX.BDW 011504-R  GP$13  026200-R  L$APT  002036-R  L$STA  002030-R  RPT4   005202-R  SC.SWP 016530 R
EGD.13 006464-R  EX.CBR 011760-R  GP$14  026214-R  L$AU   032676-R  L$SW   025670-R  RPT5   005246-R  SC.UNK 014674-R
EGD.14 006512-R  EX.CBW 012030-R  GP$15  026226-R  L$AUT  002070-R  L$SWLE 025666-R  RPT6   005334-R  SC.VOL 014754-R
EGD.15 006542-R  EX.CRD 014214-R  GP$16  026240-R  L$AUTO 032104-R  L$TEST 002114-R  RPT7   005400-R  SC.576 016036-R
EGD.16 006560-R  EX.DGM 014264-R  GP$17  026252-R  L$CCP  002106-R  L$TIML 002014-R  RPT8   005422-R  SEND   034450-R
EGD.17 006610-R  EX.ELP 014444-R  GP$18  026264-R  L$CLEA 032170-R  L$UNIT 002012-R  RPT9   005444-R  SET.CP 032716-R
EGD.18 006626-R  EX.ESP 014414-R  GP$19  026272-R  L$CO   002032-R  MSCP.P 075224-R  RP.ADD 077230-R  SET.UP 033012 R
EGD.20 006646-R  EX.GDS 014372-R  GP$2   026022-R  L$DEPO 002011-R  MSG.01 004646-R  RP.IND 077226-R  SFPTBL 025670-R
EGD.21 006734-R  EX.LBR 011572-R  GP$20  026306-R  L$DESC 025766-R  MSG.02 004700-R  RP.USE 077222-R  SPACE4 025622 R
EGD.22 007046-R  EX.LBW 011636-R  GP$21  026316-R  L$DESP 002076-R  MSG.03 004734-R  SA.REG 100760-R  STEP   100754 R
EGD.23 007106-R  EX.MTN 014244-R  GP$22  026330-R  L$DEVP 002060-R  M.ASC  022630-R  SB.COD 100752-R  ST.COD 100750-R
EGH.30 007150-R  EX.ONL 014334-R  GP$23  026344-R  L$DISP 002124-R  M.BIN  022664-R  SC.CLK 017504-R  SWM1   003760-R
EGS.01 006240-R  EX.OP  025610-R  GP$24  026354-R  L$DLY  002116-R  M.COD  022756-R  SC.CON 014560-R  SWP.DP 025676 R
EGS.02 006260-R  EX.RBN 011702-R  GP$25  026370-R  L$DTP  002040-R  M.DAT  023014-R  SC.CTO 016772-R  SWP.ER 025670-R
EH.0   007552-R  EX.RCD 014504-R  GP$26  026406-R  L$DTYP 002034-R  M.TER  022722-R  SC.DIS 015102-R  SWP.FL 025674 R
EH.1   007610-R  EX.RD  014302-R  GP$27  026420-R  L$DU   032604-R  M.UL   023264-R  SC.DST 015500-R  SWP.RA 025702 R
EH.10  010172 R  EX.SCC 014346-R  GP$3   026032-R  L$DUT  002072-R  M.UP   023204-R  SC.DS2 015554-R  SWP.UC 025700-R
EH.12  010224-R  EX.SDD 014470-R  GP$4   026044-R  L$DVTY 025756-R  M.UR   023046-R  SC.DUP 014602-R  SWP.UD 025706-R
EH.13  010262-R  EX.SEQ 014174-R  GP$5   026056-R  L$EF   002052-R  M.URP  023116-R  SC.ECC 015626-R  SWP.XF 025672 R
EH.2   007646-R  EX.WRD 025600-R  GP$6   026066-R  L$ENVI 002044-R  NEX    100764-R  SC.ECD 015710-R  SWQ1   002700-R
EH.3   007706-R  EX.WRT 014312-R  GP$7   026076-R  L$ERRT 002126-R  NEXT.P 101002-R  SC.EC1 016160-R  SWQ10  003154-R
EH.4   007744-R  E.BLK  022466-R  GP$8   026106-R  L$ETP  002102-R  NEX.TR 032706-R  SC.EC2 016210-R  SWQ11  003220-R
EH.5   007770-R  E.DEV  022550-R  GP$9   026120-R  L$EXP1 002046-R  NULL   004050-R  SC.EC3 016240-R  SWQ12  003252 R
EH.6   010020-R  E.UNT  022432-R  HOE    100000    L$EXP4 002064-R  OF.RC  100756-R  SC.EC4 016272-R  SWQ13  003350-R
EH.7   010052-R  E.ZER  022574-R  HWPT.B 025654-R  L$EXP5 002066-R  OUT.IO 034140-R  SC.EC5 016322-R  SWQ14  003426-R
EH.8   010104-R  FREE.M 100744-R  HWPT.D 025656-R  L$HARD 026012-R  PKT.US 076706-R  SC.EC6 016352-R  SWQ15  003444 R
EH.9   010140-R  F.1    017712-R  HWPT.E 025662-R  L$HIME 002120-R  PNT    001000    SC.EC7 016402-R  SWQ17  003514-R
ELG.FM 011402-R  F.10   020400-R  HWPT.I 025650-R  L$HPCP 002016-R  PRI    002000    SC.EC8 016434-R  SWQ19  003602-R
ELG.OO 011046-R  F.11   020430-R  HWPT.S 025660-R  L$HPTP 002022-R  PRI00  000000    SC.EC9 016466-R  SWQ2   002722-R
ELOG.P 077232-R  F.12   020454-R  HWPT.V 025652-R  L$HRDL 026010-R  PRI01  000040    SC.EDC 017122-R  SWQ22  003672-R
EMSCMD 036134-R  F.13   020502-R  HWQ1   002136-R  L$HW   025650-R  PRI02  000100    SC.FCT 016112-R  SWQ4   003004-R
