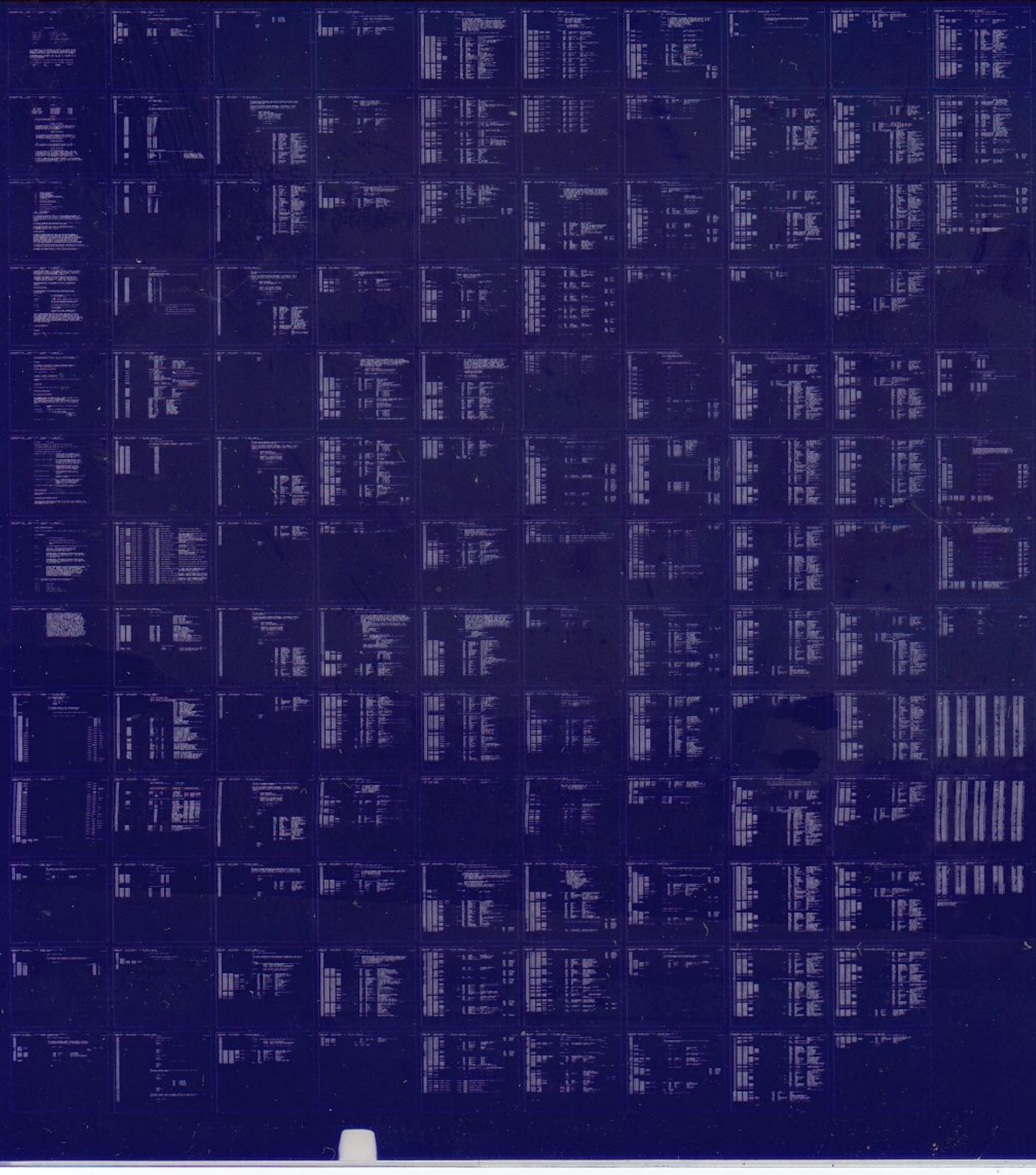
TU58 TU58 PERF EXERCISER AH-E649E-MC
CZTUUEO FICHE 1 OF 1
OCT 1983
COPYRIGHT@ 79-83
MADE IN USA



```
.REM &
```

IDENTIFICATION

PRODUCT CODE:	AC-E648E-MC
PRODUCT NAME:	CZTUUE0 TU58 PERF EXER
PRODUCT DATE:	11 JULY 1983
MAINTAINER:	TAPE DIAGNOSTIC ENGINEERING
AUTHOR:	R. J. ROSS

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

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

COPYRIGHT (C) 1979,1983 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

HISTORY

JUNE 18,1979	INITIAL RELEASE	CZTUUA0
JULY 1,1979	SECOND RELEASE	CZTUUB0
JUNE 1,1980	THIRD RELEASE	CZTUUB1
OCTOBER 1,1981	FOURTH RELEASE	CZTUUC0
MARCH 1,1982	FIFTH RELEASE	CZTUUD0
JUNE 1,1983	SIXTH RELEASE	CZTUUE0

CZTUUA0

1. INITIAL REALEASE--PERF. EXER. FOR UP TO 8 TU58 CONTROLLERS WITH ONE OR TWO DRIVES EACH.

CHANGES TO CZTUUA0

1. THE PROGRAM WAS MODIFIED TO RUN UNDER THE NEW DIAGNOSTIC SUPERVISOR CHSAA0. AS A RESULT OF THIS CONVERSION, THIS PROGRAM NOW OPERATES IN 8K AND PAPERTAPE DISTRIBUTION REQUIRES ONLY ONE PART AK-E650B-MC.

CHANGES TO CZTUUB0

1. "CLR @ XMSR(R5)" HAS BEEN CHANGED TO "DEC @ XMSR(R5)" TO ALLEVIATE THE PROBLEM OF DESTROYING ANY PREVIOUSLY SET PROGRAMMABLE SPEED IN THE DLV11-E,F, OR DC319 DLART WHEN THE TU58 INIT SEQUENCE WAS TERMINATED.

CHANGES TO CZTUUB1

1. TEST 9 WAS ADDED TO THE DIAGNOSTICS BECAUSE THE TU58 HAS BEEN UPDATED TO USE MODIFIED RADIAL SERIAL PROTOCOL.

CHANGES TO CZTUUC0

1. A TEST WAS ADDED TO VERIFY 128 BYTE/BLOCK MODE. THE TEST IS SIMULAR TO TEST 3. IT WRITES, READS, AND VERIFIES SEQUENTIAL BLOCKS OF TAPE FROM BLOCK 0 THOUGH BLOCK 2047. THIS IS DONE FOR EACH SELECTED DRIVE IN EACH SELECTED UNIT. THIS WILL BE TEST 4. TEST NUMBERED 4-8 WILL BECOME TEST 5-9.

2. IN TEST 9, 'MRSP' WILL BE TESTED DIFFERENTLY. IN THIS VERSION TO TEST THE NEED FOR HANDSHAKING. THE WAIT LOOP IS BEFORE SENDING THE 'CONTINUE' INSTEAD OF AFTER. THIS WILL VERIFY THAT THE TU58 CANNOT SEND DATA WITHOUT A HANDSHAKE.

TABLE OF CONTENTS

1.0 GENERAL INFORMATION

THIS DIAGNOSTIC EXERCISES FROM 1 TO 8 TU58 CONTROLLER BOARDS, EACH OF WHICH MAY SUPPORT 1 OR 2 DRIVES. THE PROGRAM IMPLEMENTS THE "MAINTENANCE MODE" SWITCH WITHIN ALL PACKET COMMANDS, THUS RETRIEVING MAXIMUM INFORMATION FROM THE DEVICE UPON CERTAIN DEVICE RECOGNIZED ERRORS.

STATISTICAL SUMMARIES ARE PROVIDED FOR ALL UNITS TESTED. RETRIES ARE PERFORMED ON DATA-RELATED ERROR CONDITIONS.

USE OF LOOP ON ERROR FLAG (:LOE) IS IMPLEMENTED BUT NOT RECOMMENDED FOR USE, SINCE THE LOOPS ARE QUITE LENGTHLY DUE TO COMMUNICATIONS PROTOCOL OVERHEAD.

1.1 PROGRAM ABSTRACT

IN ORDER TO EXERCISE MULTIPLE UNITS IN AN EFFICIENT MANNER, A SCHEDULING ALGORITHM BUILDS, THEN SENDS THE NEXT COMMUNICATION PACKET (COMMAND OR DATA) FORMULATED BY EXECUTING MACRO CODE WITHIN THE TEST ALGORITHMS. THE USE OF MACROS TO IMPLEMENT THE COMMUNICATIONS PROTOCOL SIMPLIFIES CONTEXT SWITCHING FROM UNIT TO UNIT BY NOT REQUIRING 8 SEPARATE DEVICE STACKS IN ADDITION TO THE SYSTEM STACK. THE TEST CODE RUNS AS A CO-ROUTINE WITH THE SCHEDULER,SO A TEST CODE PROGRAM COUNTER IS MAINTAINED FOR EACH UNIT "TSTPC(R5)".

THE TESTS ARE PERFORMED USING THE SPECIFIED ALGORITHM ON ALL DRIVE 0'S, THEN REPEAT THE TEST AFTER SWITCHING DRIVES, IF ANY DRIVE "1'S" WERE SELECTED.

FOLLOWING THE TRANSMISSION OF 1 PACKET TO EACH DEVICE (WITH XOFF PRECEEDING) THE UNITS ARE POLLED, AND THEIR ENTIRE RESPONSES

EVALUATED ROUND ROBIN. IF ANY ERROR INITIATES A RETRY, THE SCHEDULING PROCESS IS MODIFIED TO COMMUNICATE WITH ONLY 1 UNIT UNTIL COMPLETION OF THE RETRY PROCEDURE. THEN, A RETRY BY ANOTHER UNIT MAY PROCEED, OR THE SYSTEM CONTINUES NORMALLY.

THROUGHOUT THE PROGRAM, R5 POINTS TO ONE OF 8 POSSIBLE DATA STRUCTURES CONTAINING STATUS, TEST PARAMETERS, AND STATISTICAL INFORMATION FOR THE CURRENT UNIT, CALLED "UNIT'S DATA BLOCK". "START" CLEARS STATISTICS. "RESTART" AND "CONTINUE" DO NOT.

UPON OCCURANCE OF A FATAL ERROR, THAT UNIT IS DESCHEDULED (ABORTED) ALLOWING THE REMAINING (IF ANY) TO PROCEED WITH TESTING.

ERROR DESCRIPTIONS:

AN EXPLANATION OF THE EXTENDED ERROR INFORMATION FOLLOWS. SEE ALSO THE SECTION IN THIS LISTING SUBTITLED "ERROR MESSAGE DESCRIPTIONS".

BLOCK #:	THE RECORD NUMBER (1 PER 512. BYTES) IN LAST COMMAND PACK.
COMMAND:	THE MOST RECENT COMMAND PACKET OP CODE.
EXPCTD:	THE DATA PATTERN USED ON WRITE COMMAND AND FOR DATA COMPARE AFTER READ OP.
SUCCESS:	THE SUCCESS CODE RECEIVED IN END PACKET.
PAK SENT:	TYPE OF PACKET JUST SENT (0 FOR DATA; 1 FOR COMMAND)
FLAG RCVD:	FLAG BYTE OF PACKET CURRENTLY BEING CHECKED, OR 1ST BYTE OF RESPONSE.

SINCE IN MAINTENANCE MODE TU58 WILL SEND A BAD DATA PACK WITH A "DATA CHECK" SUCCESS STATUS IN THE FOLLOWING END PACK, THE HOST WILL, UPON CHECKING THOSE DATA PACK(S), DETERMINE "BAD DATA" IN PACKET ERROR FIRST, THEN INTERPRET THE SUCCESS CODE TO DIFFERENTIATE A COMMUNICATIONS GLITCH (GOOD SUCCESS) VS. TU 'DATA-CHECK' ERROR CODE. THIS WOULD SEEM TO RESULT IN TWO "ERROR" MESSAGES FOR ONE ERROR CONDITION, BUT ONLY THE SECOND ERROR MESSAGE WILL CONTAIN PERTINENT (NOT ZERO) ERROR NUMBER.

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE

PDP-11/LSI-11 CPU WITH AT LEAST 24K WORDS OF MEMORY AND CONSOLE DEVICE.

TU58 CONTROLLER AND DRIVE(S). DL, DLV, OR PDT COMPATABLE INTERFACE; AND REVISION "I" TU58 MICROCODE (OR LATER) ASSUMED.

1.2.2 SOFTWARE

THE PROGRAM IS REVISION D DIAGNOSTIC SUPERVISOR COMPATIBLE.
CONSULT XXDP+ USERS MANUAL FOR OPERATING INSTRUCTIONS.

1.3 RELATED DOCUMENTS AND STANDARDS

XXDP+ USERS MANUAL CHQUS

1.4 DIAGNOSTIC HIERARCY PREREQUISITES

APPROPRIATE INTERFACE DIAGNOSTICS MAY BE RUN TO ISOLATE INTERFACE ERRORS.

1.5 ASSUMPTIONS

SYSTEM HARDWARE OTHER THAN TU58(S) IS OPERATIONAL.

2.0 OPERATING INSTRUCTIONS

2.1 HOW TO RUN THIS DIAGNOSTIC

THE DIAGNOSTIC MAY BE INVOLKED WITH A 'START' RESPONSE TO THE SUPERVISOR PROMPT. 'STA'(CR) IS SUFFICIENT.
IF THE DEVICE IS NOT AT THE STANDARD ADDRESS AND VECTOR (176500, 300), THEN ANSWER "CHANGE HW?" WITH 'YES' INITIALLY TO SET UP HARDWARE CONFIGURATION TABLES FOR EACH UNIT. THAT INFORMATION IS:

TU58 CSR - ADDRESS OF RCSR OF DLV-11 OR OTHER INTERFACE BOARD.

VECTOR ADDR. - ADDRESS OF INTERRUPT VECTOR LOCATION.

PDT INTERFACE -- IS THE TU58 IN A PDT 11/130,
OR SYSTEM WHOSE BUFFERS ARE:
RCSR
RCDB (AND XMDB)
XMSR

TEST DRO - YES OR NO

TEST DR1 - YES OR NO

SUBSEQUENT RESPONSES TO "CHANGE HW?" MAY THEN BE "NO".

THE STANDARD ADDRESS AND VECTOR LOCATIONS FOR THE PDT 11/130

ARE 177170 AND 260 RESPECTIVELY.

THE SOFTWARE QUESTIONS ARE AS FOLLOWS:

NUMBER OF BLOCKS: TEST 5-8 -- ONE MAY SELECT A MINIMUM OF 8, TO A MAXIMUM OF 512 BLOCKS TO WRITE, READ; WRITE VERIFY; AND READ REDUCED, AS EXPLAINED IN SECTION 6.0.

ADD DR # TO DATA PATTERN -- FOR THOSE SAME READ AND WRITE TESTS 5-8, THE DRIVE NUMBER (0 OR 1) MAY BE ADDED TO DATA WRITTEN ON TAPE TO INSURE DRIVE SELECT BIT OPERATION.

STATISTICS PRINTED AT EOP -- SELECTS WHETHER OR NOT TO PRINT INFORMATION AT END OF PASS OR ^C. THESE STATISTICS MAY ALSO BE RE-TRIEVED WITH THE "PRI" COMMAND.

COMPARE DATA ON READ -- SELECTS WHETHER OR NOT TO DO A DATA COMPARE ON DATA PACKETS RE-CEIVED.

PRINT PACKET ON ERROR -- PRINTS 132. BYTE DATA PACKET ON A COMPARE ERROR,IF SELECTED.

ERRORS=DVC FATAL IF 'EVL' SET -- IF USER SETS EVL FLAG (EVALUATE) MODE), HRD OR SFT ERROR MESSAGES BECOME DVC FTL ERRORS AFTER THE NUMBER SPECIFIED IS EXCEEDED.

PRINT UNIT PROTOCOL SUMMARY (TEST 9) -- PRINTS A TABLE INDICATING THE PROTOCOL OF EACH UNIT.

3.0 ERROR INFORMATION

ERROR INFORMATION IS PROVIDED ON OCCURRANCE OF ERRORS AS OUTLINED IN SECTION 1.1.

4.0 PERFORMANCE AND PROGRESS REPORTS

STATISTICS ARE AVAILABLE PER SECTION 1.1 AT END OF PASS, CONTROL-C, OR UPON ENTERING A "PRI" COMMAND. THEY CONSIST OF # BLOCKS WRITTEN AND READ, # OF DATA ERRORS, HARD OR SOFT.

5.0 DEVICE INFORMATION TABLES

CONSULT SECTION SUBTITLED "DATA BLOCK FORMAT" FURTHER ON IN THIS LISTING.

6.0 TEST SUMMARIES

INIT: INIT IS SENT TO DEVICE IF:

1. INIT CODE IN SUPERVISOR IS EXECUTED

OR

2. INIT IS REQUESTED BY DEVICE AS A RESULT OF ERROR.

TEST 1: INITIATES FIRMWARE DIAGNOSTICS AT DEVICE LEVEL (SELF TEST)

TEST 2: SEEK TEST. SEEKS BOT ON BOTH TRACKS,THEN VERIFIES 60 IPS OPERATION TO SEEK EOT ON ON BOTH TRACKS,ENDING THEN AT BOT.

TEST 3: PERFORMS WRITE, THEN READ OF ADJACENT BLOCKS AT BOT WITH VARYING DATA, THEN SEEKS HALF WAY INTO REMAINING TAPE AND REPEATS THE ABOVE UNTIL EOT. THIS TEST IS IN 512 BYTE/BLOCK MODE.

TEST 4: PERFORMS WRITE, THEN READ OF ADJACENT BLOCKS AT BOT WITH VARYING DATA, THEN SEEKS HALF WAY INTO REMAINING TAPE AND REPEATS THE ABOVE UNTIL EOT. THIS TEST IS IN 128 BYTE/BLOCK MODE.

TESTS 5-8: READS OR WRITES BLOCK # AS DATA INTO SUCCESSIVE BLOCKS ON TAPE, THE LENGTH OF WHICH IS DETERMINED BY SOFTWARE QUESTION #1: DEFAULT IS SHORT TAPE (8.).MINIMUM, (8.) RESULTS IN TRANSFER OF 8. (OR 4 PER TRACK) 512. BYTE BLOCKS OF DATA PER READ (OR WRITE) OPERATION. THE ALGORITHM SWITCHES TRACKS REGARDLESS OF THE NUMBER BLOCKS SELECTED. DRIVE NUMBER IS ADDED TO RECORD AS DEFAULT,SO FOR TAPE INTERCHANGE TESTING, ANSWER (N) TO SOFTWARE (SW) QUESTION #2.

NOTE: THE AMOUNT OF TIME SPENT IN TESTS 5-8 IS QUITE LONG IF THE FULL TAPE (512.) IS SELECTED.