EMS.BL 043610-R  F.14   020560-R  HWQ10  002560-R  L$HWLE 025646-R  PRI03  000140    SC.FER 015174-R  SWQ7   003026-R
EMS.CM 045170-R  F.15   020620-R  HWQ11  002646-R  L$ICP  002104-R  PRI04  000200    SC.FE2 015262-R  SWQ9   003100-R
EMS.DB 043060-R  F.16   020646-R  HWQ2   002152-R  L$INIT 032072-R  PRI05  000240    SC.FUL 015762-R  S.DUPP 100774-R
EMS.EL 044050-R  F.17   020712-R  HWQ3   002162-R  L$LADP 002026-R  PRI06  000300    SC.HWP 016570-R  S.PATT 100776 R
EMS.01 045512-R  F.18   020756-R  HWQ4   002174-R  L$LAST 102566-R  PRI07  000340    SC.IDS 017136-R  TALLY  074660-R
```

```
T$FREE 102646 R  T.SPL  022412 R  XX15  012136-R  XX22  012322-R  XX30  012664-R  XX38  013142-R  $SAVE4 073472 R
T$PTHV 000003    T.TER  022342-R  XX16  012160-R  XX23  012356-R  XX31  012712-R  XX39  013206-R  $SAVE5 073512 R
T.ADDR 075220-R  T1     046266-R  XX17  012206-R  XX24  012412-R  XX32  012744-R  XX40  013234-R
T.DEF  022264 R  UAM    000200    XX18  012224-R  XX25  012452-R  XX33  012772-R  XX41  013254 R
T.FAT  022366 R  WAIT   035064-R  XX19  012234 R  XX26  012522-R  XX34  013030-R  $END.L 102650-R
T.INF  022316 R  XX13   012100 R  XX20  012246 R  XX27  012576-R  XX35  013100 R  $SAVE2 073440 R
T.QUE  022242 R  XX14   012122-R  XX21  012262 R  XX29  012640 R  XX37  013116 R  $SAVE3 073454 R


*** Task builder statistics:

    Total work file references: 124902.
    Work file reads: 0.
    Work file writes: 0.
    Size of core pool: 5616. words (21. pages)
    Size of work file: 4608. words (18. pages)

    Elapsed time:00:00:53
```

GLOBAL CROSS REFERENCE CREF V02

SYMBOL VALUE REFERENCES...

ADR 000020 # NRQAM1 # NRQAM2 # NRQAM3
ASTERI 025640 R # NRQAM1 NRQAM2
AZINTO 067140-R # NRQAM3
BIT0 000001 # NRQAM1 # NRQAM2 # NRQAM3
BIT00 000001 # NRQAM1 # NRQAM2 # NRQAM3
BIT01 000002 # NRQAM1 # NRQAM2 # NRQAM3
BIT02 000004 # NRQAM1 # NRQAM2 # NRQAM3
BIT03 00001C # NRQAM1 # NRQAM2 # NRQAM3
BIT04 000020 # NRQAM1 # NRQAM2 # NRQAM3
BIT05 000040 # NRQAM1 # NRQAM2 # NRQAM3
BIT06 000100 # NRQAM1 # NRQAM2 # NRQAM3
BIT07 000200 # NRQAM1 # NRQAM2 # NRQAM3
BIT08 000400 # NRQAM1 # NRQAM2 # NRQAM3
BIT09 001000 # NRQAM1 # NRQAM2 # NRQAM3
BIT1 000002 # NRQAM1 # NRQAM2 # NRQAM3
BIT10 002000 # NRQAM1 # NRQAM2 # NRQAM3
BIT11 004000 # NRQAM1 # NRQAM2 # NRQAM3
BIT12 010000 # NRQAM1 # NRQAM2 # NRQAM3
BIT13 020000 # NRQAM1 # NRQAM2 # NRQAM3
BIT14 040000 # NRQAM1 # NRQAM2 # NRQAM3
BIT15 100000 # NRQAM1 # NRQAM2 # NRQAM3
BIT2 000004 # NRQAM1 # NRQAM2 # NRQAM3
BIT3 000010 # NRQAM1 # NRQAM2 # NRQAM3
BIT4 000020 # NRQAM1 # NRQAM2 # NRQAM3
BIT5 000040 # NRQAM1 # NRQAM2 # NRQAM3
BIT6 000100 # NRQAM1 # NRQAM2 # NRQAM3
BIT7 000200 # NRQAM1 # NRQAM2 # NRQAM3
BIT8 000400 # NRQAM1 # NRQAM2 # NRQAM3
BIT9 001000 # NRQAM1 # NRQAM2 # NRQAM3
BL$DIV 073346-R # B16MUL NRQAM2 NRQAM3
BL$LAS 102562-R # NRQAM4
BL$MOD 073360-R # B16MUL NRQAM3
BL$MUL 073122-R # B16MUL NRQAM2 NRQAM3
BL$SHF 073372-R # B16MUL NRQAM3
BOE 000400 # NRQAM1 # NRQAM2 # NRQAM3
BUFF.A 100662-R # NRQAM1 NRQAM2 NRQAM3
BUFF.O 100702-R # NRQAM1 NRQAM2 NRQAM3
BYTS.P 100746-R # NRQAM1 NRQAM2 NRQAM3
CCTLR 100726-R # NRQAM1 NRQAM2 NRQAM3
CDISK 100730 R # NRQAM1 NRQAM2 NRQAM3
CER.01 014054-R # NRQAM1 NRQAM2
CER.02 014120-R # NRQAM1 NRQAM2
CMD.TI 100762-R # NRQAM1 NRQAM2 NRQAM3
CNTR.E 025042-R # NRQAM1 NRQAM2
CREDIT 101000-R # NRQAM1 NRQAM2 NRQAM3
CRLF 025626-R # NRQAM1 NRQAM2 NRQAM3
CRN.HI 100770-R # NRQAM1 NRQAM2 NRQAM3
CRN.LO 100766-R # NRQAM1 NRQAM2 NRQAM3
CST 073546-R # NRQAM1 NRQAM2 NRQAM3
CST.AD 073624-R # NRQAM1 NRQAM2 NRQAM3
CTLR.C 100734-R # NRQAM1 NRQAM2 NRQAM3
CUOFF 100732 R # NRQAM1 NRQAM2 NRQAM3

GLOBAL CROSS REFERENCE CREF V

SYMBOL VALUE REFERENCES...

C.ERR.	075222 R	# NRQAM1	NRQAM2	NRQAM3	
DASH	025632 R	# NRQAM1	NRQAM2		
DCT	073626 R	# NRQAM1	NRQAM2	NRQAM3	
DCT.AD	073650 R	# NRQAM1	NRQAM2	NRQAM3	
DFPTBL	025650 R	# NRQAM1			
DF.0	021644 R	# NRQAM1	NRQAM2		
DF.1	021700 R	# NRQAM1	NRQAM2		
DF.2	021734 R	# NRQAM1	NRQAM2		
DF.3	022010 R	# NRQAM1	NRQAM2		
DF.4	022050 R	# NRQAM1	NRQAM2		
DF.5	022076-R	# NRQAM1	NRQAM2		
DF.6	022122-R	# NRQAM1	NRQAM2		
DF.7	022172-R	# NRQAM1	NRQAM2		
DROP.C	034264-R	# NRQAM2	NRQAM3		
DRV.CT	034372-R	# NRQAM2	NRQAM3		
DUPPKT	073656-R	# NRQAM1	NRQAM2	NRQAM3	
DUPROU	025704 R	# NRQAM1	NRQAM3		
DUP.FL	100724-R	# NRQAM1	NRQAM2	NRQAM3	
DUR	100736-R	# NRQAM1	NRQAM2	NRQAM3	
DU.MSG	004052-R	# NRQAM1	NRQAM2		
DU.RSN	004616 R	# NRQAM1	NRQAM2		
DSPCNT	002122-R	# NRQAM1			
EBD.10	007236-R	# NRQAM1	NRQAM2		
EBD.12	007276 R	# NRQAM1	NRQAM2		
EBD.13	007344-R	# NRQAM1	NRQAM2		
EBD.14	007376 R	# NRQAM1	NRQAM2		
EBD.18	007436-R	# NRQAM1	NRQAM2		
EBD.19	007472-R	# NRQAM1	NRQAM2		
EBH.30	021160-R	# NRQAM1	NRQAM2		
EBH.44	021176 R	# NRQAM1	NRQAM2		
EBH.45	021300-R	# NRQAM1	NRQAM2		
EBH.46	021332-R	# NRQAM1	NRQAM2		
EBH.47	021362-R	# NRQAM1	NRQAM2		
EBH.48	021444-R	# NRQAM1	NRQAM2		
EBH.49	021532-R	# NRQAM1	NRQAM2		
EBNEX1	013606-R	# NRQAM1	NRQAM2		
EBNEX3	013760-R	# NRQAM1	NRQAM2		
EBS.01	007174-R	# NRQAM1	NRQAM2		
EB.ADD	013544-R	# NRQAM1	NRQAM2		
EB.COM	013360-R	# NRQAM1	NRQAM2		
EB.DCT	013304-R	# NRQAM1	NRQAM2		
EB.NEX	013662-R	# NRQAM1	NRQAM2		
EB.PKT	013442-R	# NRQAM1	NRQAM2		
EB.RAL	013504-R	# NRQAM1	NRQAM2		
EF.CON	000036	# NRQAM1	# NRQAM2	# NRQAM3	
EF.NEW	000035	# NRQAM1	# NRQAM2	# NRQAM3	
EF.PWR	000034	# NRQAM1	# NRQAM2	# NRQAM3	
EF.RES	000037	# NRQAM1	# NRQAM2	# NRQAM3	
EF.STA	000040	# NRQAM1	# NRQAM2	# NRQAM3	
EGD.10	006352-R	# NRQAM1	NRQAM2	NRQAM3	
EGD.11	006412-R	# NRQAM1	NRQAM2	NRQAM3	
EGD.12	006436-R	# NRQAM1	NRQAM2	NRQAM3	

GLOBAL CROSS REFERENCE CREF V02

SYMBO	VALUE	REFERENCES...		