TEST 5: WRITE TAPE

TEST 6: READ TAPE

TEST 7: 'WRITE VERIFY' TAPE

TEST 8: READ MODIFIED THRESHOLD TAPE

TEST 9:

THE FIRST PART OF TEST 9 DETERMINES IF A UNIT IS CAPABLE OF MODIFIED RADIAL SERIAL PROTOCOL. THIS PART OF THE TEST IS WRITTEN USING RADIAL SERIAL PROTOCOL, AND DETERMINES THE PROTOCOL OF A UNIT BY SENDING THE TU58 A GET CHARACTERISTICS COMMAND AND MONITORING THE RESPONSE. IF THE TU58 RETURNS AN END PACKET IT IS A MODIFIED UNIT. IF THE TU58 RETURNS A DATA PACKET IT IS A NON-MODIFIED UNIT. NOTE, THE DATA PACKET RETURNED ON A GET CHARACTERISTICS COMMAND IS NOT NORMAL, RATHER IT CONSISTS OF A DATA PACKET THAT IS 28. BYTES PLUS AN END PACKET WHICH IS 14. BYTES. THE SECOND PART OF TEST 9 TESTS ONLY THOUGHS UNITS THAT ARE MODIFIED. THIS IS ACHIEVED BY LETTING NON-MODIFIED UNITS JUMP OVER CODE. IT WAS ASSUMED THAT IF A UNIT CAN READ,WRITE,ETC... WHEN OPERATING IN RSP, THEN IT CAN READ,WRITE,ETC... WHEN OPERATING IN MRSP. THEREFORE ALL THAT HAD TO BE TESTED WAS THE ABILITY OF MODIFIED UNIT TO BE ABLE TO SEND ONE BYTE AND WAIT FOR A CONTINUE FROM THE HOST BEFORE SENDING THE NEXT BYTE. A PROTOCOL SUMMARY OF THE UNITS IS ADVAILABLE BY ANSWERING YES (Y) TO SOFTWARE (SW) QUESTION # 5.

```
                                        .TITLE PROGRAM HEADER AND TABLES
                                        .SBTTL  PROGRAM HEADER

                                                .ENABL  ABS,AMA
                       002000                   .       =       2000
        002000                                  .NLIST  BEX
                                                BGNMOD

                                        ;++
                                        ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
                                        ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
                                        ;--

        002000                                  POINTER BGNRPT,BGNSW,BGNSFT,BGNAU,BGNDU,BGNSETUP

        002000                                  HEADER  CZTUU,E,0,3600.,1,PRI07
  (4)   002000                                                                          L$NAME::
  (4)   002000          103                                                                     .ASCII /C/
  (4)   002001          132                                                                     .ASCII /Z/
  (4)   002002          124                                                                     .ASCII /T/
  (4)   002003          125                                                                     .ASCII /U/
  (4)   002004          125                                                                     .ASCII /U/
  (6)   002005          000                                                                     .BYTE   0
  (6)   002006          000                                                                     .BYTE   0
  (5)   002007          000                                                                     .BYTE   0
  (5)   002010                                                                          L$REV::
  (4)   002010          105                                                                     .ASCII  /E/
  (5)   002011                                                                          L$DEPO::
  (4)   002011          060                                                                     .ASCII  /0/
  (5)   002012                                                                          L$UNIT::
  (4)   002012  000001                                                                          .WORD   T$PTHV
  (5)   002014                                                                          L$TIML::
  (4)   002014  007020                                                                          .WORD   3600.
  (5)   002016                                                                          L$HPCP::
  (4)   002016  041342                                                                          .WORD   L$HARD
  (5)   002020                                                                          L$SPCP::
  (4)   002020  041504                                                                          .WORD   L$SOFT
  (5)   002022                                                                          L$HPTP::
  (4)   002022  002176                                                                          .WORD   L$HW
  (5)   002024                                                                          L$SPTP::
  (4)   002024  002210                                                                          .WORD   L$SW
  (5)   002026                                                                          L$LADP::
  (4)   002026  042150                                                                          .WORD   L$LAST
  (5)   002030                                                                          L$STA::
  (4)   002030  000000                                                                          .WORD   0
  (5)   002032                                                                          L$CO::
  (4)   002032  000000                                                                          .WORD   0
  (5)   002034                                                                          L$DTYP::
  (4)   002034  000001                                                                          .WORD   1
  (5)   002036                                                                          L$APT::
  (4)   002036  000000                                                                          .WORD   0
  (5)   002040                                                                          L$DTP::
  (4)   002040  002152                                                                          .WORD   L$DISPAT
  (5)   002042                                                                          L$PRIO::
  (4)   002042  000340                                                                          .WORD   PRI07
```

```
(5)  002044
(4)  002044  000000
(5)  002046
(4)  002046  000000
(5)  002050
(4)  002050  003
(3)  002051  003
(5)  002052
(4)  002052  000000
(5)  002054  000000
(5)  002056
(4)  002056  000000
(5)  002060
(4)  002060  005510
(5)  002062
(4)  002062  015150
(5)  002064
(4)  002064  000000
(5)  002066
(4)  002066  000000
(5)  002070
(4)  002070  017302
(5)  002072
(4)  002072  017156
(5)  002074
(4)  002074  000000
(5)  002076
(4)  002076  002122
(5)  002100
(4)  002100  104035
(5)  002102
(4)  002102  000000
(5)  002104
(4)  002104  016164
(5)  002106
(4)  002106  017136
(5)  002110
(4)  002110  016754
(5)  002112
(4)  002112  002142
(5)  002114
(4)  002114  000000
(5)  002116
(4)  002116  000000
(5)  002120
(4)  002120  000000
3808
3809 002122                    DESCRIP <TU58 PERF EXER>
     002122
(3)  002122  052524  034065  050040
(3)  002130  051105  020106  054105
(3)  002136  051105     000
(2)          002142
```



```
LSENVI::
        .WORD   0
LSEXP1::
        .WORD   0
LSMREV::
        .BYTE   CSREVISI
        .BYTE   CSEDIT
LSEF::
        .WORD   0
        .WORD   0
LSSPC::
        .WORD   0
LSDEVP::
        .WORD   LSDVTYP
LSREPP::
        .WORD   LSRPT
LSEXP4::
        .WORD   0
LSEXP5::
        .WORD   0
LSAUT::
        .WORD   LSAU
LSDUT::
        .WORD   LSDU
LSLUN::
        .WORD   0
LSDESP::
        .WORD   LSDESC
LSLOAD::
        EMT     ESLOAD
LSETP::
        .WORD   0
LSICP::
        .WORD   LSINIT
LSCCP::
        .WORD   LSCLEAN
LSACP::
        .WORD   LSAUTO
LSPRT::
        .WORD   LSPROT
LSTEST::
        .WORD   0
LSDLY::
        .WORD   0
LSHIME::
        .WORD   0

LSDESC::
        .ASCIZ  /TU58 PE

        .EVEN
```

```
                                        ;++
                                        ;THE PROTECT TABLE IS USED BY THE MONITOR TO WARN THE OPERATOR WHEN HE
                                        ;TRIES TO TEST THE LOAD DEVICE.
                                        ;--

                                        BGNPROT

                                                                                LSPROT::
                                                .WORD   0               ;DEVICE CSR
                                                .WORD   -1              ;NO MASS BUS
                                                .WORD   -1              ;NO DRIVE
                                        ENDPROT
```

```
                                        .SBTTL  DISPATCH TABLE

                                        ;++
                                        ; THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
                                        ; IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
                                        ;--

                                               DISPATCH 9
                                                                                .WORD   9
                                                                      LSDISPATCH::
                                                                                .WORD   T1
                                                                                .WORD   T2
                                                                                .WORD   T3
                                                                                .WORD   T4
                                                                                .WORD   T5
                                                                                .WORD   T6
                                                                                .WORD   T7
                                                                                .WORD   T8
                                                                                .WORD   T9
```

```
                                        .SBTTL  DEFAULT HARDWARE P-TABLE

                                        ;++
                                        ; THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
                                        ; THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
                                        ; IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P-TABLES.
                                        ;--

                                                BGNHW   DFPTBL
                                                                                .WORD   L10001-L
                                                                        L$HW::
                                                                        DFPTBL::
                                                .WORD   176500          ;CSR ADDRESS
                                                .WORD   300             ;VECTOR ADDR.
                                                .WORD   3               ;TEST DRIVE ZERO AND ONE
                                                .WORD   0               ;NOT PDT TYPE INTERFACE

                                                ENDHW
                                                                        L10001:
```

```
                                .SBTTL  SOFTWARE P-TABLE

                                ;++
                                ; THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
                                ; PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
                                ;--

002206                                  BGNSW   SFPTBL
  (3) 002206  000007                                                    .WORD   L10002-L
  (3) 002210                                                          L$SW::
  (3) 002210                                                          SFPTBL::

002210  000010                  LENGTH: .WORD   8.               ;TAPE LENGTH
002212  000001                  STAEOP: .WORD   1                ;PRINT STATISTICS AT EOP
002214  000001                  PRBUF:  .WORD   1                ;PRINT DATA BUF ON COMP. ERROR
002216  000001                  CMPDAT: .WORD   1                ;COMPARE DATA
002220  000001                  DRVCHK: .WORD   1                ;ADD DR # TO DATA
002222  000001                  EVLTHR: .WORD   1                ;THRESHOLD FOR EVL TEST
002224  000000                  PPSOT9: .WORD   0                ;PRINT UNIT PROTOCOL SUMMARY (TST9)

002226                                  ENDSW
  (3) 002226                                                          L10002:

002226                                  ENDMOD
```

```
                                .TITLE GLOBAL AREAS
                                .SBTTL  GLOBAL EQUATES SECTION

002226                                  BGNMOD

                                ;++
                                ; THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
                                ; ARE USED IN MORE THAN ONE TEST.
                                ;--

002226                                  EQUALS
                                ;
                                ; BIT DIFINITIONS
                                ;
100000                          BIT15== 100000
040000                          BIT14== 40000
020000                          BIT13== 20000
010000                          BIT12== 10000
004000                          BIT11== 4000
002000                          BIT10== 2000
001000                          BIT09== 1000
000400                          BIT08== 400
000200                          BIT07== 200
000100                          BIT06== 100
000040                          BIT05== 40
000020                          BIT04== 20
000010                          BIT03== 10
000004                          BIT02== 4
000002                          BIT01== 2
000001                          BIT00== 1
                                ;
001000                          BIT9==  BIT09
000400                          BIT8==  BIT08
000200                          BIT7==  BIT07
000100                          BIT6==  BIT06
000040                          BIT5==  BIT05
000020                          BIT4==  BIT04
000010                          BIT3==  BIT03
000004                          BIT2==  BIT02
000002                          BIT1==  BIT01
000001                          BIT0==  BIT00
                                ;
                                ; EVENT FLAG DEFINITIONS
                                ;   EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION
                                ;
000040                          EF.START==      32.             ; START COMMAND WAS ISSUED
000037                          EF.RESTART==    31.             ; RESTART COMMAND WAS ISSUED
000036                          EF.CONTINUE==   30.             ; CONTINUE COMMAND WAS ISSUED
000035                          EF.NEW==        29.             ; A NEW PASS HAS BEEN STARTED
000034                          EF.PWR==        28.             ; A POWER-FAIL/POWER-UP OCCURRED
                                ;
                                ; PRIORITY LEVEL DEFINITIONS
                                ;
000340                          PRI07== 340
```

```
(1)           000300                        PRI06== 300
(1)           000240                        PRI05== 240
(1)           000200                        PRI04== 200
(1)           000140                        PRI03== 140
(1)           000100                        PRI02== 100
(1)           000040                        PRI01== 40
(1)           000000                        PRI00== 0
(1)                                         ;
(1)                                         ;OPERATOR FLAG BITS
(1)                                         ;
(1)           000004                        EVL==         4
(1)           000010                        LOT==        10
(1)           000020                        ADR==        20
(1)           000040                        IDU==        40
(1)           000100                        ISR==       100
(1)           000200                        UAM==       200
(1)           000400                        BOE==       400
(1)           001000                        PNT==      1000
(1)           002000                        PRI==      2000
(1)           004000                        IXE==      4000
(1)           010000                        IBE==     10000
(1)           020000                        IER==     20000
(1)           040000                        LOE==     40000
(1)           100000                        HOE==    100000
3954
```

```
                        .SBTTL  ERROR CODE EQUATES

                        ;THE ERROR CODE OFFSET VALUES :
                        ;USED BY ROUTINE 'LOG' TO INDEX (BY R5) INTO DEVICE'S DATA BLOCK AND
                        ;INCREMENT STATISTICS.

000002                  SFTRD   ==      2
000004                  SFTWR   ==      4
000006                  RCINIT  ==      6
000010                  OTL     ==      8.
000012                  OVRN    ==      10.
000014                  BDCOM   ==      12.
000016                  HRDRD   ==      14.
000020                  HRDWR   ==      16.
000022                  BDCHK   ==      18.
000024                  SKERR   ==      20.
000026                  WRLOCK  ==      22.
000030                  NOMOT   ==      24.
000032                  CNINIT  ==      26.
000034                  PARTL   ==      28.
000036                  NOUNIT  ==      30.
000040                  CMNDER  ==      32.
000042                  RECERR  ==      34.
000044                  SLFER   ==      36.
000046                  SUCOTL  ==      38.
000050                  TORCVB  ==      40.
000054                  NCART   ==      44.
000056                  TOSNDB  ==      46.

                        ;               IN ADDITION, SYSTEM SETUP OR RUNTIME ERRORS ARE:
                        ;       100.    -       ALL UNITS ABORTED
                        ;       101.    -       MORE THAN 8. UNITS (16 DRIVES) REQUESTED
                        ;       102.    -       NEITHER DRIVE SELECTED FOR THIS CONTROLLER
                        ;               ALL THE ABOVE ARE CLASSIFIED AS SYSTEM FATAL
```

```
                        .SBTTL  GENERAL EQUATES
                        ;RADIAL SERIAL CODES:
                        ;-----------------------------------------------------------------------
                        ;THE FLAG BYTE CODES ARE:
000002                          RSCMND == 2             ;"COMMAND" PACKET
000020                          RSCONT == 20            ;"CONTINUE" SINGLE BYTE
000020                          RSXON  == 20            ;"XON" SINGLE BYTE
000023                          RSXOFF == 23            ;"XOFF" SINGLE BYTE
000004                          RSINIT == 4             ;"INIT" SINGLE BYTE
000001                          RSDATA == 1             ;"DATA" PACKET
000002                          RSEND  == RSCMND        ;"END" PACKET FLAG IS "COMMAND"
                        ;-----------------------------------------------------------------------
                        ;END PACK SIZE:
000016                          RSNDSZ == 14.           ;TOTAL BYTES IN COMMAND PACKET
                        ;MESSAGE PACK SIZE:
000012                          RSMSIZ == 12            ;10. BYTES FOR BYTE COUNT INSIDE CMND PACK
                        ;DATA PACK SIZE:
000204                          RSDASZ == 132.          ;TOTAL BYTES IN DATA PACKET
                        ;DATA + END PACK SIZE:
000222                          RSDNSZ == RSDASZ+RSNDSZ
                        ;GET CHARACTERISTICS DATA PACKET SIZE
000034                          RSGCDP == 28.           ;TOTAL BYTES FOR GET CHAR DATA PACKET
                                                        ;MINUS THE END PACKET
000016                          RSSNSZ == RSMSIZ + 4            ;SIZE FOR SENDING COMMAND PACK
001036                          RCBFSZ== 4+RSDASZ+RSNDSZ        ;4 DATA PAKS AND END PACK
                                                                ;IS SIZE OF RCV BUFFERS
                        ;-----------------------------------------------------------------------
                        ;       THE OP CODES ARE:
000100                          RSSEND == 100           ;END PACK DESCRIPTOR
000003                          RSSWR  == 3             ;WRITE
000002                          RSSRD == 2              ;READ
000005                          RSSSEK == 5             ;SEEK
000012                          RSSGET == 12            ;GET CHARACTERISTICS
000000                          RSSNOP == 0             ;NO-OPERATION
000001                          RSSNIT == 1             ;INITIALIZE
000007                          RSSSLF == 7             ;SELF TEST
                        ;-----------------------------------------------------------------------
                        ;THE SUCCESS CODES ARE:
177720                  ESABO   ==-48.          ;BAD COMMAND FROM HOST
177767                  ESNCRT  ==-9.           ;NO CARTRIDGE
177770                  ESNONX  ==-8.           ;NO DRIVE
000000                  ESOK    ==0             ;OP COMPLETE SUCCESS
177776                  ESPART  ==-2            ;PARTIAL OP
177740                  ESSK    ==-32.          ;SEEK ERROR
000001                  ESTRY   ==1             ;RETRYS OCCURRED
177765                  ESWLOC  ==-11.          ;WRITE PROTECTED
177737                  ESNOMO  ==-33.          ;MOTOR STOPPED
177720                  ESCMD   ==-48.          ;COMMAND ERROR
177711                  ESREC   ==-55.          ;BAD RECORD NUMBER.
177757                  ESCKS   ==-17.          ;TU CHKSUM ERROR
177777                  ESSLF   ==-1.           ;SELF TEST ERROR
177757                  ESCKSM=ESCKS
177757                  ESWR=ESCKS
177757                  ESRD=ESCKS
                        ;-----------------------------------------------------------------------
```

```
                                .SBTTL  ERROR MESSAGE DESCRIPTIONS

                                ;THE TABLE OF ERROR MESSAGES (ADDRESSES).  ABNDX(R5) CONTAINS THE OFFSET
                                ;OF THE REASON.  IT'S ABSOLUTE ADDRESS IS RSNTAB + ABNDX(R5).

002226  002322                  RSNTAB: MSNLOG
002230  003054                          MSSFRD
002232  003114                          MSSFWR
002234  002536                          MSRNIT
002236  003000                          MSQRSP
002240  003260                          MSOVRN
002242  002404                          MSCOM
002244  003154                          MSHDRD
002246  003216                          MSHDWR
002250  002556                          MSHCHK
002252  002306                          MSSKER
002254  002514                          MSWPRO
002256  002446                          MSNOMO
002260  002620                          MSNIT
002262  002634                          MSPART
002264  002656                          MSUNIT
002266  002704                          MSCMD