EGD.13	006464 R	# NRQAM1	NRQAM2	NRQAM3
EGD.14	006512 R	# NRQAM1	NRQAM2	NRQAM3
EGD.15	006542-R	# NRQAM1	NRQAM2	NRQAM3
EGD.16	006560 R	# NRQAM1	NRQAM2	NRQAM3
EGD.17	006610-R	# NRQAM1	NRQAM2	NRQAM3
EGD.18	006626-R	# NRQAM1	NRQAM2	NRQAM3
EGD.20	006646 R	# NRQAM1	NRQAM2	NRQAM3
EGD.21	006734 R	# NRQAM1	NRQAM2	NRQAM3
EGD.22	007046 R	# NRQAM1	NRQAM2	NRQAM3
EGD.23	007106 R	# NRQAM1	NRQAM2	NRQAM3
EGH.30	007150 R	# NRQAM1	NRQAM2	NRQAM3
EGS.01	006240 R	# NRQAM1	NRQAM2	
EGS.02	006260-R	# NRQAM1	NRQAM2	NRQAM3
EH.0	007552-R	# NRQAM1	NRQAM2	NRQAM3
EH.1	007610-R	# NRQAM1	NRQAM2	NRQAM3
EH.10	010172-R	# NRQAM1	NRQAM2	NRQAM3
EH.12	010224-R	# NRQAM1	NRQAM2	NRQAM3
EH.13	010262-R	# NRQAM1	NRQAM2	NRQAM3
EH.2	007646-R	# NRQAM1	NRQAM2	NRQAM3
EH.3	007706-R	# NRQAM1	NRQAM2	NRQAM3
EH.4	007744-R	# NRQAM1	NRQAM2	NRQAM3
EH.5	007770-R	# NRQAM1	NRQAM2	NRQAM3
EH.6	010020-R	# NRQAM1	NRQAM2	NRQAM3
EH.7	010052-R	# NRQAM1	NRQAM2	NRQAM3
EH.8	010104-R	# NRQAM1	NRQAM2	NRQAM3
EH.9	010140-R	# NRQAM1	NRQAM2	NRQAM3
ELG.FM	011402-R	# NRQAM1	NRQAM2	
ELG.00	011046-R	# NRQAM1	NRQAM2	
ELOG.P	077232-R	# NRQAM1	NRQAM2	NRQAM3
EMSCMD	036134-R	# NRQAM2	NRQAM3	
EMS.BL	043610-R	# NRQAM2		
EMS.CM	045170-R	# NRQAM2	NRQAM3	
EMS.DB	043060-R	# NRQAM2		
EMS.EL	044050-R	# NRQAM2	NRQAM3	
EMS.01	045512-R	# NRQAM2		
EMS.10	045550-R	# NRQAM2	NRQAM3	
EMS.12	045612-R	# NRQAM2	NRQAM3	
EMS.13	045650-R	# NRQAM2	NRQAM3	
EMS.14	045712-R	# NRQAM2	NRQAM3	
EMS.18	045754-R	# NRQAM2	NRQAM3	
EMS.21	046016-R	# NRQAM2	NRQAM3	
EMS.22	046034-R	# NRQAM2	NRQAM3	
EMS.30	046052-R	# NRQAM2	NRQAM3	
ENTRY.	100722-R	# NRQAM1	NRQAM2	NRQAM3
EOP.FL	100723-R	# NRQAM1	NRQAM2	NRQAM3
ERRBLK	002134-R	# NRQAM1		
ERRMSG	002132-R	# NRQAM1		
ERRNBR	002130-R	# NRQAM1		
ERRTYP	002126-R	# NRQAM1		
ERR.CO	011012-R	# NRQAM1	NRQAM2	
EVL	000004	# NRQAM1	# NRQAM2	# NRQAM3
EX.ABP	014524-R	# NRQAM1	NRQAM2	

GLOBAL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES...		
EX.ACC	014322-R	# NRQAM1	NRQAM2	
EX.BDR	011414 R	# NRQAM1	NRQAM2	
EX.BDW	011504-R	# NRQAM1	NRQAM2	
EX.CBR	011760 R	# NRQAM1	NRQAM2	
EX.CBW	012030-R	# NRQAM1	NRQAM2	
EX.CRD	014214-R	# NRQAM1	NRQAM2	
EX.DGM	014264-R	# NRQAM1	NRQAM2	
EX.ELP	014444 R	# NRQAM1	NRQAM2	
EX.ESP	014414 R	# NRQAM1	NRQAM2	
EX.GDS	014372-R	# NRQAM1	NRQAM2	
EX.LBR	011572-R	# NRQAM1	NRQAM2	
EX.LBW	011636-R	# NRQAM1	NRQAM2	
EX.MTN	014244-R	# NRQAM1	NRQAM2	
EX.ONL	014334-R	# NRQAM1	NRQAM2	
EX.OP	025610 R	# NRQAM1	NRQAM2	
EX.RBN	011702-R	# NRQAM1	NRQAM2	
EX.RCD	014504-R	# NRQAM1	NRQAM2	
EX.RD	014302-R	# NRQAM1	NRQAM2	
EX.SCC	014346-R	# NRQAM1	NRQAM2	
EX.SDD	014470-R	# NRQAM1	NRQAM2	
EX.SEQ	014174-R	# NRQAM1	NRQAM2	
EX.WRD	025600-R	# NRQAM1	NRQAM2	
EX.WRT	014312-R	# NRQAM1	NRQAM2	
E.BLK	022466-R	# NRQAM1	NRQAM2	
E.DEV	022550-R	# NRQAM1	NRQAM2	
E.UNT	022432 R	# NRQAM1	NRQAM2	
E.ZER	022574-R	# NRQAM1	NRQAM2	
FREE.M	100744-R	# NRQAM1	NRQAM2	NRQAM3
F.1	017712-R	# NRQAM1	NRQAM2	
F.10	020400-R	# NRQAM1	NRQAM2	
F.11	020430-R	# NRQAM1	NRQAM2	
F.12	020454-R	# NRQAM1	NRQAM2	
F.13	020502-R	# NRQAM1	NRQAM2	
F.14	020560-R	# NRQAM1	NRQAM2	
F.15	020620-R	# NRQAM1	NRQAM2	
F.16	020646-R	# NRQAM1	NRQAM2	
F.17	020712-R	# NRQAM1	NRQAM2	
F.18	020756-R	# NRQAM1	NRQAM2	
F.19	021016-R	# NRQAM1	NRQAM2	
F.2	017746-R	# NRQAM1	NRQAM2	
F.20	021056-R	# NRQAM1	NRQAM2	
F.21	021130-R	# NRQAM1	NRQAM2	
F.3	020002-R	# NRQAM1	NRQAM2	
F.4	020034-R	# NRQAM1	NRQAM2	
F.5	020074-R	# NRQAM1	NRQAM2	
F.6	020152-R	# NRQAM1	NRQAM2	
F.7	020226-R	# NRQAM1	NRQAM2	
F.8	020302-R	# NRQAM1	NRQAM2	
F.9	020360-R	# NRQAM1	NRQAM2	
GET.IO	033760-R	# NRQAM2	NRQAM3	
GET.PK	033112-R	# NRQAM2	NRQAM3	
GET.RE	033636 R	# NRQAM2	NRQAM3	

GLOBAL CROSS REFERENCE CRF .0

SYMBOL VALUE REFERENCES...

GP$DIS 026324 R # NRQAM2
GP$1 026012 R # NRQAM2
GP$10 026140-R # NRQAM2
GP$11 026152 R # NRQAM2
GP$12 026170-R # NRQAM2
GP$13 026200-R # NRQAM2
GP$14 026214-R # NRQAM2
GP$15 026226-R # NRQAM2
GP$16 026240-R # NRQAM2
GP$17 026252 R # NRQAM2
GP$18 026264-R # NRQAM2
GP$19 026272-R # NRQAM2
GP$2 026022-R # NRQAM2
GP$20 026306-R # NRQAM2
GP$21 026316-R # NRQAM2
GP$22 026330 R # NRQAM2
GP$23 026344-R # NRQAM2
GP$24 026354 R # NRQAM2
GP$25 026370-R # NRQAM2
GP$26 026406-R # NRQAM2
GP$27 026420-R # NRQAM2
GP$3 026032-R # NRQAM2
GP$4 026044-R # NRQAM2
GP$5 026056-R # NRQAM2
GP$6 026066-R # NRQAM2
GP$7 026076-R # NRQAM2
GP$8 026106-R # NRQAM2
GP$9 026120-R # NRQAM2
HOE 100000 # NRQAM1 # NRQAM2 # NRQAM3
HWPT.B 025654 R # NRQAM1
HWPT.D 025656-R # NRQAM1
HWPT.E 025662-R # NRQAM1
HWPT.I 025650-R # NRQAM1
HWPT.S 025660-R # NRQAM1
HWPT.V 025652-R # NRQAM1
HWQ1 002136-R # NRQAM1 NRQAM2
HWQ10 002560-R # NRQAM1 NRQAM2
HWQ11 002646-R # NRQAM1 NRQAM2
HWQ2 002152-R # NRQAM1 NRQAM2
HWQ3 002162-R # NRQAM1 NRQAM2
HWQ4 002174 R # NRQAM1 NRQAM2
HWQ5 002220-R # NRQAM1 NRQAM2
HWQ6 002310-R # NRQAM1 NRQAM2
HWQ7 002326-R # NRQAM1 NRQAM2
HWQ8 002406-R # NRQAM1 NRQAM2
HWQ9 002460 R # NRQAM1 NRQAM2
IBE 010000 # NRQAM1 # NRQAM2 # NRQAM3
IDU 000040 # NRQAM1 # NRQAM2 # NRQAM3
IER 020000 # NRQAM1 # NRQAM2 # NRQAM3
IN.IOO 034176-R # NRQAM2 NRQAM3
IOOQ 100712-R # NRQAM1 NRQAM2 NRQAM3
IOOQ.I 100716 R # NRQAM1 NRQAM2 NRQAM3

GLOBAL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES...		