002270  002720                          MSREC
002272  002364                          MSSELF
002274  002740                          MSWRSP
002276  002764                          MSNRSP
002300  002322                          MSNLOG
002302  002464                          MSNOTP
002304  003032                          MSTOSN
```

```
                                ;HERE ARE THE MESSAGES PROPER:

002306 042523 045505 042440     MSSKER:: .ASCIZ /SEEK ERROR/            ;DEVICE COULD NOT READ HEADER
       002322                            .EVEN
002322 054523 052123 046505     MSNLOG:: .ASCIZ /SYSTEM ERROR/          ;DIAGNOSTIC HUNG. BETTER RE-BOOT
       002340                            .EVEN
002340 040502 020104 040504     MSBDA::  .ASCIZ  /BAD DATA IN PACKET/   ;HOST DATA CHECK FOUND ERROR, DEVICE MAY
       002364                            .EVEN                          ;HAVE READ CORRECTLY.
002364 042523 043114 052040     MSSELF:: .ASCIZ /SELF TEST ERROR/       ;MICRO DIAGNOSTIC FAILED, BUT DEVICE COU
                                         .EVEN                          ;SEND AN END PACKET.
002404 040502 020104 040504     MSCOM::  .ASCIZ  /BAD DATA W-O DATA CHECK ERR AT TU/  ;PREVIOUS DATA CHECK
                                         .EVEN                          ;ERROR NOT DUE TO DEVICE READ OPERATION
002446 047515 047524 020122     MSNOMO:: .ASCIZ /MOTOR STOPPED/         ;DEVICE COULD NOT GET ANY MEANINGFUL SIG
                                         .EVEN                          ;FROM TAPE.
002464 040503 052122 044522     MSNOTP:: .ASCIZ /CARTRIDGE NOT IN PLACE/ ;NO MEDIA OR BAD SWITCH
       002514                            .EVEN
002514 051127 052111 020105     MSWPRO:: .ASCIZ /WRITE PROTECTION/      ;CARTRIDGE WRITE PROTECT TAB MISSING OR
       002536                            .EVEN                          ;SWITCH BAD
002536 042522 044503 053105     MSRNIT:: .ASCIZ /RECIEVING INIT/        ;DEVICE SENT INIT REQUEST
       002556                            .EVEN
002556 047510 052123 043040     MSHCHK:: .ASCIZ /HOST FOUND PACKET CHECKSUM ERROR/  ;DEVICE SENT PACK WITH
       002620                            .EVEN                          ;BAD CHECKSUM
002620 040503 023516 020124     MSNIT::  .ASCIZ  /CAN'T INIT/           ;DEVICE SENT BYTE OTHER THAN "CONTINUE"
       002634                            .EVEN                          ;DURING INITIALIZATION
002634 040520 052122 040511     MSPART:: .ASCIZ /PARTIAL OPERATION/     ;END OF MEDIUM ENCOUNTERED
                                         .EVEN
002656 047042 047117 042455     MSUNIT:: .ASCIZ /"NON-EXISTENT" DRIVE/  ;DEVICE RECV'D TOO LARGE DRIVE NUMBER
       002704                            .EVEN
002704 040502 020104 047503     MSCMD::  .ASCIZ  /BAD COMMAND/          ;DEVICE COULD NOT UNDERSTAND HOST
                                         .EVEN
002720 040502 020104 042522     MSREC::  .ASCIZ  /BAD RECORD NO./       ;DEVICE RECV'D TOO LARGE A RECORD NUMBER
       002740                            .EVEN
002740 051127 047117 020107     MSWRSP:: .ASCIZ /WRONG SUCCESS CODE/    ;HOST COULD NOT DECIPHER CODE IN END PAC
       002764                            .EVEN
002764 047516 051040 051505     MSNRSP:: .ASCIZ /NO RESPONSE/           ;TIME OUT WAITING FOR BYTE IN RCV BUF ON
                                         .EVEN
003000 047111 042504 044503     MSQRSP:: .ASCIZ \INDECIPHERABLE FLAG BYTE\  ;HOST COULD NOT UNDERSTAND 1ST BYTE
       003032                            .EVEN                          ;RESPONSE FROM TU AS PROPER PROTOCOL
003032 044524 042515 047440     MSTOSN:: .ASCIZ /TIME OUT ON SEND/      ;DLV 'READY' NEVER WENT HIGH
       003054                            .EVEN
003054 042522 047503 027126     MSSFRD:: .ASCIZ /RECOV. DATA CHECK ERR ON RD OP/  ;TU58 RESPONDED WITH 'DATA-CHE
       003114                            .EVEN                          ;ERROR ON READ OP. ;HOST RETRY(S) SUCCES
003114 042522 047503 027126     MSSFWR:: .ASCIZ /RECOV. DATA CHECK ERR ON WR OP/  ;SAME BUT WR OR WR VERIFY OPER
       003154                            .EVEN
003154 047125 042522 047503     MSHDRD:: .ASCIZ /UNRECOV. DATA CHECK ERR ON RD OP/  ;TU58 RESPONDED WITH 'DATA-C
       003216                            .EVEN                          ;ERROR ON READ OP.  ;RETRIES UNSUCCESSFU
003216 047125 042522 047503     MSHDWR:: .ASCIZ /UNRECOV. DATA CHECK ERR ON WR OP/  ;SAME BUT WR OPERATION
       003260                            .EVEN
003260 046104 020126 051105     MSOVRN:: .ASCIZ /DLV ERROR IN RECEIVE/  ;DLV ERROR (THE CONTENTS PRINTED OUT)
       003306                            .EVEN
```

```
                                .SBTTL  MISC STORAGE AND EQUATES

 003306  000000                 SYSTAT:: .WORD                         ;SYSTEM STATUS WORD
                                                                       ;BIT7-BIT15 = 1ST BYTE OF PACK RCVD
                                                                       ;BIT05 = NOT USED
                                                                       ;BIT04 = NOT USED
                                                                       ;BIT03 = NOT USED
                                                                       ;BIT02 =RETRY BAD FLAG BYTE
                                                                       ;BIT01 = RETRY OCCURRING
                                                                       ;BIT00 = CHKSUM IS ODD (TEMP STORAGE)

 003310  000000                 TAPLEN:: .WORD                         ;# RECORDS
 003312  000000                 DEVPTR:: .WORD                         ;->NEXT UNIT ADDRESS
 003314  000000                 RCFLG::  .WORD                         ;TEMP STORE FOR GTBYTE
 003316  000000                 RCBCNT:: .WORD                         ;TEMP STORAGE FOR GTBYTE
 003320  000004                 FTLNM:   .WORD  4                      ;NUMBER OF TRIES BEFORE ABORT
 003322  000000                 DONE::   .WORD                         ;1 IF TEST ALGORITHM COMPLETE
 003324  000000                 IDPTR::  .WORD                         ;PTR IN TESTID MACRO
 003326  000000                 TSTTOP:  .WORD                         ;POINTER TO CURRENT TEST
 003330  000010                 MXRTRY:: .WORD  8.                     ;NUMBER OF RETRIES, BEFORE "HARD" ERROR
 003332  000000                 BLKER::  .WORD                         ;RECORD # FOR ERROR REPORT
 003334  000000                 SECREC:: .WORD                         ;RECORD # TO START AFTER SWITCHING TRACKS
 003336  000000                 PRNSIZ:: .WORD                         ;SIZE OF PACK FOR "PRNPAK"
 003340  000000                 ALLGON:: .WORD                         ;SECONDARY DON'T PRINT STATISTICS FLAG.
 003342  000000                 TEST9::  .WORD                         ;***** FLAG THAT INDICATES IF IN TST 9



                                ;               TIME OUT CONSTANTS:

 003344  177777                                 CSNRDY:: -1            ;TIME OUT ON SEND DELAY (DLV NOT READY)
 003346  000226                                 CSRCVB:: 150.          ;45 SEC. MULTIPLIER  REWIND TIME
                                                                       ;CAN BE OPTIMISED PER CPU
```

```
                                .SBTTL  DATA BLOCK FORMAT
                                ;-------------------------------------------------------------------------
                                ;R5 --> TOP OF 1 OF THE 8 DATA BLOCKS (1 PER UNIT) DURING EXECUTION
                                ;@R5 IS THE STATUS WORD CONTAINING:
                                                        ;BIT15 = ABORTED
                                                        ;BIT14 = SEND "BREAK"
                                                        ;BIT13 = RETRY FLAG BYTE ERROR (DATA PACKS)
                                                        ;BIT12 = TEMP STOR WRITE MACRO
                                                        ;BIT11 = UNIT NOT BEING TESTED
                                                        ;BIT10 = RETRYING DATA ERROR
                                                        ;BIT9  = TU58 CHKSUM ERROR
                                                        ;BIT8  = RD/WR OPERATION
                                                        ;BIT7  = NORMAL/REDUCED THRESHOLD (MACROS)
                                                        ;BIT6  = HOST DATA COMPARE ERROR
                                                        ;BIT5  = WR VERIFY OPERATION
                                                        ;BIT4  = TYPE OF PAK SENT ODATA 1CMD
                                                        ;BIT3  = RETRY FLAG BYTE ERR.(SEND COMMAND PACK)
                                                        ;BIT0,1,2=UNIT NO.
000000                  STATUS  ==      0.              ;DEVICE STATE
000002                  RETRY   ==      2.              ;# OF RETRIES
000004                  ABNDX   ==      4.              ;ERROR NUMBER FOR LOG
                        ;R0                             ;STORAGE FOR REGISTERS USED IN TEST BODY
                        ;R1                             ;STORED WITH SWAPOW
                        ;R2                             ;RETRIEVED WITH SWAPIN
                        ;R3                             ;
                        ;R4                             ;
000020                  TSTPC   ==      16.             ;POINTER TO NEXT EXECUTABLE TEST INST.
000022                  RCSR    ==      18.             ;DLV RCV STATUS ADDRESS
000024                  RCDB    ==      20.             ;DLV RCV DATA ADDRESS
000026                  XMSR    ==      22.             ;DLV SND STATUS ADDRESS
000030                  XMDB    ==      24.             ;DLV SND DATA ADDRESS
000032                  XSPKNM  ==      26.             ;THE NUMBER OF PACKETS TO RECEIVE
000034                  XSFLG   ==      28.             ;THE EXPECTED FLAG OF 1ST PACKET
000036                  XSCNT   ==      30.             ;THE EXPECTED COUNT OF 1ST PACKET
                                                        ;FOR MULTIPLE PACKET RECIEVES (MAX.4)
                        .       BLKW    8.              ;CONSECUTIVE XSFLGS AND XSCNTS
000060                  DR      ==      48.             ;DR==0 OR 1; BIT8,9 DRIVE SELECTED BY OPERATOR
000062                  TRK     ==      50.             ;COUNTER FOR TRACK NUMBER
000064                  REC     ==      52.             ;RECORD (BLOCK #)

000066                  TMP     ==      54.             ;TEST MACRO REGISTER
000070                  SNDCNT  ==      56.             ;THE # OF BYTES FOR SENDING PACKET
000072                  PATTEN  ==      58.             ;DATA PATTERN-LOWER BYTE USED
000074                  DLV     ==      60.             ;CONTENTS OF RCDB ON DLV ERROR
000076                  SUCCS   ==      62.             ;SUCCESS CODE OF LAST END PACKET
000100                  CMDSNT  ==      64.             ;TYPE OF COMMAND CURRENT IN EVEN BYTE; BIT15==VE

000102                  RCVBUF  ==      66.             ;POINTER TO 542. BYTE BUFFER (4 DATA PAKS + END
000104                  PKPTR   ==      68.             ;POINTER TO TOP OF PACKET
000106                  XSPTR   ==      70.             ;POINTER TO CURRENTLY USED XSFLG OR XSCNT
000110                  WRTN0   ==      72.             ;THE # OF 512. BYTE BLOCKS WRITTEN DR0
000112                  WRTN1   ==      74.             ;THE # OF 512. BYTE BLOCKS WRITTEN DR1
000114                  RDN0    ==      76.             ;THE # OF 512. BYTE BLOCKS READ DR0
000116                  RDN1    ==      78.             ;THE # OF 512. BYTE BLOCKS READ DR1
```

```
                                ;AND THE ERROR LOG...                  +-----------+
                                ;SPLIT INTO A BYTE PER DRIVE:          ! DR1 ! DR0 !
                                ;                                      +-----------+

                                ;-------------------------------------------------------------------
                                ;OFFSET IN DATA BLOCK           ;ERROR TYPE     ;ERRCODE;MSG CODE;SUC. CODE
                                ;-------------------------------------------------------------------

000120                          LGOFST  ==      80.             ;**RESERVED**
000122                          SOFTR   ==      82.             ;SOFT READ      ;SFTRD  ;MSSFRD ;ESCKSM
000124                          SOFTW   ==      84.             ;SOFT WRITE     ;SFTWR  ;MSSFWR ;ESSKSM
                                ;       WORD                    ;RECIEVED INIT  ;RCINIT ;MSRNIT ;******
                                ;       WORD                    ;BAD FLAG BYTE  ;OTL    ;MSQRSP ;******

                                ;THEN THOSE CODES WHICH HAVE N TRIES BEFORE ABORT

000132                          T4TRY   ==      90.             ;DLV ERROR      ;OVRN   ;MSOVRN ;******
000134                          BDATA   ==      92.             ;BAD DATA       ;BDCOM  ;MSDATA ;******
000136                          HARDR   ==      94.             ;HARD READ      ;HRDRD  ;MSHDRD ;ESCKSM
000140                          HARDW   ==      96.             ;HARD WRITE     ;HRDWR  ;MSHDWR ;ESCKSM
                                ;       WORD                    ; CHKSM AT HOST ;BDCHK  ;MSHCHK ;******
                                ;       WORD                    ;SEEK ERROR TOTAL;SKERR ;MSSKER ;******
000146                          T1TRY   ==      102.            ;WRITE PROTECT  ;WRLOCK ;MSWPRO ;ESWLOC
                                ;       WORD                    ;NO MOTOR       ;NOMOT  ;MSNOMO ;ESNOMO
                                ;       WORD                    ;CANT INIT      ;CNINIT ;MSNIT  ;******
                                ;       WORD                    ;PARTIAL OP     ;PARTL  ;MSPART ;ESPART
                                ;       WORD                    ;NO UNIT        ;NOUNIT ;MSUNIT ;ESNONX
                                ;       WORD                    ;COMMAND ERROR  ;CMNDER ;MSCMD  ;ESCMD
                                ;       WORD                    ;BAD RECORD NO.;RECERR  ;MSREC  ;ESREC
                                ;       WORD                    ;SELF TEST ERROR;SLFER  ;MSSELF ;******
                                ;       WORD                    ;WRONG SUC.CODE ;SUCOTL ;MSWRSP ;******
                                ;       WORD                    ;NO RESPONSE    ;TORCVB ;MSNRSP ;******
                                ;       WORD                    ;**RESERVED**
                                ;       WORD                    ;NO CARTRIDGE   ;NOCART ;MSNOTP ;ESNCRT
                                ;       WORD                    ;TIME OUT  SEND;TOSNDB  ;MSTOSN ;******

000202                          BLKEND  ==      130.            ;OFFSET OF END OF STATISTICS (RESERVED)
000204                          TUVECT  ==      132.            ;VECTOR ADDRESS
000206                          SAVCNT  ==      134.            ;BYTE COUNT SAVED DURING RETRY ON WRITE OPERATIO
000210                          MRSP    ==      136.            ;***** FLAG INDICATING MRSP
000212                          BLKSIZ  ==      138.            ;** RESERVED **
                                ;---------------- -------------------------------------------------
```

```
                        .SBTTL  DEVICE DATA BLOCK ALLOCATION

                        ;TABLE OF DEVICE DATA BLOCK ADDRESSES

003350  003370  BLKTBL::        .WORD   DEV0
003352  003602                  .WORD   DEV1
003354  004014                  .WORD   DEV2
003356  004226                  .WORD   DEV3
003360  004440                  .WORD   DEV4
003362  004652                  .WORD   DEV5
003364  005064                  .WORD   DEV6
003366  005276  LSTDEV::        .WORD   DEV7

                        ;AND STORAGE FOR EACH:

003370  000212  DEV0:           .BLKB   BLKSIZ
003602  000212  DEV1:           .BLKB   BLKSIZ
004014  000212  DEV2:           .BLKB   BLKSIZ
004226  000212  DEV3:           .BLKB   BLKSIZ
004440  000212  DEV4:           .BLKB   BLKSIZ
004652  000212  DEV5:           .BLKB   BLKSIZ
005064  000212  DEV6:           .BLKB   BLKSIZ
005276  000212  DEV7:           .BLKB   BLKSIZ
```

```
                        .SBTTL  GLOBAL TEXT SECTION

                        ;
                        ; NAMES OF DEVICES SUPPORTED BY PROGRAM
                        ;
                              DEVTYP  <TU58 CONTROLLER>

005510
005510
005510  052524  034065  041440                          LSDVTYP::
005516  047117  051124  046117                                .ASCIZ  /TU58 CO
005524  042514  000122
                                                          .EVEN
```

```
                        .SBTTL  SYSTEM MACRO DEFINITIONS

                                .MACRO  PUSH    ,REG

                                .NLIST
                                .LIST ME
                                .LIST
                                                        MOV     REG,-(SP)


                                .NLIST
                                .NLIST ME
                                .LIST
                                .ENDM

                                .MACRO  POP,REG

                                .NLIST
                                .LIST ME
                                .LIST
                                                        MOV     (SP)+,REG

                                .NLIST
                                .NLIST ME
                                .LIST
                                .ENDM

                        ;++
                        ;THE MACRO 'SWAPIN' RETRIEVES THE TEST REGISTERS WHICH WERE SAVED
                        ;IN THE DEVICE DATA BLOCK.
                        ;--

                                .MACRO  SWAPIN

                                .NLIST
                                .LIST ME
                                .LIST
                                                        MOV     6.(R5),R0
                                                        MOV     8.(R5),R1
                                                        MOV     10.(R5),R2
                                                        MOV     12.(R5),R3
                                                        MOV     14.(R5),R4

                                .NLIST
                                .NLIST ME
                                .LIST
                                .ENDM

                        ;++
                        ;THE MACRO 'SWAPOW' SAVES THE CURRENT STATE OF THE UNIT IN THE DRIVE
                        ;DATA BLOCK IN SO THAT THE SCHEDULER MAY 'SWAPIN' ANOTHER UNIT.
                        ;--

                                .MACRO  SWAPOW

                                .NLIST
```

```
                                        .LIST ME
                                        .LIST

                                                MOV     R0,6.(R5)
                                                MOV     R1,8.(R5)
                                                MOV     R2,10.(R5)
                                                MOV     R3,12.(R5)
                                                MOV     R4,14.(R5)

                                        .NLIST
                                        .NLIST ME
                                        .LIST
                                        .ENDM
```

```
4436                    ;++
4437                    ;THE WRITE MACRO IMPLEMENTS THE COMPLETE PROTOCOL NECESSARY TO BUILD
4438                    ;A COMMAND PACKET AND SUBSEQUENT DATA PACKETS (UNTIL THE BYTE COUNT
4439                    ;(BCNT) IS SATISFIED).
4440                    ;
4441                    ;SETS UP THE EXPECTED PROTOCOL RESPONSES:  THE NUMBER OF PACKETS
4442                    ;(XSPKNM) AND THEIR FLAG BYTES AND COUNTS (XSFLG, XSCNT).  CALLS
4443                    ;'RSVP' TO SEND EACH PACKET, AND 'CHKSUM' TO CALC. THE PACKET
4444                    ;CHECKSUM.
4445                    ;
4446                    ; INPUTS  - DEVICE BLOCK @R5
4447                    ;           TRBUF - BUFFER ADDRESS
4448                    ;           UNIT'S TEST REGISTERS FROM 'SWAPIN'
4449                    ; OUTPUTS - SNDCNT(R5) = # OF BYTES TO SEND
4450                    ;           XSPKNM = # OF PACKETS EXPECTED
4451                    ;           XSFLG = FLAG BYTE OF 1ST PACKET
4452                    ;           XSCNT = BYTE COUNT OF 1ST PACKET
4453                    ;             . ***
4454                    ;             .   *  SUBSEQUENT XSFLGS
4455                    ;             .    >
4456                    ;             .   *  AND XSCNTS
4457                    ;             . ***
4458                    ;--
4459
4460                            .MACRO  TUWRIT  PTRN,REC,BCNT,DR,VER,?A,?B,?C,?D,?E,?F,?G,?H,?T
4461
4462                            .NLIST
4463                            .LIST ME
4464                            .LIST
4465
4466                    T:      MOV     #TRBUF,R0       ;MAKE COMMAND PACKET:
4467                            MOVB    #RSCMND,@R0     ;COMMAND FLAG
4468                            MOVB    #RSMSIZ,1(R0)   ;THIS SIZE
4469                            MOVB    #RSSWR,2(R0)    ;INSERT OP CODE-WRITE
4470                            MOVB    VER,3.(R0)      ;VERIFY (1 OR 0)
4471                            MOVB    DR,4.(R0)       ;DRIVE #
4472                            MOVB    #020,5.(R0)     ;MAINTENANCE MODE SWITCH
4473                            CLR     6.(R0)          ;NO SEQUENCE #
4474                            MOV     BCNT,8.(R0)     ;TOTAL COUNT TO WRITE
4475                            MOV     REC,10.(R0)     ;AT  RECORD N
4476                            MOV     #RSMSIZ,R1      ;THE PACKET SIZE PLUS+2
4477                            TST     (R1)+           ;(FLAG AND COUNT) INTO R
4478                            MOV     #RSSNSZ,SNDCNT(R5)  ;LOAD THE  SIZE TO S
4479                            CALL    CHKSUM          ;R0 --> R1=COUNT
4480                            MOV     R1,(R0)         ;PUT CHKSUM IN PACKET
4481                                                    ;SET UP EXPECTATIONS:
4482                            MOV     #RSCONT,XSFLG(R5)   ;THE FLAG
4483                            MOV     #1,XSCNT(R5)    ;THE COUNT
4484                            MOV     #1,XSPKNM(R5)   ;THE # PACKETS EXPECTED
4485                            MOV     BCNT,R2         ;GET # OF DATA BYTES
4486                            CALL    RSVP            ;SEND (AND RETURN TO SCH
4487                            BIT     #BIT3,@R5       ;FLAG BYTE ERROR?
4488                            BNE     T               ;YES
4489                            BIC     #BIT12,@R5      ;FLAG FOR LAST PACKET
4490                    A:      MOV     #TRBUF,R0       ;POINT TO TOP OF BUFFER
4491                            CMP     R2,#128.        ;START DATA PACKET(S)
                                BHI     B               ;BCNT > 128.!
```

D 3

SEQ 0029

```
                        MOV     R2,R1           ;BCNT<128.
                        BIS     #BIT12,@R5      ;SO LAST PACKET NOW
                        BR      C               ;USE REMAINING COUNT
                B:      MOV     #128.,R1        ;USE 128. BYTES
                C:      MOVB    R1,1(R0)        ;COPY COUNT TO BUFFER
                        MOV     R1,R3           ;R3=COUNTER TO LOAD BUFF
                        MOVB    #RSDATA,@R0     ;FLAG FIRST
                        TST     (R0)+           ;SKIP COUNT
                D:      MOVB    PTRN,(R0)+      ;INSERT DATA
                        DEC     R3              ;MORE?
                        BHI     D               ;YES
                        MOV     #TRBUF,R0       ;-->TOP AGAIN
                        MOVB    1(R0),R1        ;GET COUNT
                        BIC     #177400,R1      ;ZERO SIGN EXTEND
                        MOV     R1,SNDCNT(R5)   ;HOW MANY TO SEND PLUS
                        ADD     #4,SNDCNT(R5)   ;FLAG,COUNT,CHKSUM
                        ADD     #2,R1           ;COMPENSATE FOR FLAG + C
                        CALL    CHKSUM          ;FOR CHECKSUM CALC.
                        MOVB    R1,(R0)+        ;CHKSUM INTO PACKET
                        SWAB    R1              ;EVEN ON AN ODD
                        MOVB    R1,(R0)+        ;BYTE BOUNDARY
                        BIT     #BIT12,@R5      ;LAST DATA PACKET?
                        BEQ     E               ;NO
                        MOV     #RSEND,XSFLG(R5)  ;YES-EXPECT 'END'
                        MOV     #RSNDSZ,XSCNT(R5)  ;OF THIS SIZE
                        MOV     #1,XSPKNM(R5)   ;AND 1 PACKET
                        BR      F               ;SEND
                E:      MOV     #RSCONT,XSFLG(R5)  ;(NOT LAST), EXPECT '
                        MOV     #1,XSCNT(R5)    ;AND 1 BYTE
                        MOV     #1,XSPKNM(R5)   ;AND 1 PACKET
                F:      CALL    RSVP            ;SEND PACKET
                                                ;AND RETURN TO SCHEDULER
                        BIT     #BIT3,@R5       ;FLAG BYTE RETRY?
                        BNE     T               ;YES
                        BIT     #BIT10,@R5      ;RETRY DATA ERROR?
                        BNE     G               ;YES
                        SUB     #128.,R2        ;NO, MORE DATA TO SEND?
                        BHI     A               ;YES
                        BR      H               ;NO
                G:      TURTRY  REC,BCNT,DR     ;RETRY HERE
                        BIT     #BIT10!BIT3,@R5 ;RETRY AGAIN?
                        BNE     G               ;YES
                H:      NOP                     ;DONE


.NLIST
.NLIST ME
.LIST
.ENDM
```

```
;++
;THE SEEK MACRO IMPLIMENTS THE COMPLETE PROTOCOL TO INITIATE A SEEK
;SEQUENCE.
;
;SETS UP THE EXPECTED PROTOCOL RESPONSES:  THE NUMBER OF PACKETS
;(XSPKNM) AND THEIR FLAG BYTES AND COUNTS (XSFLG, XSCNT).  CALLS
;'RSVP' TO SEND EACH PACKET, AND 'CHKSUM' TO CALC. THE PACKET
;CHECKSUM.
;
; INPUTS  - DEVICE BLOCK @R5
;           UNITS TEST REGISTERS FROM SWAPIN
;           TRBUF - BUFFER ADDRESS
;
; OUTPUTS -
;           XSPKNM = # OF PACKETS EXPECTED
;           XSFLG = FLAG BYTE OF 1ST PACKET
;           XSCNT = BYTE COUNT OF 1ST PACKET
;             . ***
;             .   *  SUBSEQUENT XSFLGS
;             .  >
;             .  *   AND XSCNTS
;             . ***
;
;--

        .MACRO  TUSEEK  REC,DR,?A

        .NLIST
        .LIST ME
        .LIST

A:      MOV     #TRBUF,R0       ;-->(POINT TO) XMIT BUFF
        MOVB    #RSCMND,@R0     ;FORM COMMAND MESSAGE PA
        MOVB    #RSMSIZ,1(R0)   ;THIS BIG
        MOVB    #RSSSEK,2(R0)   ;OP CODE IS SEEK
        MOV     REC,10.(R0)     ;TO THIS RECORD
        MOVB    DR,4.(R0)       ;AND WHICH DRIVE
        CLRB    3.(R0)          ;NO MODIFIER
        CLRB    5.(R0)          ;NO SWITCHES
        CLR     6.(R0)          ;NO SEQUENCE #
        CLR     8.(R0)          ;NO BYTE COUNT
        MOV     #RSMSIZ,R1      ;GET COUNT
        TST     (R1)+           ;PLUS FLAG + BCNT
                                ;FOR CHECKSUM CALC
        CALL    CHKSUM          ;R0-->TOP   R1=# OF BYTE
        MOV     R1,(R0)         ;INSERT INTO PACKET
                                ;SET UP EXPECTATIONS:
        MOV     #RSSNSZ,SNDCNT(R5)  ;HOW MANY TO SEND
        MOVB    #RSCMND,XSFLG(R5)   ;EXPECT END PACK
        MOV     #RSNDSZ,XSCNT(R5)   ;COUNT WITH THIS
        MOV     #1.,XSPKNM(R5)  ;EXPECT ONLY 1 PACKET

        CALL    RSVP            ;SEND
                                ;AND RETURN TO SCHEDULER
        BIT     #BIT3,@R5       ;RETRY (FLAG BYTE ERROR)
        BNE     A               ;YES
```

```
4599
4600                                                    .NLIST
4601                                                    .NLIST ME
4602                                                    .LIST
4603                                                    .ENDM
```

```
4606                    ;++
4607                    ;THE RETRY MACRO IMPLIMENTS THE COMPLETE PROTOCOL NECESSARY TO INITIATE
4608                    ;A RETRY (READ OPERATION) SEQUENCE.
4609                    ;
4610                    ;SETS UP THE EXPECTED PROTOCOL RESPONSES:  THE NUMBER OF PACKETS
4611                    ;(XSPKNM) AND THEIR FLAG BYTES AND COUNTS (XSFLG, XSCNT).  CALLS
4612                    ;'RSVP' TO SEND EACH PACKET, AND 'CHKSUM' TO CALC. THE PACKET
4613                    ;CHECKSUM.
4614                    ;
4615                    ; INPUTS  - DEVICE BLOCK @R5
4616                    ;           TRBUF - BUFFER ADDRESS
4617                    ;           UNITS TEST REGISTERS FROM SWAPIN
4618                    ;
4619                    ; OUTPUTS - SNDCNT(R5) = # OF BYTES TO SEND
4620                    ;           XSPKNM = # OF PACKETS EXPECTED
4621                    ;           XSFLG = FLAG BYTE OF 1ST PACKET
4622                    ;           XSCNT = BYTE COUNT OF 1ST PACKET
4623                    ;             . ***
4624                    ;             .   *  SUBSEQUENT XSFLGS
4625                    ;             .    >
4626                    ;             .   *  AND XSCNTS
4627                    ;             . ***
4628                    ;--
4629
4630                            .MACRO  TURTRY  REC,BCNT,DR,?A,?B,?C,?D,?E
4631
4632                            .NLIST
4633                            .LIST ME
4634                            .LIST
4635
4636
4637                    D:      MOV     #TRBUF,R0       ;FORM CMND PACK:
4638                            MOVB    #RSCMND,@R0     ;MESSAGE PACK TYPE
4639                            MOVB    #RSMSIZ,1(R0)   ;THIS BIG
4640                            MOVB    #RSSRD,2(R0)    ;OP CODE-READ
4641                            MOV     REC,10.(R0)     ;THIS RECORD
4642                            MOVB    DR,4.(R0)       ;THIS DRIVE
4643                            CLRB    3(R0)           ;PRESET NORM THRESHOLD
4644                            TSTB    @R5             ;REDUCED?
4645                            BPL     E               ;NO
4646                            INCB    3(R0)           ;YES-CHANGE THRESHOLD
4647                    E:      MOV     BCNT,8.(R0)     ;# BYTES DESIRED
4648                            MOVB    #020,5.(R0)     ;MAINTENANCE MODE
4649                            CLR     6.(R0)          ;NO SEQUENCE #
4650                            MOV     #RSMSIZ,R1      ;SIZE OF PACKET
4651                            TST     (R1)+           ;PLUS FLAG+COUNT INTO R1
4652                            MOV     #RSSNSZ,SNDCNT(R5)  ;SET UP SIZE TO SEND
4653
4654                            CALL    CHKSUM          ;FORM CHECKSUM R1=COUNT
4655                            MOV     R1,(R0)         ;INSERT IN PACKET
4656
4657                            MOV     BCNT,R1         ;SET EXPECTATIONS:
4658                                                    ;CALC # OF DATA PACKETS
4659                            MOV     #XSFLG,R3       ;OFFSET OF FLAG
4660                            ADD     R5,R3           ;ABS. ADDR. OF XSFLG
4661                            CLR     R2              ;PRESET
                        A:      INC     R2              ;# PACKETS EXPECTED
```

```
                                                        MOV     #RSDATA,(R3)+   ;LOAD XSFLG
                                                        MOV     #132.,(R3)+     ;AND EXPECT COUNT
                                                        SUB     #128.,R1        ;NEG RESULT LAST TIME
                                                        BLOS    C               ;LAST TIME!
                                                        BR      A               ;MORE TO DO
                                                C:      INC     R2              ;ADD ONE FOR END PACK
                                                        MOV     R2,XSPKNM(R5)   ;SAVE # PACKETS TO EXPEC
                                                        MOV     #RSEND,(R3)+    ;EXPECT AN END
                                                        MOV     #RSNDSZ,(R3)    ;THIS BIG-14. BYTES

                                                        CALL    RSVP            ;SEND
                                                                                ;AND RETURN TO SCHEDULER

                                        .NLIST
                                        .NLIST ME
                                        .LIST
                                        .ENDM
```

```
4681                    ;++
4682                    ;THE READ MACRO IMPLIMENTS THE COMPLETE PROTOCOL NECESSARY TO INITIATE
4683                    ;A READ SEQUENCE.
4684                    ;
4685                    ;SETS UP THE EXPECTED PROTOCOL RESPONSES:  THE NUMBER OF PACKETS
4686                    ;(XSPKNM) AND THEIR FLAG BYTES AND COUNTS (XSFLG, XSCNT).  CALLS
4687                    ;'RSVP' TO SEND EACH PACKET, AND 'CHKSUM' TO CALC. THE PACKET
4688                    ;CHECKSUM.
4689                    ;
4690                    ; INPUTS  - DEVICE BLOCK @R5
4691                    ;           TRBUF - BUFFER ADDRESS
4692                    ;           UNITS TEST REGISTERS FROM SWAPIN
4693                    ;
4694                    ; OUTPUTS - SNDCNT(R5) = # OF BYTES TO SEND
4695                    ;           XSPKNM = # OF PACKETS EXPECTED
4696                    ;           XSFLG = FLAG BYTE OF 1ST PACKET
4697                    ;           XSCNT = BYTE COUNT OF 1ST PACKET
4698                    ;             .***
4699                    ;             .   *  SUBSEQUENT XSFLGS
4700                    ;             .    >
4701                    ;             .   *  AND XSCNTS
4702                    ;             .***
4703                    ;--
4704
4705                            .MACRO  TUREAD  REC,BCNT,DR,VER,?A,?B,?C,?D,?E
4706
4707                            .NLIST
4708                            .LIST ME
4709                            .LIST
4710
4711
4712
4713                    E:      MOV     #TRBUF,R0       ;FORM CMND PACK:
4714                            MOVB    #RSCMND,@R0     ;MESSAGE PACK TYPE
4715                            MOVB    #RSMSIZ,1(R0)   ;THIS BIG
4716                            MOVB    #RSSRD,2(R0)    ;OP CODE IS READ
4717                            MOV     REC,10.(R0)     ;THIS RECORD
4718                            MOVB    DR,4.(R0)       ;THIS DRIVE
4719                            MOVB    VER,3.(R0)      ;VERIFY
4720                            MOV     BCNT,8.(R0)     ;TOTAL BYTES TO READ
4721                            MOVB    #020,5.(R0)     ;MAINTENANCE MODE
4722                            CLR     6.(R0)          ;NO SEQUENCE #
4723                            MOV     #RSMSIZ,R1      ;GET SIZE OF PACKET
4724                            TST     (R1)+           ;+2 FOR CHECKSUM
4725                            MOV     #RSSNSZ,SNDCNT(R5)  ;SIZE TO SEND
4726                            CALL    CHKSUM          ;FORM CHECKSUM R1=COUNT
4727                            MOV     R1,(R0)         ;INSERT CHECKSUM
4728
4729                            MOV     BCNT,R1         ;SET EXPECTATIONS:
4730                                                    ;CALC # OF DATA PACKETS
4731                            MOV     #XSFLG,R3       ;GET OFFSET
4732                            ADD     R5,R3           ;ABS. ADDR. OF XSFLG
4733                            CLR     R2              ;PRESET AS NONE
4734                    A:      INC     R2              ;# PACKETS EXPECTED
4735                            MOV     #RSDATA,(R3)+   ;LOAD XSFLG
4736                            MOV     #132.,(R3)+     ;AND EXPECTED COUNT
                                SUB     #128.,R1        ;NEG RESULT LAST TIME
```

```
                                                        BLOS    C               ;LAST TIME
                                                        BR      A               ;MORE TO DO
                                                C:      INC     R2              ;ADD ONE FOR END PACK
                                                        MOV     R2,XSPKNM(R5)   ;SAVE # PACKETS TO EXPEC
                                                        MOV     #RSEND,(R3)+    ;EXPECT AN END ALSO...
                                                        MOV     #RSNDSZ,(R3)    ;THIS BIG-14. BYTES
                                                        CALL    RSVP            ;SEND
                                                                                ;AND RETURN TO SCHEDULER
                                                D:      BIT     #BIT10!BIT3,@R5 ;RETRY?
                                                        BEQ     B               ;NO.
                                                        TURTRY  REC,BCNT,DR     ;YES
                                                        BR      D               ;ANOTHER RETRY?
                                                B:      NOP                     ;NO


                                .NLIST
                                .NLIST ME
                                .LIST
                                .ENDM
```

```
4758                    ;++
4759                    ;THE SELF TEST MACRO IMPLIMENTS THE COMPLETE PROTOCOL NECESSARY TO
4760                    ;INITIATE A 'DIAGNOSE' SEQUENCE.
4761                    ;
4762                    ;SETS UP THE EXPECTED PROTOCOL RESPONSES:  THE NUMBER OF PACKETS
4763                    ;(XSPKNM) AND THEIR FLAG BYTES AND COUNTS (XSFLG, XSCNT).  CALLS
4764                    ;'RSVP' TO SEND EACH PACKET, AND 'CHKSUM' TO CALC. THE PACKET
4765                    ;CHECKSUM.
4766                    ;
4767                    ; INPUTS  - DEVICE BLOCK @R5
4768                    ;           TRBUF - BUFFER ADDRESS
4769                    ;           UNITS REGISTERS TEST FROM SWAPIN
4770                    ;
4771                    ; OUTPUTS - SNDCNT(R5) = # OF BYTES TO SEND
4772                    ;           XSPKNM = # OF PACKETS EXPECTED
4773                    ;           XSFLG = FLAG BYTE OF 1ST PACKET
4774                    ;           XSCNT = BYTE COUNT OF 1ST PACKET
4775                    ;             . ***
4776                    ;             .   *  SUBSEQUENT XSFLGS
4777                    ;             .    >
4778                    ;             .   *  AND XSCNTS
4779                    ;             . ***
4780                    ;--
4781
4782                            .MACRO  TUSELF  ?A
4783
4784                            .NLIST
4785                            .LIST ME
4786                            .LIST
4787
4788    A:          MOV     #TRBUF,R0           ;FORM COMMAND PACKET
4789                MOVB    #RSCMND,@R0         ;COMMAND FLAG
4790                MOVB    #RSMSIZ,1(R0)       ;SIZE OF MESSAGE
4791                MOVB    #RSSSLF,2(R0)       ;SELF TEST OPERATION
4792                CLRB    3(R0)               ;NO MODIFIER.
4793                CLR     4(R0)               ;NO DRIVE OR SWITCHES
4794                CLR     6(R0)               ;NO SEQUENCE NUMBER
4795                CLR     8.(R0)              ;NO BYTES
4796                CLR     10.(R0)             ;NO RECORD #
4797                MOV     #RSMSIZ,R1          ;GET SIZE
4798                TST     (R1)+               ;+2 FOR CHECKSUM
4799                MOV     #RSSNSZ,SNDCNT(R5)  ;SIZE TO SEND
4800                CALL    CHKSUM              ;FORM CHECKSUM
4801                MOV     R1,(R0)             ;INSERT INTO PACKET
4802                MOV     #RSEND,XSFLG(R5)    ;EXPECT END,
4803                MOV     #RSNDSZ,XSCNT(R5)   ;THIS BIG
4804                MOV     #1,XSPKNM(R5)       ;AND 1 PACKET
4805                                            ;SEND
4806                CALL    RSVP                ;RETURN TO SCHEDULER
4807                BIT     #BIT3,@R5           ;RETRY?(BAD FLAG)
4808                BNE     A                   ;YES
4809
4810
4811                .NLIST
4812                .NLIST ME
4813                .LIST
                    .ENDM
```

```
4816                    ;++
4817                    ;THE TEST ID MACRO INTERFACES THE SUPERVISOR'S TEST DISPATCH TO THE
4818                    ;DIAGNOSTIC'S FORMAT BY IMPLEMENTING CALLS THAT:  1) INITIALIZE THE
4819                    ;PC OF THE TEST CODE (TSTPC(R5)).  2) ASSIGN THE 1ST DRIVES,  3) RUN
4820                    ;THE TEST,  4) SWITCH DRIVES AND REINITIALIZE, 5) RUN THE TEST AGAIN.
4821                    ;--
4822
4823                            .MACRO  TSTID   ADDRES,?A
4824
4825                            .NLIST
4826                            .LIST ME
4827                            .LIST
4828
4829                                            MOV     ADDRES,TSTTOP   ;SAVE ADDR OF TEST
4830                                            CALL    SETUP           ;INIT UNITS TSTPC
4831                                            CALL    SETDR           ;GET 1ST DRVS.
4832                                            CALL    RUN             ;DO TEST
4833                                            CALL    SWAPDR          ;GET NEXT DRVS.
4834                                            BCC     A               ;BR NO 2ND DRVS
4835                                            CALL    SETUP           ;REINIT UNITS TSTPC
4836                                            CALL    RUN             ;REPEAT TEST
4837                                    A:                              ;DONE
4838                            .NLIST
4839                            .NLIST ME
4840                            .LIST
4841                            .ENDM
                        ;-------------------------------------------------------------------------
```

```
                                        .SBTTL  GLOBAL SUBROUTINES SECTION

                                        ;++
                                        ; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES THAT ARE USED
                                        ; TO LINK THE DIAGNOSTIC TO THE SUPERVISOR (THROUGH THE TSTID MACRO).
                                        ;--

                                        ;++
                                        ; SWAPDR
                                        ;     SUBROUTINE TO DETERMINE IF TO TEST OTHER DRIVE (FOR ALL UNITS)
                                        ; INPUTS:       DR(R5) - DRIVE CONFIGURATION
                                        ;               BLKTBL - TOP OF DATA BLOCK ALLOCATION TABLE
                                        ;               LSTDEV - ADDR. OF LAST UNIT'S DATA BLOCK
                                        ;
                                        ; OUTPUTS:      DR(R5) UPDATED TO TEST SAME OR OTHER DRIVE
                                        ;               CARRY SET IF SECOND PASS NECESSARY