IODQ.C	100720-R	# NRQAM1	NRQAM2	NRQAM3
IPKT.A	076704 R	# NRQAM1	NRQAM2	NRQAM3
IRDRX.	073654 R	# NRQAM1	NRQAM2	NRQAM3
ISR	000100	# NRQAM1	# NRQAM2	# NRQAM3
IXE	004000	# NRQAM1	# NRQAM2	# NRQAM3
LOE	040000	# NRQAM1	# NRQAM2	# NRQAM3
LOT	000010	# NRQAM1	# NRQAM2	# NRQAM3
L$ACP	002110-R	# NRQAM1		
L$APT	002036-R	# NRQAM1		
L$AU	032676 R	NRQAM1	# NRQAM2	
L$AUT	002070-R	# NRQAM1		
L$AUTO	032104-R	NRQAM1	# NRQAM2	
L$CCP	002106-R	# NRQAM1		
L$CLEA	032170-R	NRQAM1	# NRQAM2	
L$CO	002032-R	# NRQAM1		
L$DEPO	002011-R	# NRQAM1		
L$DESC	025766-R	NRQAM1	# NRQAM2	
L$DESP	002076-R	# NRQAM1		
L$DEVP	002060-R	# NRQAM1		
L$DISP	002124-R	# NRQAM1		
L$DLY	002116-R	# NRQAM1	NRQAM2	NRQAM3
L$DTP	002040-R	# NRQAM1		
L$DTYP	002034-R	# NRQAM1		
L$DU	032604-R	NRQAM1	# NRQAM2	
L$DUT	002072-R	# NRQAM1		
L$DVTY	025756-R	NRQAM1	# NRQAM2	
L$EF	002052-R	# NRQAM1		
L$ENVI	002044-R	# NRQAM1		
L$ERRT	002126-R	# NRQAM1		
L$ETP	002102-R	# NRQAM1		
L$EXP1	002046-R	# NRQAM1		
L$EXP4	002064-R	# NRQAM1		
L$EXP5	002066-R	# NRQAM1		
L$HARD	026012-R	NRQAM1	# NRQAM2	
L$HIME	002120-R	# NRQAM1	NRQAM2	
L$HPCP	002016-R	# NRQAM1		
L$HPTP	002022-R	# NRQAM1		
L$HRDL	026010-R	# NRQAM2		
L$HW	025650-R	# NRQAM1		
L$HWLE	025646-R	# NRQAM1		
L$ICP	002104-R	# NRQAM1		
L$INIT	032072-R	NRQAM1	# NRQAM2	
L$LADP	002026-R	# NRQAM1		
L$LAST	102566-R	NRQAM1	# NRQAM4	
L$LOAD	002100-R	# NRQAM1		
L$LUN	002074-R	# NRQAM1	NRQAM2	NRQAM3
L$MREV	002050-R	# NRQAM1		
L$NAME	002000-R	# NRQAM1		
L$NDHR	026210 R	# NRQAM2		
L$NDHW	025664-R	# NRQAM1		
L$NDSF	026436-R	# NRQAM2		
L$NDSW	025746 R	# NRQAM1		

SYMBOL	VALUE	REFERENCES...		
L$PRIO	002042-R	# NRQAM1		
L$PROT	025750-R	# NRQAM1		
L$PRT	002112-R	# NRQAM1		
L$REPP	002062-R	# NRQAM1		
L$REV	002010-R	# NRQAM1		
L$RPT	027574-R	NRQAM1	# NRQAM2	
L$SFTL	026212-R	# NRQAM2		
L$SOFT	026214 R	NRQAM1	# NRQAM2	
L$SPC	002056 R	# NRQAM1		
L$SPCP	002020-R	# NRQAM1		
L$SPTP	002024 R	# NRQAM1		
L$STA	002030-R	# NRQAM1		
L$SW	025670-R	# NRQAM1		
L$SWLE	025666-R	# NRQAM1		
L$TEST	002114 R	# NRQAM1		
L$TIML	002014-R	# NRQAM1		
L$UNIT	002012-R	# NRQAM1	NRQAM2	NRQAM3
MSCP.P	075224-R	# NRQAM1	NRQAM2	NRQAM3
MSG.01	004646-R	# NRQAM1	NRQAM2	
MSG.02	004700-R	# NRQAM1	NRQAM2	NRQAM3
MSG.03	004734-R	# NRQAM1	NRQAM2	NRQAM3
M.ASC	022630-R	# NRQAM1	NRQAM2	
M.BIN	022664-R	# NRQAM1	NRQAM2	
M.COD	022756-R	# NRQAM1	NRQAM2	
M.DAT	023014-R	# NRQAM1	NRQAM2	
M.TER	022722-R	# NRQAM1	NRQAM2	
M.UL	023264-R	# NRQAM1	NRQAM2	
M.UP	023204-R	# NRQAM1	NRQAM2	
M.UR	023046-R	# NRQAM1	NRQAM2	
M.URP	023116-R	# NRQAM1	NRQAM2	
NEX	100764-R	# NRQAM1	NRQAM2	NRQAM3
NEXT.P	101002-R	# NRQAM1	NRQAM2	NRQAM3
NEX.TR	032706-R	# NRQAM2	NRQAM3	
NULL	004050-R	# NRQAM1	NRQAM2	
OF.RC	100756-R	# NRQAM1	NRQAM2	NRQAM3
OUT.IO	034140-R	# NRQAM2	NRQAM3	
PKT.US	076706-R	# NRQAM1	NRQAM2	NRQAM3
PNT	001000	# NRQAM1	# NRQAM2	# NRQAM3
PRI	002000	# NRQAM1	# NRQAM2	# NRQAM3
PRI00	000000	# NRQAM1	# NRQAM2	# NRQAM3
PRI01	000040	# NRQAM1	# NRQAM2	# NRQAM3
PRI02	000100	# NRQAM1	# NRQAM2	# NRQAM3
PRI03	000140	# NRQAM1	# NRQAM2	# NRQAM3
PRI04	000200	# NRQAM1	# NRQAM2	# NRQAM3
PRI05	000240	# NRQAM1	# NRQAM2	# NRQAM3
PRI06	000300	# NRQAM1	# NRQAM2	# NRQAM3
PRI07	000340	# NRQAM1	# NRQAM2	# NRQAM3
PUTA.B	034100-R	# NRQAM2	NRQAM3	
PUT.IO	034034-R	# NRQAM2	NRQAM3	
PUT.PK	033476-R	# NRQAM2	NRQAM3	
PUT.RE	033744-R	# NRQAM2	NRQAM3	
P.INDE	100772-R	# NRQAM1	NRQAM2	NRQAM3

GLOBAL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES...		