                                        ;--

005530  005002                          SWAPDR:: CLR    R2                      ;FOR # OF DRIVE 1'S.
005532  012737  003350  005630                 MOV     #BLKTBL,SWPTR           ;TABLE ADDR. OF 1ST UNIT
005540  017705  000064                  1$:     MOV     @SWPTR,R5               ;GET DATA BLOCK ADDR.
005544  032715  100000                          BIT     #BIT15,@R5              ;ABORTED?
005550  001013                                  BNE     3$                      ;YES
005552  032765  000001  000060                 BIT     #BIT0,DR(R5)            ;DID DR. 0?
005560  001007                                  BNE     3$                      ;NO, DID DR.1 1ST PASS
005562  032765  001000  000060                 BIT     #BIT9,DR(R5)            ;YES; 1 SELECTED?
005570  001403                                  BEQ     3$                      ;NO, ALL DONE
005572  105265  000060                          INCB    DR(R5)                  ;YES, SWAP
005576  005202                                  INC     R2                      ;ONE MORE TO TEST
005600  023727  005630  003366          3$:     CMP     SWPTR,#LSTDEV           ;LAST DEVICE?
005606  103004                                  BHIS    4$                      ;YES
005610  062737  000002  005630                 ADD     #2,SWPTR                ;NO-POINT NEXT
005616  000750                                  BR      1$                      ;DO

005620  005702                          4$:     TST     R2                      ;(CLEAR CARRY),MORE TO DO?
005622  001401                                  BEQ     5$                      ;NO
005624  000261                                  SEC                             ;YES
005626  000207                          5$:     RETURN                          ;RETURN

005630  000000                          SWPTR:  .WORD
```

```
                                        ;++
                                        ; SETDR - SUBROUTINE TO GET DRIVE FOR 1ST PASS FOR EACH TEST
                                        ;
                                        ; INPUTS:       DR(R5) - DRIVE CONFIGURATION
                                        ;               BLKTBL - TOP OF DATA BLOCK ALLOCATION TABLE
                                        ;               LSTDEV - ADDR. OF LAST UNIT'S DATA BLOCK
                                        ;
                                        ; OUTPUTS:      DR(R5) IS SET TO TEST DRIVE 0 OR DRIVE 1
                                        ;--

005632  012737  003350  005706  SETDR:: MOV     #BLKTBL,SETPTR  ;TABLE OF ADDR. 1ST UNIT
005640  017705  000042          1$:     MOV     @SETPTR,R5      ;GET DATA BLOCK ADDR.
005644  105065  000060                  CLRB    DR(R5)          ;PRESET AS DR0
005650  032765  000400  000060          BIT     #BIT8,DR(R5)    ;DO DR0?
005656  001002                          BNE     2$              ;YES
005660  105265  000060                  INCB    DR(R5)          ;NO-USE DRIVE 1
005664  023727  005706  003366  2$:     CMP     SETPTR,#LSTDEV  ;MORE UNITS
005672  103004                          BHIS    3$              ;NO-EXIT
005674  062737  000002  005706          ADD     #2,SETPTR       ;YES-GET TABLE ENTRY
005702  000756                          BR      1$              ;CONFIGURE THAT UNIT
005704  000207          3$:     RETURN
005706  000000  SETPTR: .WORD
```

```
4970
4971                                    ;++
4972                                    ; CLRALL - CLEARS INPUT BUFFER FOR RESPONSE FROM UNIT.
4973                                    ;
4974                                    ; INPUTS:       BLKTBL - TOP OF DATA BLOCK ALLOCATION TABLE
4975                                    ;               LSTDEV - ADDR. OF LAST UNIT'S DATA BLOCK
4976                                    ;
4977                                    ; OUTPUTS:      ALL UNITS BUFFERS CLEARED.
4978                                    ;
4979                                    ; CALLS:        CLRBUF
4980                                    ;--
4981  005710  012737  003350  006002    CLRALL:: MOV    #BLKTBL,CLRPTR  ;TOP OF TABLE OF ADDRESSES
4982  005716  017705  000060            1$:      MOV    @CLRPTR,R5      ;GET DATA BLOCK
4983  005722  004737  005750                     CALL   CLRBUF          ;CLEAR IT'S RECEIVE BUFFER
4984  005726  023727  006002  003366             CMP    CLRPTR,#LSTDEV  ;LAST DEV?
4985  005734  103004                             BHIS   2$              ;YES
4986  005736  062737  000002  006002             ADD    #2,CLRPTR       ;-->NEXT
4987  005744  000764                             BR     1$              ;CONTINUE
4988  005746  000207            2$:      RETURN
```

```
 4991
 4992
 4993
 4994
 4995
 4996
 4997
 4998
 4999  005750
   (1) 005750  010046
   (1)
   (1)
 5000  005752
   (1) 005752  010446
   (1)
   (1)
 5001  005754  016500  000102
 5002  005760  012704  001036
 5003  005764  005020
 5004  005766  162704  000002
 5005  005772  001374
 5006  005774
   (1) 005774  012604
   (1)
 5007  005776
   (1) 005776  012600
   (1)
 5008  006000  000207
 5009  006002  000000
```

```
                        ;++
                        ; CLRBUF - CLEARS 1 UNIT'S INPUT BUFFER.
                        ; INPUTS:  RCVBUF(R5) IS BUFFER START
                        ;          RCBFSZ - SIZE OF RECEIVE BUFFER IN BYTES
                        ;          RCBFSZ IS SIZE OF BUFFER
                        ; OUTPUTS:  CLEARED AREA.
                        ;--

CLRBUF:: PUSH   R0              ;SAVE R0
                                MOV     R0,-(SP)

         PUSH   R4              ;SAVE R4
                                MOV     R4,-(SP)

         MOV    RCVBUF(R5),R0   ;GET ADDRESS OF BUFFER
         MOV    #RCBFSZ,R4      ;SIZE IN BYTES
1$:      CLR    (R0)+           ;CLEAR IT
         SUB    #2,R4           ;2 BYTES LESS
         BNE    1$              ;MORE
         POP    R4              ;RESTORE
                                MOV     (SP)+,R4

         POP    R0              ;
                                MOV     (SP)+,R0

         RETURN                 ;EXIT
CLRPTR:  .WORD
```

```
5012
5013                                    ;++
5014                                    ; SETUP - CALLED WITHIN EACH TEST TO INSERT BEGINNING ADDRESS OF THE
5015                                    ;         TEST INTO ALL UNITS TEST PC'S.
5016                                    ; INPUTS:       TSTTOP LOADED WITH TEST ALGORITHMS STARTING ADDR.
5017                                    ;               BLKTBL - TOP OF DATA BLOCK ALLOCATION TABLE
5018                                    ;               LSTDEV - ADDR. OF LAST UNIT'S DATA BLOCK
5019                                    ;
5020                                    ; OUTPUTS:      TSTPC(R5) FOR ALL UNITS
5021                                    ;               DONE - CLEARED
5022                                    ;--
5023  006004  005037  003322
5024  006010  012737  003350  003324    SETUP:: CLR     DONE                ;NOT DONE YET
5025  006016  017705  175302                    MOV     #BLKTBL,IDPTR       ;TABLE TOP ADDR
5026  006022  013765  003326  000020    1$:     MOV     @IDPTR,R5           ;DEVICE'S DATA BLOCK
5027  006030  023727  003324  003366            MOV     TSTTOP,TSTPC(R5)    ;INSERT PC FOR TOP OF TEST
5028  006036  103004                            CMP     IDPTR,#LSTDEV       ;ALL UNITS SET?
5029  006040  062737  000002  003324            BHIS    2$                  ;YES
5030  006046  000763                            ADD     #2,IDPTR            ;NO,GET NEXT POINTER
5031  006050  000207                            BR      1$                  ;SET HIM UP
                                        2$:     RETURN                      ;DONE
```

```
                                        ;++
                                        ; RUN - IMPLEMENTS THE CALLS TO SEND PACKETS, RECEIVE PACKETS, THEN
                                        ;       CHECK ANSWERS DURING TEST RUN TIME.
                                        ; INPUTS:  DONE
                                        ; OUTPUTS: NONE
                                        ;--

006052  004737  006102          RUN::   CALL    NXTST           ;MAKE AND SEND NEXT PACK TO ALL
                                                                ;UNABORTED UNITS
006056  005737  003322                  TST     DONE            ;COMPLETE?
006062  001006                          BNE     2$              ;YES
006064  004737  007152                  CALL    GETANS          ;NO,GET ALL RESPONSES

006070                                  BREAK                   ;SUPERVISOR CHECK
  (3)   006070  104422                                                  TRAP    C$BRK

006072  004737  010576                  CALL    CHKANS          ;CHECK ALL RESPONSES
006076  000765                          BR      RUN             ;CONTINUE TILL DONE
006100  000207                  2$:     RETURN
```

```
                                        .SBTTL  NXTST / THE SCHEDULER

                                        ;++
                                        ; NXTST - DISPATCH EXECUTION USING EACH UN-ABORTED UNIT'S TEST PROGRAM
                                        ;         COUNTER, (TSTPC(R5)). (THE POINTER TO THE TEST CODE THAT COMPRISES
                                        ;         MAKING A PACKET AND SENDING IT. CHECKS FIRST FOR ANY UN-ABORTED UNIT
                                        ;         THAT IS RETRYING EITHER A DATA ERRCR OR A 'INDECIPHERABLE FLAG BYTE'
                                        ;         ERROR, IN ORDER TO SERVICE ONLY THAT UNIT THIS PASS.  INITS
                                        ;       NON-RETRYING UNITS IF NECESSARY. IF NO RETRIES,DISPATCH ALL
                                        ;         UNITS IN ROUND ROBIN FASHION.
                                        ;
                                        ; INPUTS:      (IMPLIED) DATA BLOCKS.
                                        ;              BLKTBL - TOP OF DATA BLOCK ALLOCATION TABLE
                                        ;              LSTDEV - ADDR. OF LAST UNIT'S DATA BLOCK
                                        ;
                                        ; OUTPUTS:     ERRSF IF ALL UNITS ARE ABORTED.(TO NOTIFY APT)
                                        ;              SYSTAT IS UPDATED
                                        ;--

006102  000240                          NXTST:: NOP
006104  012737  003350  003312                  MOV     #BLKTBL,DEVPTR  ;UNIT 0 TO START
006112  017705  175174                  1$:     MOV     @DEVPTR,R5      ;GET DATA BLOCK
006116  005715                          1$:     TST     @R5             ;ABORTED?
006120  100504                                  BMI     2$              ; YES... CHECK NEXT UNIT
006122  032715  000010                  3$:     BIT     #BIT3,@R5       ;NO-RETRY 'BAD FLAG'?
006126  001040                                  BNE     5$              ;YES...(SEND BREAK;THEN CMD PACK)
006130  032715  020000                          BIT     #BIT13,@R5      ;NO-RETRYING STILL (NO END PACK YET)?
006134  001426                                  BEQ     7$              ;NO...
006136  032715  000400                          BIT     #BIT8,@R5       ;RETRYING A WRITE?
006142  001453                                  BEQ     4$              ;NO...
006144                                          SWAPIN                  ;YES-GET DEVICE REGESTERS
(1)     006144  016500  000006                                          MOV     6.(R5),R0
(1)     006150  016501  000010                                          MOV     8.(R5),R1
(1)     006154  016502  000012                                          MOV     10.(R5),R2
(1)     006160  016503  000014                                          MOV     12.(R5),R3
(1)     006164  016504  000016                                          MOV     14.(R5),R4
(1)
006170  020265  000206                          CMP     R2,SAVCNT(R5)   ;CURRENT COUNT = SAVED COUNT? (WHERE WE STARTED)
006174  001036                                  BNE     4$              ;NO...(CONTINUE SENDING DATA PACKS)
006176  042737  000004  003306                  BIC     #BIT2,SYSTAT    ;YES-CLEAR RETRY FLAGS
006204  042715  020000                          BIC     #BIT13,@R5      ;
006210  000450                                  BR      2$              ;
006212  032715  002000                  7$:     BIT     #BIT10,@R5      ;CHECK NEXT UNIT.
006216  001445                                  BEQ     2$              ;NO-RETRY DATA ERROR?
006220  052737  000002  003306                  BIS     #BIT1,SYSTAT    ;NO...ON TO NEXT UNIT
006226  000424                                  BR      6$              ;SET RETRY STATUS TO 'DATA ERROR' TYPE
                                                                        ;YES...
006230                                  5$:     SWAPIN                  ;GET DEVICE REGISTERS
(1)     006230  016500  000006                                          MOV     6.(R5),R0
(1)     006234  016501  000010                                          MOV     8.(R5),R1
(1)     006240  016502  000012                                          MOV     10.(R5),R2
(1)     006244  016503  000014                                          MOV     12.(R5),R3
(1)     006250  016504  000016                                          MOV     14.(R5),R4
(1)
006254  010265  000206                          MOV     R2,SAVCNT(R5)   ;SAVE THE BYTE COUNT (FOR WRITE OPERATION)
                                                                        ;TO MARK HOW MANY DATA PACKS TO SEND
```

```
 5098  006260  004737  014010                    CALL    DOBRK           ;SEND INIT
 5099  006264  032715  100000                    BIT     #BIT15,@R5      ;ABORTED?
 5100  006270  001020                            BNE     2$              ;YES...
 5101  006272  052737  000004  003306    4$:     BIS     #BIT2,SYSTAT    ;NOT ABORTED-SET RETRY STATUS
 5102  006300                            6$:     SWAPIN                  ;GET DEVICE REGISTERS
   (1) 006300  016500  000006                                            MOV     6.(R5),R0
   (1) 006304  016501  000010                                            MOV     8.(R5),R1
   (1) 006310  016502  000012                                            MOV     10.(R5),R2
   (1) 006314  016503  000014                                            MOV     12.(R5),R3
   (1) 006320  016504  000016                                            MOV     14.(R5),R4
   (1)
 5103  006324  004775  000020                    JSR     PC,@TSTPC(R5)   ;DO TEST FOR
 5104  006330  000477                            BR      NXTRET          ;THIS UNIT ONLY-EXIT
 5105  006332  023727  003312  003366    2$:     CMP     DEVPTR,#LSTDEV  ;TRY NEXT UNIT?
 5106  006340  103004                            BHIS    NXTST2          ;NO
 5107  006342  062737  000002  003312            ADD     #2.,DEVPTR      ;YES,->NEXT
 5108  006350  000660                            BR      1$              ;GET BLOCK
 5109
 5110  006352  005037  006532            NXTST2: CLR     ABONM           ;HERE=NO RETRIES TO DO, NO UNIT ABORTED YET
 5111  006356  012737  003350  003312            MOV     #BLKTBL,DEVPTR  ;-->UNIT 0 STORAGE BLOCK
 5112  006364  017705  174722            PERDEV: MOV     @DEVPTR,R5      ;R5-->NEXT DEVICE STORAGE BLOCK
 5113
 5114  006370  005715                    3$:     TST     @R5             ;ABORTED?
 5115  006372  100426                            BMI     4$              ;YES
 5116  006374  032715  040000                    BIT     #BIT14,@R5      ;SEND BREAK?
 5117  006400  001407                            BEQ     6$              ;NO
 5118  006402  004737  014010                    CALL    DOBRK           ;YES
 5119  006406  032715  040000                    BIT     #BIT14,@R5      ;SUCCESSFUL INIT?
 5120  006412  001016                            BNE     4$              ;NO ON TO NEXT UNIT
 5121  006414  005715                            TST     @R5             ;ABORTED?
 5122  006416  100414                            BMI     4$              ;YES-ON TO NEXT UNIT
 5123  006420                            6$:     SWAPIN                  ;NO,GET DEVICE REGISTERS R0-R4 CONTAINING TEST P
   (1) 006420  016500  000006                                            MOV     6.(R5),R0
   (1) 006424  016501  000010                                            MOV     8.(R5),R1
   (1) 006430  016502  000012                                            MOV     10.(R5),R2
   (1) 006434  016503  000014                                            MOV     12.(R5),R3
   (1) 006440  016504  000016                                            MOV     14.(R5),R4
   (1)
 5124  006444  004775  000020                    JSR     PC,@TSTPC(R5)   ;INITIATE 1 PACKET TRANSMISSION AND RETURN
 5125  006450  005715                    4$:     TST     @R5             ;ABORTED?
 5126  006452  100002                            BPL     8$              ;NO-ON TO NEXT UNIT
 5127  006454  005237  006532                    INC     ABONM           ;YES...ONE MORE TALLIED
 5128  006460  023727  003312  003366    8$:     CMP     DEVPTR,#LSTDEV  ;ALL TU'S TRIED?
 5129  006466  103004                            BHIS    5$              ;YES
 5130  006470  062737  000002  003312            ADD     #2.,DEVPTR      ;NO THE ADDRESS+2=NEXT ADDRESS
 5131  006476  000752                            BR      PERDEV          ;DO NEXT UNIT
 5132  006500  022737  000010  006532    5$:     CMP     #8.,ABONM       ;ALL ABORTED?
 5133  006506  001010                            BNE     NXTRET          ;NO
 5134  006510                                    ERRSF   100.,NOMOR      ;YES!
   (4) 006510  104454                                                    TRAP    C$ERSF
   (5) 006512  000144                                                    .WORD   100
   (5) 006514  006534                                                    .WORD   NOMOR
   (5) 006516  000000                                                    .WORD   0
 5135  006520                            11$:    BREAK                   ;SUPERVISOR BREAK
   (3) 006520  104422                                                    TRAP    C$BRK
 5136  006522  005237  003340                    INC     ALLGON          ;SET DON'T-PRINT STATISTICS FLAG
```

```
 5137  006526                                    DOCLN                   ;EXIT
   (3) 006526  104444
 5138  006530  000207                            NXTRET: RETURN                                                  TRAP    CSDCLN
 5139
 5140  006532  000000                            ABONM:  .WORD                   ;THE NUMBER OF ABORTED UNITS
 5141  006534  046101  020114  047125            NOMOR:  .ASCIZ  /ALL UNITS ABORTED!/
 5142          006560                                    .EVEN
```

```
                        .SBTTL  RSVP / XOFF AND SEND A PACKET TO ALL DEVICES

                        ;++
                        ; RSVP - SAVES TEST CODE PROGRAM COUNTER IN TSTPC(R5) AND UNIT'S REGIS-
                        ;        TERS.  IF NOT IN TEST 8, POINTS TO "XOFF" THAT PRECEEDS PACKET IN
                        ;        XMIT BUFFER AND SENDS PACKET WITH XOFF.  RETURNS TO SCHEDULER (NXTST)
                        ;        SO THAT OTHER UNITS PACKETS MAY BE FORMED, TO GET ALL UNITS WORKING
                        ;        AT ONCE.  IF IN TEST 8 AND THE UNIT IS NOT MODIFIED, SKIP REST OF
                        ;        ROUTINE.  IF IN TEST 8 AND THE UNIT IS MODIFIED DO NOT SEND XOFF AND
                        ;        PROCEED NORMALLY.
                        ; INPUTS:  (SP) CONTAINS UNITS PC TO SAVE SINCE RSVP WAS CALLED.  THE
                        ;         NUMBER PACKETS EXPECTED (XSPKNM), AND THE EXPECTED FLAGS AND
                        ;         BYTE COUNTS OF EACH (XSFLG, XSCNT...) ARE LOADED BY TEST CODE
                        ;         (MACROS).
                        ;         SNDCNT - # BYTES TO SEND
                        ;         REC(R5) - RECORD #
                        ;         TRBUF - BUFFER ADDR.
                        ;         XSPKNM(R5) - # EXPECTED
                        ;         RCVBUF(R5)
                        ;
                        ; OUTPUTS:  CMDSNT - UPDATED WITH PACKET OP CODE
                        ;         BLKER  - RECORD NUMBER STATISTICS UPDATED IF NOT RETRYING
                        ;                  AND COMMAND PACKET SENT.
                        ;         SUCCS(R5) - PRESET CLEAR
                        ;         STATUS WORD @R5 - BIT9 - DATA CHECK ERROR - CLEARED
                        ;                           BIT5 - "VERIFY" OPERATION
                        ;                           BIT4 - 0 = DATA PACK 1 = CMND
                        ;                           BIT8 - RD/WR OPERATION
                        ;         XSPTR - POINTS TO EXPECTED FLAG
                        ;         UPPER BYTE OF XSPKNM IS REPLICATED.
                        ;         PACKET POINTER (PKPTR(R5)) POINTS TO TOP OF UNITS RECEIVE BUFFER
                        ;         AREA (RCVBUF(R5)) FOR CURRENT UNIT.
                        ;--

006560  000240         RSVP::  NOP
006562  012665  000020          MOV     (SP)+,TSTPC(R5) ;FINISH TEST
006566                          SWAPOW                  ;SAVE WHERE YOU WERE IN TEST BODY AND
                                                        ;SAVE TEST REGISTERS
006566  010065  000006          MOV     R0,6.(R5)
006572  010165  000010          MOV     R1,8.(R5)
006576  010265  000012          MOV     R2,10.(R5)
006602  010365  000014          MOV     R3,12.(R5)
006606  010465  000016          MOV     R4,14.(R5)

                                                        ;CORRECT FOR RETURN TO SCHEDULER
006612  022737  000002  003342          CMP     #2,TEST9        ;***** IS THIS TEST 9
006620  001007                  BNE     XFNSND          ;***** NO
006622  022765  000000  000210          CMP     #0,MRSP(R5)     ;***** IF SO, IS THIS UNIT MODIFIED
006630  001523                  BEQ     ENDRSP          ;***** YES
006632  012700  027722  NOXOFF: MOV     #TRBUF,R0       ;FOR NORMAL PACKET SEND
006636  000404                  BR      SND             ;SEND XOFF+PACKET
006640  012700  027721  XFNSND: MOV     #TRBUF-1,R0     ;POINT TO XOFF
006644  005265  000070          INC     SNDCNT(R5)      ;ONE MORE TO SEND, TOO.
006650  004737  007102  SND:    CALL    SNDBYT          ;SEND BYTE
006654  005715                  TST     @R5             ;R5--> TO STATUS BLK
006656  100510                  BMI     6$              ;ABORTED? YES...QUIT
```

```
5194  006660  005365  000070                  DEC     SNDCNT(R5)          ;NO, SEND MORE
5195  006664  001371                          BNE     SND                 ;IF MORE TO SEND
5196  006666  012700  027722                  MOV     #TRBUF,R0           ;-->BUFFER
5197  006672  016537  000064  003332          MOV     REC(R5),@LKER       ;PREPARE FOR RECEIVE
5198  006700  156565  000032  000033          BISB    XSPKNM(R5),XSPKNM+1(R5)  ;REPLICATE LO. BYTE TO HI FOR GTPAKS, C
5199  006706  005065  000076                  CLR     SUCCS(R5)           ;NO SUCCESS YET
5200  006712  042715  001000                  BIC     #BIT9,@R5           ;NO DATA CHK ERROR YET
5201  006716  016565  000102  000104          MOV     RCVBUF(R5),PKPTR(R5)  ;TOP OF RCV BUFFER GOES THE 1ST PACKET
5202  006724  012704  000034                  MOV     #XSFLG,R4           ;FORM
5203  006730  060504                          ADD     R5,R4               ;ADDRESS
5204  006732  010465  000106                  MOV     R4,XSPTR(R5)        ;OF 1ST XSFLG
5205
5206  006736  042715  000020                  BIC     #BIT4,@R5           ;PRESET AS DATA PAK
5207  006742  121027  000002                  CMPB    @R0,#RSCMND         ;WAS IT COMMAND PAK?
5208  006746  001054                          BNE     6$                  ;NO...
5209  006750  116065  000002  000100          MOVB    2(R0),CMDSNT(R5)    ;YES-SAVE COMMAND
5210  006756  052715  000020                  BIS     #BIT4,@R5           ;ITS CMND PAK
5211
5212  006762  032715  002000                  BIT     #BIT10,@R5          ;RETRYING?
5213  006766  001044                          BNE     6$                  ;YES-DON'T UPDATE ANY STATS OR CONDITION
5214  006770  126027  000002  000002          CMPB    2(R0),#RSSRD        ;NO,A READ?
5215  006776  001012                          BNE     4$                  ;NO
5216  007000  042715  000400                  BIC     #BIT8,@R5           ;(FOR HARD/SOFT LOGGING) RD/WR FLAG=0
5217  007004  004737  013640                  CALL    WHCHDR              ;GET DRIVE
5218  007010  103403                          BCS     8$                  ;
5219  007012  005265  000114                  INC     RDN0(R5)            ;DRIVE 0
5220  007016  000402                          BR      4$                  ;
5221  007020  005265  000116          8$:     INC     RDN1(R5)            ;DRIVE 1
5222
5223  007024  126027  000002  000003  4$:     CMPB    2(R0),#RSSWR        ;A WRITE?
5224  007032  001022                          BNE     6$                  ;NO
5225  007034  052715  000400                  BIS     #BIT8,@R5           ;YES, RD/WR FLAG=1
5226  007040  105760  000003                  TSTB    3(R0)               ;VERIFY TOO?
5227  007044  001403                          BEQ     21$                 ;NO
5228  007046  052715  000040                  BIS     #BIT5,@R5           ;YES-SET VERIFY FLAG
5229  007052  000402                          BR      22$
5230  007054  042715  000040          21$:    BIC     #BIT5,@R5           ;(NO)-RESET VERIFY FLAG
5231  007060  004737  013640          22$:    CALL    WHCHDR              ;GET DRIVE NO
5232  007064  103403                          BCS     5$                  ;CARRY=DR1
5233  007066  005265  000110                  INC     WRTN0(R5)           ;# BLKS WRITTEN DR0
5234  007072  000402                          BR      6$                  ;EXIT
5235
5236  007074  005265  000112          5$:     INC     WRTN1(R5)           ;# BLKS WRITTEN DRV1
5237  007100                          6$:
5238  007100  000207                  ENDRSP: RETURN                      ;RETURN
```

```
                        .SBTTL  SNDBYT / OUTPUT A BYTE TO UNIT

                        ;++
                        ; SNDBYT - TEST 'READY' ON INTERFACE.  IF 'READY', SEND BYTE AND EXIT.
                        ;          IF TIMED OUT, LOG ERROR.
                        ; INPUTS - R0 = POINTER TO BUFFER
                        ;        - IMPLIED UNIT DATA BLOCK
                        ;        - CSNRDY - TIMEOUT CONSTANT
                        ; OUTPUTS - R0 IS INCREMENTED.
                        ; ERROR - NOT-READY-TO-SEND TIME OUT
                        ;--

007102                  SNDBYT:: PUSH   R1              ;ENTER R0-->BYTE
007102 010146                    MOV    R1,-(SP)

007104 013701 003344    4$:     MOV     CSNRDY,R1       ;GET TIMEOUT CONSTANT FOR NOT READY ERROR
007110 105775 000026    1$:     TSTB    @XMSR(R5)       ;READY TO SEND?
007114 100412                   BMI     2$              ;YES
007116                          PUSH    R0              ;NO, SAVE R0
007116 010046                   MOV     R0,-(SP)

007120                          BREAK                   ;MONITOR BREAK
007120 104422                                                           TRAP    C$BRK
007122                          POP     R0              ;RESTORE
007122 012600                   MOV     (SP)+,R0

007124 005301                           DEC     R1              ;ABORTED?
007126 001370                           BNE     1$              ;NO
007130 012704 000056                    MOV     #TOSNDB,R4      ;YES,SET CODE FOR TIMEOUT ERROR
007134 004737 012634                    CALL    LOG             ;LOG IT
007140 000402                           BR      3$              ;QUIT
007142 112075 000030    2$:     MOVB    (R0)+,@XMDB(R5) ;SEND IT
007146                  3$:     POP     R1              ;RESTORE
007146 012601                   MOV     (SP)+,R1

007150 000207                           RETURN                  ;DONE
```

```
5271
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282  007152  000240
5283  007154  032737  000006  003306
5284  007162  001010
5285  007164  012737  177777  010342
5286  007172  004737  005710
5287  007176  004737  007430
5288  007202  000404
5289  007204  004737  005750
5290
5291  007210  004737  007220
5292  007214  000207
5293
5294  007216  000000

                        .SBTTL  GETANS / GETS RESPONSES ROUND ROBIN USING "XON"

                        ;++
                        ; GETANS - IF A UNIT IS RETRYING CLEAR HIS RECEIVE BUFFER (CLRBUF) AND GET
                        ;          HIS RESPONSE (GTPKS1), ELSE, CLEAR ALL BUFFERS (CLRALL) AND
                        ;          GET ALL RESPONSES (GTPKS8).
                        ; INPUTS:  SYSTAT - SYSTEM STATUS WORD.
                        ;
                        ; OUTPUTS: SERVST = -1 IF NO RETRIES.
                        ;--

                        GETANS:: NOP                    ;1 UNIT IF RETRY;  ELSE ALL
                                 BIT    #BIT1!BIT2,SYSTAT       ;RETRY?
                                 BNE    1$              ;YES
                                 MOV    #-1,SERVST      ;PRESET NO UNITS SERVICED
                                 CALL   CLRALL          ;CLEAR ALL INPUT BUFFERS
                                 CALL   GTPKS8          ;GET ALL REPLYS
                                 BR     2$              ;EXIT
                        1$:      CALL   CLRBUF          ;RETRY-CLEAR 1 UNIT ONLY
                                                        ;R5->UNIT BY NXTST
                                 CALL   GTPKS1          ;GET 1 REPLY
                        2$:      RETURN                 ;DONE

                        GETPTR: .WORD
```

```
                                        .SBTTL  GTPKS1 / GET RETRY RESPONSE-1 UNIT

                                        ;++
                                        ; GTPKS1 - SENDS 'XON' TO UNIT, GETS FLAG BYTE (IF ANY), CHECKS IF IT IS
                                        ;          WHAT WAS EXPECTED.  IF IT IS, USE EXPECTED BYTE COUNT(XSCNT).  IF
                                        ;          NOT, CHECK IF PREMATURE-END PACK OR (SINCE MAINTENANCE MODE)
                                        ;          IF IT'S A PREMATURE DATA PACK.  ADJUST COUNT, GET REST OF
                                        ;          PACKET, AND REPEAT ABOVE UNTIL NO MORE PACKETS.
                                        ; INPUTS:  (IMPLIED) UNITS DATA BLOCK
                                        ;          RSNDSZ - END PACKET SIZE
                                        ;
                                        ; OUTPUTS: SYSTAT UPPER BYTE = FLAG BYTE RECEIVED
                                        ;--

007220  000240                  GTPKS1:: NOP                     ;R5->THE UNIT
007222  012703  000034                  MOV     #XSFLG,R3       ;THE OFFSET VALUE OF FLAG
007226  060503                          ADD     R5,R3           ;FORM THE ABSOLUTE ADDRESS
007230  010301                          MOV     R3,R1           ;R3-->ADDR. OF EXPECTED FLAG
007232  062701  000002                  ADD     #2.,R1          ;R1-->ADDR. OF EXPECTED COUNT
007236  012700  007426                  MOV     #EXON,R0        ;R0=ADDRESS
007242  004737  007102                  CALL    SNDBYT          ;XON THE DEVICE
                                                                ;*** TIME CRITICAL
007246  016500  000102                  MOV     RCVBUF(R5),R0   ;***--> TO THE BUFFER
007252  116502  000033                  MOVB    XSPKNM+1(R5),R2 ;***GET THE # OF PACKETS TO RECEIVE
007256  032702  177400                  BIT     #177400,R2      ;***SIGN UN-EXTEND
007262  011137  003316          1$:     MOV     @R1,RCBCNT      ;***HOW MANY BYTES IT SHOULD BE
007266  011337  003314                  MOV     @R3,RCFLG       ;***WHAT THE FIRST BYTE SHOULD BE
007272  004737  010346                  CALL    GTBYTE          ;***GET THE ALL IMPORTANT FLAG
007276  032715  100000                  BIT     #BIT15,@R5      ;TIMEOUT?
007302  001050                          BNE     4$              ;YES
007304  005300                          DEC     R0              ;-> BYTE RECIEVED
007306  111037  003307                  MOVB    @R0,SYSTAT+1    ;SAVE IT AS FLAG BYTE
007312  121037  003314                  CMPB    @R0,RCFLG       ;1ST BYTE WHAT WAS EXPECTED?
007316  001420                          BEQ     2$              ;YES
007320  121027  000002                  CMPB    @R0,#RSEND      ;NO, WAS IT END PAK?
007324  001006                          BNE     14$             ;NO
007326  012737  000016  003316          MOV     #RSNDSZ,RCBCNT  ;YES, USE END SIZE FOR COUNT
007334  012702  000001                  MOV     #1,R2           ;AND ASSUME IT'S LAST PACKET!
007340  000407                          BR      2$              ;CONTINUE RECEIVE
007342  121027  000001          14$:    CMPB    @R0,#RSDATA     ;WAS IT DATA?
007346  001026                          BNE     4$              ;NO,CHKANS MAY FIND INIT...
007350  012737  000204  003316          MOV     #RSDASZ,RCBCNT  ;YES, SET FOR DATA PAK SIZE
007356  005202                          INC     R2              ;ONE MORE PACK THAN EXPECTED (END PACK)

007360  005200                  2$:     INC     R0              ;RESTORE TO -> NEXT BYTE
007362  005337  003316          5$:     DEC     RCBCNT          ;THAT'S ONE LESS BYTE TO GO
007366  001411                          BEQ     3$              ;DONE
007370  004737  010346                  CALL    GTBYTE          ;GET REST OF PACKET
007374  005765  000074                  TST     DLV(R5)         ;ERROR
007400  001011                          BNE     4$              ;YES-ALL OVER
007402  032715  100000                  BIT     #BIT15,@R5      ;OR IF ABORTED
007406  001006                          BNE     4$              ;THEN QUIT
007410  000764                          BR      5$              ;CONTINUE RECEIVE

007412  005302                  3$:     DEC     R2              ;ONE LESS PACKET TO GO
007414  001403                          BEQ     4$              ;MORE PACKETS IN TRANSACTION?
```

```
5353
5354  007416  022121                          CMP     (R1)+,(R1)+     ;YES
5355  007420  022323                          CMP     (R3)+,(R3)+     ;POINT TO NEW EXPECTED COUNT
5356  007422  000717                          BR      1$              ;AND FLAG,
5357  007424  000207                  4$:     RETURN                  ;AND RECEIVE,
5358                                                                  ;RETURN
5359  007426      020                 EXON:   .BYTE   RSXON
5360  007427      023                 EXOFF:  .BYTE   RSXOFF
```

```
                        .SBTTL  GTPKS8 / GET RESPONSES (NO RETRIES)

                        ;++
                        ; GTPKS8 - IF IN TEST 9 AND THE UNIT IS NOT MODIFIED, SKIP THE REST
                        ;          OF THE ROUTINE.  OTHERWISE;
                        ;          SET ALL ABORTED UNITS SERVICED (SERVST: BIT POSITION).  UNTIL
                        ;          ALL UNITS SERVICED (SERVST=0), IF NO MORE PACKETS, SET UNIT
                        ;          SERVICED, ELSE, GET A FLAG BYTE FROM UNIT, DECREMENTING THE
                        ;          NUMBER OF PACKETS LEFT.  CHECK TO SEE IF EXPECTED FLAG,
                        ;          ADJUST COUNT IF NOT, GET REST OF PACKET.  IF WAS DATA PACK,
                        ;          AND NOT IN TEST 9, SEND "XOFF" TO ENHANCE THROUGHPUT AND GO ON
                        ;          TO NEXT UNIT (IF ANY).  IF IN TEST 9, DO NOT SEND "XOFFF".
                        ; INPUTS:  (IMPLIED)UNITS DATA BLOCK POINTED TO BY R5.  NONE PASSED.
                        ;          RSNDSZ - END PACK SIZE
                        ;          RSDNSZ - DATA + END SIZE
                        ;
                        ; OUTPUTS: SYSTAT - UPPER BYTE=1ST BYTE RECEIVED, CURRENT UNIT
                        ;--
007430  000240          GTPKS8:: NOP
007432  022737  000002  003342          CMP     #2,TEST9        ;GET ALL UNITS RESPONSES XOFF IF DATA PAK (THROU
007440  001006                          BNE     1$              ;***** IS THIS TEST 9
007442  022765  000000  000210          CMP     #0,MRSP(R5)     ;***** NO
007450  001002                          BNE     1$              ;***** IF SO, IS THIS UNIT MODIFIED
007452  000137  010146                  JMP     ENDGP8          ;***** YES, CONTINUE NORMALLY
007456  012737  003350  010344  1$:     MOV     #BLKTBL,GTPTR   ;***** ELSE, SKIP ROUTINE
007464  017705  000654          GTAGIN: MOV     @GTPTR,R5       ;->1ST
007470  032715  100000                  BIT     #BIT15,@R5      ;GET DATA BLOCK
007474  001404                          BEQ     2$              ;ABORTED?
007476  004737  010256                  CALL    SETSRV          ;NO
007502  000137  010104                  JMP     GTDOWN          ;YES-SET' SERVICED' AND
007506  105765  000033          2$:     TSTB    XSPKNM+1(R5)    ;ON TO NEXT UNIT
007512  001004                          BNE     3$              ;NO, ANY PACKETS LEFT?
007514  004737  010256                  CALL    SETSRV          ;YES
007520  000137  010104                  JMP     GTDOWN          ;NO-HE'S DONE
007524  105365  000033          3$:     DECB    XSPKNM+1(R5)    ;SO ON TO NEXT UNIT
007530  017537  000106  003314          MOV     @XSPTR(R5),RCFLG  ;NOW ITS ONE LESS PACKET
007536  062765  000002  000106          ADD     #2,XSPTR(R5)    ;GET EXPECTED FLAG
007544  017537  000106  003316          MOV     @XSPTR(R5),RCBCNT  ;--> COUNT
007552  022737  000002  003342          CMP     #2,TEST9        ;AND EXPECTED COUNT
007560  001404                          BEQ     1$              ;***** IF TEST 9
007562  012700  007426                  MOV     #EXON,R0        ;***** DO NOT SEND XON
                                                                ;-> XON
007566  004737  007102                  CALL    SNDBYT          ;***TIME CRITICAL
007572  016500  000104          1$:     MOV     PKPTR(R5),R0    ;***SEND IT
007576  004737  010346                  CALL    GTBYTE          ;***->WHERE 1ST BYTE GOES
007602  032715  100000                  BIT     #BIT15,@R5      ;***GET IT
007606  001404                          BEQ     4$              ;ABORTED?
007610  105065  000033                  CLRB    XSPKNM+1(R5)    ;NO-CONTINUE
007614  000137  010104                  JMP     GTDOWN          ;YES-NO MORE PACKETS EXPECTED
007620  005300                  4$:     DEC     R0              ;ON TO NEXT
007622  111037  003307                  MOVB    @R0,SYSTAT+1    ;-->BYTE JUST RECEIVED
007626  121037  003314                  CMPB    @R0,RCFLG       ;SAVE IT
007632  001436                          BEQ     GTOK            ;IS IT WHAT EXPECTED?
007634  105065  000033          UNXPCT: CLRB    XSPKNM+1(R5)    ;YES
007640  121027  000002                  CMPB    @R0,#RSEND      ;NO, MUST BE LAST REPLY
                                                                ;MAYBE AN END PAK?
```

```
5419  007644  001004                        BNE     4$                      ;NO
5420  007646  012737  000016  003316        MOV     #RSNDSZ,RCBCNT          ;YES, USE PROPER COUNT
5421  007654  000406                        BR      GTUM                    ;AND GET IT
5422  007656  121027  000001          4$:   CMPB    @R0,#RSDATA             ;IS IT DATA?