QIO	100742 R	# NRQAM1	NRQAM2	NRQAM3
RDRX.A	073652 R	# NRQAM1	NRQAM2	NRQAM3
RDRX.E	025562 R	# NRQAM1	NRQAM2	
RETPKT	076722 R	# NRQAM1	NRQAM2	NRQAM3
RPT1	004766 R	# NRQAM1	NRQAM2	
RPT10	005472 R	# NRQAM1	NRQAM2	
RPT11	005524 R	# NRQAM1	NRQAM2	
RPT12	005612-R	# NRQAM1	NRQAM2	
RPT13	005660-R	# NRQAM1	NRQAM2	
RPT14	005760 R	# NRQAM1	NRQAM2	
RPT15	006056-R	# NRQAM1	NRQAM2	
RPT16	006156-R	# NRQAM1	NRQAM2	
RPT2	005052-R	# NRQAM1	NRQAM2	
RPT3	005116-R	# NRQAM1	NRQAM2	
RPT4	005202-R	# NRQAM1	NRQAM2	
RPT5	005246 R	# NRQAM1	NRQAM2	
RPT6	005334-R	# NRQAM1	NRQAM2	
RPT7	005400-R	# NRQAM1	NRQAM2	
RPT8	005422-R	# NRQAM1	NRQAM2	
RPT9	005444 R	# NRQAM1	NRQAM2	
RP.ADD	077230-R	# NRQAM1	NRQAM2	NRQAM3
RP.IND	077226-R	# NRQAM1	NRQAM2	NRQAM3
RP.USE	077222-R	# NRQAM1	NRQAM2	NRQAM3
SA.REG	100760 R	# NRQAM1	NRQAM2	NRQAM3
SB.COD	100752-R	# NRQAM1	NRQAM2	NRQAM3
SC.CLK	017504 R	# NRQAM1	NRQAM2	
SC.CON	014560-R	# NRQAM1	NRQAM2	
SC.CTO	016772 R	# NRQAM1	NRQAM2	
SC.DIS	015102-R	# NRQAM1	NRQAM2	
SC.DST	015500 R	# NRQAM1	NRQAM2	
SC.DS2	015554 R	# NRQAM1	NRQAM2	
SC.DUP	014602-R	# NRQAM1	NRQAM2	
SC.ECC	015626-R	# NRQAM1	NRQAM2	
SC.ECD	015710 R	# NRQAM1	NRQAM2	
SC.EC1	016160 R	# NRQAM1	NRQAM2	
SC.EC2	016210-R	# NRQAM1	NRQAM2	
SC.EC3	016240-R	# NRQAM1	NRQAM2	
SC.EC4	016272-R	# NRQAM1	NRQAM2	
SC.EC5	016322-R	# NRQAM1	NRQAM2	
SC.EC6	016352-R	# NRQAM1	NRQAM2	
SC.EC7	016402-R	# NRQAM1	NRQAM2	
SC.EC8	016434-R	# NRQAM1	NRQAM2	
SC.EC9	016466-R	# NRQAM1	NRQAM2	
SC.EDC	017122-R	# NRQAM1	NRQAM2	
SC.FCT	016112-R	# NRQAM1	NRQAM2	
SC.FER	015174-R	# NRQAM1	NRQAM2	
SC.FE2	015262-R	# NRQAM1	NRQAM2	
SC.FUL	015762-R	# NRQAM1	NRQAM2	
SC.HNP	016570-R	# NRQAM1	NRQAM2	
SC.IDS	017136-R	# NRQAM1	NRQAM2	
SC.IOP	015034-R	# NRQAM1	NRQAM2	
SC.ISH	015340-R	# NRQAM1	NRQAM2	

J1

GLOBAL CROSS REFERENCE CREI V02

SYMBOL	VALUE	REFERENCES...		