5423  007662  001110                        BNE     GTDOWN                  ;NO, ALL OVER, CHKANS WILL INIT UNIT
5424  007664  012737  000222  003316        MOV     #RSDNSZ,RCBCNT          ;YES, USE COUNT OF DATA + END PAK SURE TO FOLLOW
5425  007672  005200                  GTUM: INC     R0                      ;WHERE TO STUFF THE REST
5426  007674  005337  003316          5$:   DEC     RCBCNT                  ;ONE DOWN
5427  007700  001501                        BEQ     GTDOWN                  ;NONE TO GO
5428  007702  004737  010346                CALL    GTBYTE                  ;MORE TO GO
5429  007706  032715  100000                BIT     #BIT15,@R5              ;TIMEOUT?
5430  007712  001074                        BNE     GTDOWN                  ;YES
5431  007714  005765  000074                TST     DLV(R5)                 ;BUT DLV ERROR?
5432  007720  001765                        BEQ     5$                      ;NO
5433  007722  105065  000033                CLRB    XSPKNM+1(R5)            ;YES-LAST TIME
5434  007726  000466                        BR      GTDOWN                  ;ON TO NEXT
5435
5436  007730  005200                  GTOK: INC     R0                      ;NEXT PLACE IN BUFFER
5437
5438                                  ;*****************************************************************************
5439  007732  022737  000002  003342  1$:   CMP     #2,TEST9                ;*** REV.- IF,   NOT TEST 9
5440  007740  001022                        BNE     7$                      ;*** REV.- THEN, NO MRSP HANDSHAKING REQUIRED
5441  007742                                PUSH    R0                      ;*** REV.- ELSE, TEST MRSP HANDSHAKE.
  (1) 007742  010046                                                      MOV     R0,-(SP)
  (1)
  (1)
5442  007744  012737  000002  010254        MOV     #2,MRSDLY               ;*** REV.- DELAY FOR WAIT LOOP
5443                                                                      ;*** REV.- THIS IS THE BEGINNING DELAY LOOP
5444  007752  005000                  2$:   CLR     R0                      ;*** REV.-
5445  007754  005300                  3$:   DEC     R0                      ;*** REV.-
5446  007756  001376                        BNE     3$                      ;*** REV.-
5447  007760  005337  010254                DEC     MRSDLY                  ;*** REV.-
5448  007764  001372                        BNE     2$                      ;*** REV.- THIS IS THE END OF DELAY LOOP
5449
5450  007766  105775  000022                TSTB    @RCSR(R5)               ;*** REV.- IF,   DONE SET,
5451  007772  001066                        BNE     ERRMOD                  ;*** REV.- THEN, IT'S AN ERROR BECAUSE
5452                                                                      ;*** REV.-       THERE WAS NO MRSP HANDSHAKE.
5453  007774  012700  010252                MOV     #MODRSP,R0              ;*** REV.- ELSE, SEEMS TO BE OK, LETS
5454  010000  004737  007102                CALL    SNDBYT                  ;*** REV.-       SEND A 'CONTINUE' AND
5455  010004                                POP     R0                      ;*** REV.-       SEE IF HANDSHAKE WORKS.
  (1) 010004  012600                                                      MOV     (SP)+,R0
  (1)
5456                                  ;*****************************************************************************
5457  010006  005337  003316          7$:   DEC     RCBCNT                  ;MORE BYTES?
5458  010012  001413                        BEQ     4$                      ;NO-ALL DONE
5459  010014  004737  010346                CALL    GTBYTE                  ;YES-GET IT
5460  010020  032715  100000                BIT     #BIT15,@R5              ;TIMEOUT?
5461  010024  001027                        BNE     GTDOWN                  ;YES
5462  010026  005765  000074                TST     DLV(R5)                 ;ERROR?
5463  010032  001737                        BEQ     1$                      ;NO
5464  010034  105065  000033                CLRB    XSPKNM+1(R5)            ;LAST TIME
5465  010040  000421                        BR      GTDOWN                  ;EXIT
5466  010042  122775  000001  000104  4$:   CMPB    #RSDATA,@PKPTR(R5)  ;WAS DATA?
5467  010050  001015                        BNE     GTDOWN                  ;NO, ALL DONE
5468  010052  010065  000104                MOV     R0,PKPTR(R5)            ;START OF NEXT PACK NEXT TIME
5469                                  ;*****************************************************************************
```

```
5470  010056  022737  000002  003342          CMP     #2,TEST9        ;*** REV.- IF,   TEST 9
5471  010064  001003                          BNE     20$             ;*** REV.- ELSE,
5472  010066  005765  000210                  TST     MRSP(R5)        ;*** REV.- ANDIF, MRSP
5473  010072  001004                          BNE     GTDOWN          ;*** REV.- THEN, NO HANDSHAKE
5474                                  ;*************************************************************************
5475  010074  012700  007427          20$:    MOV     #EXOFF,R0       ;XOFF AND SEND TO
5476  010100  004737  007102                  CALL    SNDBYT          ;ENHANCE THRCUGHPUT
5477  010104  062765  000002  000106  GTDOWN: ADD     #2.,XSPTR(R5)   ;NEXT XSFLG FOR NEXT TRY
5478  010112  023727  010344  003366          CMP     GTPTR,#LSTDEV   ;DONE ONE CYCLE ALL UNITS?
5479  010120  103005                          BHIS    1$              ;YES
5480  010122  062737  000002  010344          ADD     #2,GTPTR        ;NEXT UNIT
5481  010130  000137  007464                  JMP     GTAGIN          ;CONTINUE RECEIVE
5482  010134  105737  010342          1$:     TSTB    SERVST          ;DONE SERVICING ALL PAKS
5483                                                                  ;FROM ALL UNITS?
5484  010140  001402                          BEQ     ENDGP8          ;YES
5485  010142  000137  007430                  JMP     GTPKS8          ;NO, KEEP TRYING
5486  010146  000207                  ENDGP8: RETURN                  ;RETURN
5487
5488  010150  000240                  ERRMOD: NOP                     ;*** REV.- MRSP ERROR
5489  010152                                  PRINTF  #MESMRS,UNITNO
  (8) 010152  013746  027366                                                    MOV     UNITNO,-
  (7) 010156  012746  010200                                                    MOV     #MESMRS,
  (6) 010162  012746  000002                                                    MOV     #2,-(SP)
  (3) 010166  010600                                                            MOV     SP,R0
  (4) 010170  104417                                                            TRAP    C$PNTF
  (4) 010172  062706  000006                                                    ADD     #6,SP
5490  010176  000207                          RETURN
5491
5492  010200  047045  051445  022471  MESMRS: .ASCIZ  !%N%S9%S2%01%S9%S9%AERROR IN MRSP PROTOCOL!
5493                                          .EVEN
5494  010252      020                 MODRSP: .BYTE   RSCONT
5495          010254                          .EVEN
5496  010254  000000                  MRSDLY: .WORD
```

```
                        .SBTTL  SETSRV / SET UNIT SERVICED

                        ;++
                        ; SETSRV - RESET THE BIT IN 'SERVST' CORRESPONDING TO THE UNIT NUMBER.
                        ; INPUTS - SERVST - 'SERVICED' WORD
                        ;        - @R5 = UNIT # (BITS 0, 1, 2)
                        ; OUTPUTS - SERVST MODIFIED
                        ;--

010256                  SETSRV: PUSH    R5              ;SET UNIT SERVICED
(1) 010256  010546              MOV     R5,-(SP)
(1)
(1)
010260                          PUSH    R0
(1) 010260  010046              MOV     R0,-(SP)
(1)
(1)
010262  011505          1$:     MOV     @R5,R5          ;GET STAT WD
010264  042705  177770          BIC     #177770,R5      ;MASK UNIT #
010270  012700  010322          MOV     #SRVTBL,R0      ;->TOP OF BIT TABLE
010274  005705                  TST     R5              ;RIGHT ONE?
010276  001404                  BEQ     2$              ;YES
010300  062700  000002          ADD     #2,R0           ;NO, ->NEXT
010304  005305                  DEC     R5              ;1 LESS
010306  000772                  BR      1$              ;CONTINUE
010310  041037  010342  2$:     BIC     @R0,SERVST      ;MOW IT DOWN
010314                          POP     R0
(1) 010314  012600              MOV     (SP)+,R0
(1)
010316                          POP     R5
(1) 010316  012605              MOV     (SP)+,R5
(1)
010320  000207                  RETURN                  ;RETURN

010322  000001          SRVTBL: .WORD   BIT0            ;BIT POSITION LOOKUP TABLE
010324  000002                  .WORD   BIT1
010326  000004                  .WORD   BIT2
010330  000010                  .WORD   BIT3
010332  000020                  .WORD   BIT4
010334  000040                  .WORD   BIT5
010336  000100                  .WORD   BIT6
010340  000200                  .WORD   BIT7

010342  000000          SERVST: .WORD
010344  000000          GTPTR:  .WORD
```

```
                        .SBTTL  GTBYTE / GET A BYTE FROM UNIT

                        ;++
                        ; GTBYTE - TEST INTERFACE FOR 'READY-TO-RECEIVE' AND INPUT A BYTE, IF
                        ;          SO.  IF NOT, THE FOLLOWING OCCURS:  SEND 'XOFF' TO UNIT IN
                        ;          PREPARATION FOR ^C CHECK ('BREAK' TO SUPERVISOR).  WAIT
                        ;          TO SEE IF A CHARACTER SLOPS OVER DUE TO UART LATENCY. IF
                        ;          ONE DOES THEN MIGHT AS WELL GET IT AND SEND 'XON' TO GET
                        ;          THE REST OF THE MESSAGE, OTHERWISE, 'BREAK'.  THEN SEND
                        ;          'XON', AND TEST FOR LONG TIMEOUT (A 30 SECOND REWIND).  IF SO,
                        ;          LOG ERROR, OTHERWISE REPEAT THE ABOVE UNTIL READY OR TIME OUT.
                        ;          REMEMBER TO PRESERVE R0 SINCE THE 'BREAK' TRAP CLOBBERS IT.
                        ;
                        ; INPUTS  - R0 POINTS TO INPUT BUFFER
                        ;         - IMPLIED UNITS DATA BLOCK
                        ;         - CSRCVB TIME OUT MULTIPLIER
                        ;
                        ; OUTPUTS - R0 IS INCREMENTED
                        ;         - DLV (R5) NON-ZERO ON INTERFACE ERROR.
                        ;
                        ; ERROR   - TIME OUT ON RECEIVE
                        ;--
010346  005037  010572          GTBYTE:: CLR     GBTMP           ;TIMEOUT REGISTER
010352  013704  003346                   MOV     CSRCVB,R4       ;TIMEOUT ERROR CONSTANT (MULTIPLIER)
010356  105775  000022          1$:      TSTB    @RCSR(R5)       ;READY?
010362  100013                           BPL     3$              ;NO
010364  017565  000024  000074           MOV     @RCDB(R5),DLV(R5)  ;GET ERROR + BYTE
010372  116520  000074                   MOVB    DLV(R5),(R0)+   ;COPY BYTE TO BUFFER
010376  005765  000074                   TST     DLV(R5)         ;ERROR?
010402  100472                           BMI     4$              ;YES-EXIT
010404  005065  000074                   CLR     DLV(R5)         ;NO-RESET
010410  000467                           BR      4$              ;AND EXIT
010412  005337  010572          3$:      DEC     GBTMP           ;DEC T.O. CONSTANT
010416  001357                           BNE     1$              ;STILL VALID

                        ;CODE TO SEE ^C DURING LONG SEEK OR REWIND
010420  010037  010574                   MOV     R0,GBTMP2       ;HERE GBTMP=0
010424  012700  007427                   MOV     #EXOFF,R0       ;R0 MUST BE PRESERVED!
010430  004737  007102                   CALL    SNDBYT          ;QUIET THE DEVICE
010434  105775  000022          6$:      TSTB    @RCSR(R5)       ;BY SENDING XOFF
010440  100415                           BMI     5$              ;CHARACTER SLOP OVER?
010442  005337  010572                   DEC     GBTMP           ;YES
010446  105737  010572                   TSTB    GBTMP           ;NO-WAIT A WHILE
010452  001370                           BNE     6$              ;DONE WAITING?
010454                                   BREAK                   ;NO
010454  104422                                                   ;YES-NO SLOP OVER
                                                                                        TRAP    CSBRK
010456  012700  007426                   MOV     #EXON,R0        ;START DEVICE TALKING
010462  004737  007102                   CALL    SNDBYT          ;AGAIN
010466  013700  010574                   MOV     GBTMP2,R0       ;RESTORE R0
010472  000426                           BR      7$              ;END KLUGE
010474  013700  010574          5$:      MOV     GBTMP2,R0       ;RESTORE R0
010500  017565  000024  000074           MOV     @RCDB(R5),DLV(R5)  ;GET ERROR + BYTE
010506  116520  000074                   MOVB    DLV(R5),(R0)+   ;COPY BYTE TO BUFFER
010512  005765  000074                   TST     DLV(R5)         ;ERROR?
```

```
5591  010516  100403                    BMI     17$             ;YES-EXIT
5592  010520  005065  000074            CLR     DLV(R5)         ;NO-CLEAR
5593  010524  000400                    BR      17$             ;EXIT
5594  010526  010037  010574    17$:    MOV     R0,GBTMP2       ;AGAIN SAVE R0
5595  010532  012700  007426            MOV     #EXON,R0        ;RESTORE TO TALKING STATE
5596  010536  004737  007102            CALL    SNDBYT          ;BY SENDING 'XON'
5597  010542  013?00  010574            MOV     GBTMP2,R0       ;RESTORE R0
5598  010546  000410                    BR      4$              ;DONE
5599  010550  005037  010572    7$:     CLR     GBTMP
5600  010554  005304                    DEC     R4              ;TIMEOUT?
5601  010556  001277                    BNE     1$              ;NO
5602  010560  012704  000050            MOV     #TORCVB,R4      ;YES
5603  010564  004737  012634            CALL    LOG             ;LOG ERROR.
5604  010570  000207            4$:     RETURN                  ;RETURN
5605  010572  000000    GBTMP:  .WORD   0
5606  010574  000000    GBTMP2: .WORD   0
```

```
                        .SBTTL  CHKANS / CHECK DEVICE(S) RESPONSE

                        ;++
                        ; CHKANS - AS IN "GETANS", IF RETRYING DO ONLY 1 UNIT ELSE DO ALL NON-
                        ;          ABORTED UNITS.  NOTE, IF IN TEST 9 AND THE UNIT IS NOT
                        ;          MODIFIED DO NOT CHECK UNIT.
                        ; INPUTS:  IMPLIED SYSTAT BIT1 (RETRYING)
                        ;          BLKTBL - TOP OF DATA BLOCK ALLOCATION TABLE
                        ;          LSTDEV - ADDR. OF LAST UNIT'S DATA BLOCK
                        ;
                        ; OUTPUTS: NONE PASSED.
                        ;--

010576  000240          CHKANS:: NOP                    ;IF RETRY THEN CHECK ONE
                                                        ;ELSE CHECK ALL
010600  032737  000006  003306          BIT     #BIT1!BIT2,SYSTAT       ;RETRYING?
010606  001403          BEQ     CHK8                    ;NO DO NORMAL
010610  004737  010706  CALL    CHKPKS                  ;YES DO SINGLE UNIT
                                                        ;R5 -> UNIT
010614  000432          BR      CHKANR                  ;ALL DONE

010616  012737  003350  010704  CHK8:   MOV     #BLKTBL,CHKPTR  ;YOU KNOW ... TOP OF TABLE
010624  017705  000054  2$:     MOV     @CHKPTR,R5      ;GET UNIT'S BLOCK ADDRESS
010630  032715  100000          BIT     #BIT15,@R5      ;ABORTED?
010634  001013          BNE     3$                      ;YES
010636  022737  000002  003342  CMP     #2,TEST9        ;***** IS THIS TEST 9
010644  001004          BNE     1$                      ;***** NO-CONTINUE NORMALLY
010646  022765  000000  000210  CMP     #0,MRSP(R5)     ;***** IF SO, IS THIS UNIT MODIFIED
010654  001402          BEQ     3$                      ;***** NO SKIP NEXT INSTR
010656  004737  010706  1$:     CALL    CHKPKS          ;NO, DO THIS GUY
010662  023727  010704  003366  3$:     CMP     CHKPTR,#LSTDEV  ;ALL DONE?
010670  103004          BHIS    CHKANR                  ;YES
010672  062737  000002  010704          ADD     #2,CHKPTR       ;NO,-->NEXT DEVICE
010700  000751          BR      2$                      ;DO DA

010702  000207          CHKANR: RETURN

010704  000000          CHKPTR: .WORD
```

```
                        .SBTTL  CHKPKS / DECIPHERS RESPONSE OF UNIT POINTED TO BY R5 /

                        ;++
                        ; CHKPKS - FOR UNIT R5 AND FOR ALL PACKETS, CHECK TO SEE IF PACKET IS DATA OR
                        ;          END PACK, CHECK CHECKSUMS, COMPARE DATA IF DATA PACK, CHECK
                        ;          SUCCESS CODE IF END.. IF UNKNOWN PACKET TYPE, CHECK FOR INTERFACE
                        ;          ERROR.  IF "CONTINUE" FALL THROUGH.  IF "INIT" SET "SEND
                        ;          BREAK" FLAG.  CALL "LOG" WITH R4=ERROR NUMBER IF ERROR.
                        ;          THIS ROUTINE IS ALSO USED TO DETERMINE THE PROTOCOL OF A UNIT.  IN
                        ;          THE FIRST PART OF TEST 9 A GET CHARACTERISTICS COMMAND PACKET WAS
                        ;          SENT TO THE TU58.  IF THE RESPONSE WAS A DATA PACKET, WHICH IS
                        ;          EXPECTED, THEN THE UNIT IS NOT MODIFIED, AND THE MRSP FLAG IS
                        ;          CLEARED.  IF THE RESPONSE IS AN END PACKET, WHICH WOULD BE
                        ;          HANDLED BY THIS ROUTINE AS AN UNKNOWN, THEN THE UNIT IS MODIFIED,
                        ;          AND THE MRSP FLAG IS SET.
                        ; INPUTS:  (IMPLIED) UNITS DATA BLOCK
                        ;
                        ; OUTPUTS: ERRORS - DLV ERROR
                        ;                 - UNKNOWN FLAG BYTE ERROR
                        ;                 - CHECKSUM ERROR
                        ;                 - DATA COMPARE ERROR
                        ;         R4 = ERROR NUMBER
                        ;         SYSTAT UPPER BYTE = 1ST BYTE OF RESPONSE
                        ;--
                        ;
 5674  010706  000240   CHKPKS:: NOP                     ;CHECK WHAT WAS RECIEVED
 5675  010710  042715   000010         BIC     #BIT3,@R5       ;CLEAR 'BAD FLAG' RETRY BIT
 5676  010714  016500   000102         MOV     RCVBUF(R5),R0   ;GET BUFFER ADDR.
 5677  010720  116502   000032         MOVB    XSPKNM(R5),R2   ;AND # OF PACKETS EXPECTED
 5678  010724  012703   000034         MOV     #XSFLG,R3       ;THE OFFSET VALUE
 5679  010730  060503                  ADD     R5,R3           ;R3-->THIS UNIT XSFLG AGAIN
 5680  010732  010301                  MOV     R3,R1           ;COPY TO R1
 5681  010734  062701   000002         ADD     #2,R1           ;R1-->XSBCNT FOR 1ST PACKET
 5682  010740  010065   000104   1$:   MOV     R0,PKPTR(R5)    ;POINT TO PACKET
 5683  010744  111037   003307         MOVB    @R0,SYSTAT+1    ;SAVE RCV'D BYTE
 5684  010750  011137   003316         MOV     @R1,RCBCNT      ;GET COUNT
 5685  010754  011337   003314         MOV     @R3,RCFLG       ;AND FLAG
 5686  010760  121013                  CMPB    @R0,@R3         ;1ST BYTE=EXPECTED?
 5687  010762  001057                  BNE     5$              ;UH OH...
 5688  010764  121027   000020         CMPB    @R0,#RSCONT     ;OK, IS IT 1 BYTE?
 5689  010770  001534                  BEQ     7$              ;YES...ONTO NEXT PACK
 5690                                                          ;NO, SO > 1 BYTE (NEVER EXPECT INIT!)
 5691  010772  013704   003316         MOV     RCBCNT,R4       ;EXPECTED, SO COUNT MUST BE RIGHT
 5692  010776  005744                  TST     -(R4)           ;ADJUST FROM RECEIVE COUNT TO COUNT FOR CHECKSUM