SC.IS2	015420 R	# NRQAM1	NRQAM2	
SC.NXM	016702 R	# NRQAM1	NRQAM2	
SC.ODA	016630-R	# NRQAM1	NRQAM2	
SC.ODB	016660 R	# NRQAM1	NRQAM2	
SC.ONL	014632-R	# NRQAM1	NRQAM2	
SC.PAR	016736-R	# NRQAM1	NRQAM2	
SC.POE	017366-R	# NRQAM1	NRQAM2	
SC.PSP	017630-R	# NRQAM1	NRQAM2	
SC.RCT	015742-R	# NRQAM1	NRQAM2	
SC.RDY	017422-R	# NRQAM1	NRQAM2	
SC.RSP	017532-R	# NRQAM1	NRQAM2	
SC.SDI	014534-R	# NRQAM1	NRQAM2	
SC.SDS	017044-R	# NRQAM1	NRQAM2	
SC.SON	014654 R	# NRQAM1	NRQAM2	
SC.SRI	017300 R	# NRQAM1	NRQAM2	
SC.SRT	017206-R	# NRQAM1	NRQAM2	
SC.SUR	017600-R	# NRQAM1	NRQAM2	
SC.SWP	016530-R	# NRQAM1	NRQAM2	
SC.UNK	014674 R	# NRQAM1	NRQAM2	
SC.VOL	014754 R	# NRQAM1	NRQAM2	
SC.576	016036 R	# NRQAM1	NRQAM2	
SEND	034450-R	# NRQAM2	NRQAM3	
SET.CP	032716 R	# NRQAM2	NRQAM3	
SET.UP	033012-R	# NRQAM2	NRQAM3	
SFPTBL	025670-R	# NRQAM1		
SPACE4	025622-R	# NRQAM1	NRQAM2	
STEP	100754-R	# NRQAM1	NRQAM2	NRQAM3
ST.COD	100750-R	# NRQAM1	NRQAM2	NRQAM3
SWM1	003760-R	# NRQAM1	NRQAM2	
SWP.DP	025676-R	# NRQAM1	NRQAM3	
SWP.ER	025670-R	# NRQAM1	NRQAM3	
SWP.FL	025674-R	# NRQAM1	NRQAM2	NRQAM3
SWP.RA	025702-R	# NRQAM1	NRQAM3	
SWP.UC	025700-R	# NRQAM1	NRQAM3	
SWP.UD	025706-R	# NRQAM1	NRQAM3	
SWP.XF	025672-R	# NRQAM1	NRQAM3	
SWQ1	002700-R	# NRQAM1	NRQAM2	
SWQ10	003154-R	# NRQAM1	NRQAM2	
SWQ11	003220-R	# NRQAM1	NRQAM2	
SWQ12	003252-R	# NRQAM1	NRQAM2	
SWQ13	003350-R	# NRQAM1	NRQAM2	
SWQ14	003426-R	# NRQAM1	NRQAM2	
SWQ15	003444-R	# NRQAM1	NRQAM2	
SWQ17	003514-R	# NRQAM1	NRQAM2	
SWQ19	003602-R	# NRQAM1	NRQAM2	
SWQ2	002722-R	# NRQAM1	NRQAM2	
SWQ22	003672-R	# NRQAM1	NRQAM2	
SWQ4	003004-R	# NRQAM1	NRQAM2	
SWQ7	003026-R	# NRQAM1	NRQAM2	
SWQ9	003100-R	# NRQAM1	NRQAM2	
S.DUPP	100774-R	# NRQAM1	NRQAM2	NRQAM3
S.PATT	100776-R	# NRQAM1	NRQAM2	NRQAM3

K1

GLOBAL CROSS REFERENCE CREF V02

SYMBOL	VALUE	REFERENCES...				
TALLY	074660 R	# NRQAM1	NRQAM2	NRQAM3		
T$FREE	102646 R	# NRQAM4				
T$PTHV	000003	NRQAM1	# NRQAM4			
T.ADDR	075220-R	# NRQAM1	NRQAM2	NRQAM3		
T.DEF	022264-R	# NRQAM1	NRQAM2			
T.FAT	022366-R	# NRQAM1	NRQAM2			
T.INF	022316-R	# NRQAM1	NRQAM2			
T.QUE	022242-R	# NRQAM1	NRQAM2			
T.SPL	022412-R	# NRQAM1	NRQAM2			
T.TER	022342-R	# NRQAM1	NRQAM2			
T1	046266-R	NRQAM1	# NRQAM3			
UAM	000200	# NRQAM1	# NRQAM2	# NRQAM3		
WAIT	035064-R	# NRQAM2	NRQAM3			
XX13	012100-R	# NRQAM1	NRQAM2			
XX14	012122-R	# NRQAM1	NRQAM2			
XX15	012136-R	# NRQAM1	NRQAM2			
XX16	012160-R	# NRQAM1	NRQAM2			
XX17	012206-R	# NRQAM1	NRQAM2			
XX18	012224-R	# NRQAM1	NRQAM2			
XX19	012234 R	# NRQAM1	NRQAM2			
XX20	012246 R	# NRQAM1	NRQAM2			
XX21	012262-R	# NRQAM1	NRQAM2			
XX22	012322 R	# NRQAM1	NRQAM2			
XX23	012356-R	# NRQAM1	NRQAM2			
XX24	012412-R	# NRQAM1	NRQAM2			
XX25	012452 R	# NRQAM1	NRQAM2			
XX26	012522-R	# NRQAM1	NRQAM2			
XX27	012576-R	# NRQAM1	NRQAM2			
XX29	012640-R	# NRQAM1	NRQAM2			
XX30	012664-R	# NRQAM1	NRQAM2			
XX31	012712-R	# NRQAM1	NRQAM2			
XX32	012744 R	# NRQAM1	NRQAM2			
XX33	012772 R	# NRQAM1	NRQAM2			
XX34	013030-R	# NRQAM1	NRQAM2			
XX35	013100-R	# NRQAM1	NRQAM2			
XX37	013116 R	# NRQAM1	NRQAM2			
XX38	013142-R	# NRQAM1	NRQAM2			
XX39	013206 R	# NRQAM1	NRQAM2			
XX40	013234-R	# NRQAM1	NRQAM2			
XX41	013254-R	# NRQAM1	NRQAM2			
$ENC.L	102650-R	# NRQAM4				
$SAVE2	073440-R	B16MUL	# B16SAV	NRQAM2	NRQAM3	
$SAVE3	073454 R	# B16SAV	NRQAM2	NRQAM3		
$SAVE4	073472-R	# B16SAV	NRQAM2	NRQAM3		
$SAVE5	073512 R	B16MUL	# B16SAV	NRQAM2	NRQAM3	