 5693  011000  004737   013750         CALL    CKCKSM          ;CHECK CHECKSUM
 5694  011004  103005                  BCC     2$              ;NO CARRY...NO INCORRECT
 5695  011006  012704   000022         MOV     #BDCHK,R4       ;ERROR
 5696  011012  004737   012634         CALL    LOG             ;LOG IT
 5697  011016  000521                  BR      7$              ;ON TO NEXT PACK
 5698  011020  122710   000002   2$:   CMPB    #RSEND,(R0)     ;END PAK?
 5699  011024  001005                  BNE     3$              ;NO
 5700  011026  004737   011302         CALL    CHKEND          ;YES-CHECK
 5701  011032  012702   000001         MOV     #1,R2           ;LAST PACKET
 5702  011036  000511                  BR      7$              ;AND FALL THROUGH
 5703  011040  122710   000001   3$:   CMPB    #RSDATA,@R0     ;DATA PAK?
 5704  011044  001012                  BNE     4$              ;NO
```

```
 5705  011046  022737  000001  003342          CMP     #1,TEST9        ;***** IS THIS TEST 9
 5706  011054  001003                          BNE     11$             ;***** NO-CONTTINUE NORMALLY
 5707  011056  005065  000210                  CLR     MRSP(R5)        ;***** CLR MRSP FLAG
 5708  011062  000402                          BR      12$             ;***** SKIP INSTR
 5709  011064  004737  014550          11$:    CALL    COMPAR          ;YES-CHECK DATA
 5710  011070  000474                  12$:    BR      7$              ;ALL DONE?
 5711  011072  052715  020010          4$:     BIS     #BIT3!BIT13,@R5 ;SET 'BAD FLAG' RETRY FLAGS
 5712  011076  012704  000010                  MOV     #OTL,R4         ;OUT TO LUNCH
 5713  011102  005765  000074                  TST     DLV(R5)         ;AH,BUT DLV ERROR?
 5714  011106  001402                          BEQ     20$             ;NO
 5715  011110  012704  000012                  MOV     #OVRN,R4        ;YES-USE CORRECT ERROR #
 5716  011114  004737  012634          20$:    CALL    LOG             ;TALLY
 5717  011120  000467                          BR      8$              ;DONE
 5718
 5719                                  ;HERE CHECKS UNEXPECTED RESPONSE
 5720
 5721  011122  122710  000004          5$:     CMPB    #RSINIT,@R0     ;INIT?
 5722  011126  001007                          BNE     6$              ;NO
 5723  011130  052715  020010                  BIS     #BIT3!BIT13,@R5 ;YES-SET RETRY FLAGS
 5724  011134  012704  000006                  MOV     #RCINIT,R4      ; WE GOT AN INIT
 5725  011140  004737  012634                  CALL    LOG             ;TALLY IT
 5726  011144  000455                          BR      8$              ;DONE
 5727  011146  122710  000001          6$:     CMPB    #RSDATA,@R0     ;DATA PAK?
 5728  011152  001013                          BNE     9$              ;NO
 5729  011154  012704  000204                  MOV     #RSDASZ,R4      ;YES, USE DATA SIZE
 5730  011160  005744                          TST     -(R4)           ;ADJUST FOR CHKSUM
 5731  011162  004737  013750                  CALL    CKCKSM          ;AND CHECK
 5732  011166  103430                          BCS     10$             ;GOOF
 5733  011170  004737  014550                  CALL    COMPAR          ;OK, HOW'S THE DATA?
 5734                                                                  ;EXPECTED END, GOT
 5735                                                                  ;DATA + END.
 5736  011174  062700  000204                  ADD     #RSDASZ,R0      ;POINT TO END PACK
 5737  011200  000657                          BR      1$              ;CHECK IT, USE SAME XSFLG
 5738
 5739  011202  122710  000002          9$:     CMPB    #RSEND,(R0)     ;END?
 5740  011206  001331                          BNE     4$              ;NO-OUT TO LUNCH
 5741  011210  012704  000016                  MOV     #RSSNSZ,R4      ;YES, TOTAL SIZE MINUS
 5742  011214  005744                          TST     -(R4)           ;TWO (THE CHKSUM)
 5743  011216  004737  013750                  CALL    CKCKSM          ;CHECK IT
 5744  011222  103412                          BCS     10$             ;OOPS
 5745  011224  022737  000001  003342          CMP     #1,TEST9        ;***** IS THIS TEST 9
 5746  011232  001003                          BNE     13$             ;***** NO-CONTINUE NORMALLY
 5747  011234  012765  000001  000210          MOV     #1,MRSP(R5)     ;***** IF SO, SET THE MRSP FLAG
 5748  011242  004737  011302          13$:    CALL    CHKEND          ;OK,NOW TEST SUC. CODE
 5749
 5750  011246  000414                          BR      8$              ;ALL DONE
 5751
 5752  011250  012704  000022          10$:    MOV     #BDCHK,R4       ;CHECKSUM ERROR
 5753  011254  004737  012634                  CALL    LOG
 5754  011260  000407                          BR      8$              ;EXIT
 5755
 5756  011262  005302                  7$:     DEC     R2              ;ANY PACKETS LEFT TO CHECK?
 5757  011264  001405                          BEQ     8$              ;NO, ALL DONE
 5758  011266  063700  003316                  ADD     RCBCNT,R0       ;YES, POINT TO NEXT PACKET
 5759  011272  022121                          CMP     (R1)+,(R1)+     ;POINT TO NEXT EXPECTED COUNT
 5760  011274  022323                          CMP     (R3)+,(R3)+     ;AND EXPECTED FLAG
```

```
 5761  011276  000620                        BR      1$              ;TRY ANOTHER,THEY'RE SMALL
 5762  011300  000207                8$:     RETURN                  ;RETURN
```

```
                        .SBTTL  CHKEND / CHECK SUCCESS AND DETERMINE RETRY STATUS /

                        ;++
                        ; CHKEND - IF RETRYING;DETERMINE IF DATA ERROR OR BAD FLAG BYTE ERROR RETRY.
                        ;
                        ;       IF RETRYING BAD FLAG:RESET RETRY FLAG(SINCE OPERATION IS COMPLETE),
                        ;       AND CHECK SUCCESS CODE.
                        ;        IF RETRYING DATA ERROR ; CHECK SUCCESS CODE AND IF 0, PRINT RECOVERED,
                        ;          SOFT ERROR, END RETRY STATUS.  IF NOT 0 AND WAS STILL 'DATA
                        ;          CHECK' ERROR - DETERMINE WHETHER TO CONTINUE ANOTHER RETRY OR
                        ;          LOG 'UNRECOVERABLE' ERROR.
                        ;
                        ;          IF NOT RETRYING DATA ERROR;  CHECK IF 'DATA CHECK' ERROR SUCCESS CODE
                        ;          AND IF SO,START RETRY, ELSE EXIT.
                        ; INPUTS:  IMPLIED  UNITS DATA BLOCK
                        ; OUTPUTS: RETRY (SYSTAT BIT 1AND 2), (BIT10 @R5) RESET IF RETRYING.
                        ;          -  DATA COMARE ERROR (BIT6 @R5) CLEARED.
                        ;          -   REDUCED/NORMAL GAIN (BIT7 @R5) ADJUSTED
                        ;--

011302  000240          CHKEND:: NOP
011304                          PUSH    R0                      ;R0 --> END PAK
(1)  011304  010046                                             MOV     R0,-(SP)
(1)
(1)
011306                          PUSH    R4
(1)  011306  010446                                             MOV     R4,-(SP)
(1)
(1)
011310  032737  000006  003306  1$:     BIT     #BIT1!BIT2,SYSTAT       ;RETRYING?
011316  001406                  BEQ     NOREE                   ;NO-CHECK NORMALLY
011320  032737  000004  003306          BIT     #BIT2,SYSTAT            ;IS IT BAD FLAG TYPE?
011326  001454                  BEQ     CHKREE                  ;NO(DATA TYPE)
011330  042715  020000          BIC     #BIT13,@R5              ;YES, SO IF END PACK THEN RETRY'S COMPLETE
011334  004737  012320  NOREE:  CALL    CHKSUC                  ;CHECK SUCCESS CODE
011340  032715  100000          BIT     #BIT15,@R5              ;ABORTED?
011344  001402                  BEQ     3$                      ;NO,CONTINUE
011346  000137  012024          JMP     CHKRET                  ;YES,EXIT
011352  105765  000077  3$:     TSTB    SUCC$+1(R5)             ;NO; HOW'D WE DO?
011356  001015                  BNE     CHKERR                  ;NOT SO GOOD.
011360  032715  000100          BIT     #BIT6,@R5               ;OK, HOST FIND DATA PAK ERROR?
011364  001002                  BNE     2$                      ;YES
011366  000137  012024          JMP     CHKRET                  ;NO
011372  012704  000014  2$:     MOV     #BDCOM,R4               ;YES:  JUST BAD DATA-NO DATACHK ERR
011376  004737  012634          CALL    LOG                     ;BAD DATA IN PACKET
011402  000137  012024          JMP     CHKRET                  ;QUIT
011406  032715  001000  CHKERR: BIT     #BIT9,@R5               ;BAD SUCCESS; TU DATA CHK ERROR?
011412  001002                  BNE     1$                      ;YES
011414  000137  012024          JMP     CHKRET                  ;NO, ALL DONE.
011420  052715  002000  1$:     BIS     #BIT10,@R5              ;YES-START RETRY
011424  012765  000001  000002  MOV     #1,RETRY(R5)            ;CALL IT 1ST
011432                          PRINTX  #RTRYN,RETRY(R5)        ;** PRINT **
(8)  011432  016546  000002                                             MOV     RETRY(R5
(7)  011436  012746  012204                                             MOV     #RTRYN,-
(6)  011442  012746  000002                                             MOV     #2,-(SP)
(5)  011446  010600                                             MOV     SP,R0
```

```
   (4)  011450  104415                                                                      TRAP    C$PNTX
   (4)  011452  062706  000006                                                              ADD     #6,SP
  5811  011456  000562                      BR      CHKRET          ;ALL DONE
  5812  011460  004737  012320    CHKREE:   CALL    CHKSUC          ;CHECK SUCCESS CODE
  5813  011464  105765  000077              TSTB    SUCCS+1(R5)     ; SUCCESSFUL YET?
  5814  011470  001054                      BNE     UNSUC           ;NO, CHECK COUNT
  5815  011472                              PRINTX  #RECOV,RETRY(R5)
   (8)  011472  016546  000002                                                              MOV     RETRY(R5
   (7)  011476  012746  012044                                                              MOV     #RECOV,-
   (6)  011502  012746  000002                                                              MOV     #2,-(SP)
   (3)  011506  010600                                                                      MOV     SP,R0
   (4)  011510  104415                                                                      TRAP    C$PNTX
   (4)  011512  062706  000006                                                              ADD     #6,SP
  5816  011516  105715                      TSTB    (R5)            ;DETERMINE THRESHOLD
  5817  011520  100411                      BMI     2$              ;IT'S MODIFIED
  5818  011522                              PRINTX  #THRSLO         ;NORMAL
   (7)  011522  012746  012124                                                              MOV     #THRSLO,
   (6)  011526  012746  000001                                                              MOV     #1,-(SP)
   (3)  011532  010600                                                                      MOV     SP,R0
   (4)  011534  104415                                                                      TRAP    C$PNTX
   (4)  011536  062706  000004                                                              ADD     #4,SP
  5819  011542  000410                      BR      3$
  5820  011544                    2$:       PRINTX  #THRSHI         ;ENHANCED
   (7)  011544  012746  012152                                                              MOV     #THRSHI,
   (6)  011550  012746  000001                                                              MOV     #1,-(SP)
   (3)  011554  010600                                                                      MOV     SP,R0
   (4)  011556  104415                                                                      TRAP    C$PNTX
   (4)  011560  062706  000004                                                              ADD     #4,SP
  5821  011564  032715  000400    3$:       BIT     #BIT8,@R5       ;WRITE OR READ OPERATION?
  5822  011570  001003                      BNE     4$              ;WRITE
  5823  011572  012704  000002              MOV     #SFTRD,R4       ;READ
  5824  011576  000402                      BR      5$
  5825  011600  012704  000004    4$:       MOV     #SFTWR,R4       ;WRITE
  5826  011604  004737  012634    5$:       CALL    LOG
  5827  011610  005065  000002              CLR     RETRY(R5)       ;RESTORE TO NORMAL STATE
  5828  011614  042715  002200              BIC     #BIT10!BIT7,@R5 ;NO RETRY, NORM THRESHOLD
  5829  011620  000501                      BR      CHKRET          ;QUIT
  5830
  5831  011622  000240            UNSUC:    NOP                     ;RETRYING:  SEE IF HARD YET
  5832  011624  032715  001000              BIT     #BIT9,@R5       ;TU DATA CHECK ERROR?
  5833  011630  001015                      BNE     2$              ;YES
  5834  011632                              PRINTB  #RETERR         ;NO-''OTHER-ERROR'' ERROR
   (7)  011632  012746  012246                                                              MOV     #RETERR,
   (6)  011636  012746  000001                                                              MOV     #1,-(SP)
   (3)  011642  010600                                                                      MOV     SP,R0
   (4)  011644  104414                                                                      TRAP    C$PNTB
   (4)  011646  062706  000004                                                              ADD     #4,SP
  5835  011652  005065  000002              CLR     RETRY(R5)       ;NO RETRIES
  5836  011656  042715  002200              BIC     #BIT10!BIT7,@R5 ;NO RETRY, NORM THRESHOLD
  5837  011662  000460                      BR      CHKRET          ;EXIT
  5838  011664  023765  003330  000002  2$: CMP     MXRTRY,RETRY(R5)  ;YES. DID WE GRADUATE TO HARD?
  5839  011672  001425                      BEQ     HRD1            ;YES
  5840  011674  005265  000002              INC     RETRY(R5)       ;NO.  JUST ANOTHER
  5841  011700                              PRINTX  #RTRYN,RETRY(R5)  ;PRINT OUT
   (8)  011700  016546  000002                                                              MOV     RETRY(R5
   (7)  011704  012746  012204                                                              MOV     #RTRYN,-
```

```
 (6)  011710  012746  000002                                                      MOV     #2,-(SP)
 (3)  011714  010600                                                              MOV     SP,R0
 (4)  011716  104415                                                              TRAP    C$PNTX
 (4)  011720  062706  000006                                                      ADD     #6,SP
      011724  032715  000200                     BIT     #BIT7,@R5       ;WAS NORMAL THRESHOLD?
      011730  001403                             BEQ     1$              ;YES-REDUCE GAIN
      011732  042715  000200                     BIC     #BIT7,@R5       ;NO-NORM
      011736  000432                             BR      CHKRET
      011740  052715  000200             1$:     BIS     #BIT7,@R5       ;REDUCED
      011744  000427                             BR      CHKRET          ;DONE
      011746  000240                     HRD1:   NOP                     ;HERE IS HARD ERROR!
      011750                                     PRINTX  #UNREC
 (7)  011750  012746  012224                                                      MOV     #UNREC,-
 (6)  011754  012746  000001                                                      MOV     #1,-(SP)
 (3)  011760  010600                                                              MOV     SP,R0
 (4)  011762  104415                                                              TRAP    C$PNTX
 (4)  011764  062706  000004                                                      ADD     #4,SP
      011770  032715  000400                     BIT     #BIT8,@R5       ;RD OR WR?
      011774  001003                             BNE     4$              ;WRITE
      011776  012704  000016                     MOV     #HRDRD,R4       ;READ
      012002  000402                             BR      5$              ;LOG IT
      012004  012704  000020             4$:     MOV     #HRDWR,R4       ;WRITE
      012010  004737  012634             5$:     CALL    LOG             ;LOG IT
      012014  005065  000002                     CLR     RETRY(R5)       ;BACK TO NORMAL
      012020  042715  002200                     BIC     #BIT10!BIT7,@R5 ;NO RETRY, NOT REDUCED

      012024  042737  000006  003306     CHKRET: BIC     #BIT1!BIT2,SYSTAT       ;NO SYSTEM RETRY NEXT PASS
      012032  042715  000100                     BIC     #BIT6,@R5       ;NO MORE HOST DATA CHECK ERROR
      012036                                     POP     R4
 (1)  012036  012604                                             MOV     (SP)+,R4
 (1)
      012040                                     POP     R0
 (1)  012040  012600                                             MOV     (SP)+,R0
 (1)
      012042  000207                             RETURN


      012044  040445  042522  047503     RECOV:  .ASCIZ  /%ARECOVERED FROM DATA CHECK ERROR RETRY # %D1%N/
                                                 .EVEN
      012124  040445  047040  051117     THRSLO: .ASCIZ  /%A NORMAL THRESHOLD%N/
                                                 .EVEN
      012152  040445  046440  042117     THRSHI: .ASCIZ  /%A MODIFIED THRESHOLD %N/
              012204                             .EVEN
      012204  040445  042522  051124     RTRYN:  .ASCIZ  /%ARETRY # %D1%N/
                                                 .EVEN
      012224  040445  047125  042522     UNREC:  .ASCIZ  /%AUNRECOVERABLE%N/
                                                 .EVEN
      012246  040445  052117  042510     RETERR: .ASCIZ  /%AOTHER ERROR DURING RETRY : EXIT RETRY%N/
                                                 .EVEN
```

```
                                        .SBTTL  CHKSUC / INTERPRET SUCCESS CODE /

                                        ;++
                                        ; CHKSUC - COPY SUCCESS CODE (BYTE) TO SUCCS+1(R5).  INTERPRET SUCCESS
                                        ;          AND IF NOT 0, LOG APPROPRIATE ERROR.
                                        ; INPUTS:  R0 POINTS TO END PACKET.
                                        ;          @R5 - UNIT STATUS WORD
                                        ;          CMDSNT(R5) - COMMAND BYTE
                                        ;
                                        ; OUTPUTS: R4 IS ERROR NUMBER IF ERROR.
                                        ;          SUCCS(R5) UPDATED.
                                        ;          BIT9 @R5 SET ON DATA CHECK SUCCESS CODE
                                        ;--

012320  000240                          CHKSUC:: NOP
012322  016065  000002  000076                   MOV     2(R0),SUCCS(R5) ;R0-->END PACKET
012330  122760  000000  000003                   CMPB    #ESOK,3(R0)     ;GET SUCCESS BYTE
012336  001535                                   BEQ     12$             ;COMPLETE SUCCESS-EXIT

012340  122760  000001  000003                   CMPB    #ESTRY,3(R0)    ;OK BUT RETRIES?
012346  001012                                   BNE     20$             ;NO
012350  126527  000100  000002                   CMPB    CMDSNT(R5),#RSSRD;A READ?
012356  001001                                   BNE     22$             ;NO

012360  000520                                   BR      10$             ;NO RETRIES IN MAINTENANCE!
012362  126527  000100  000003          22$:     CMPB    CMDSNT(R5),#RSSWR       ;A WRITE?
012370  001001                                   BNE     20$             ;NO
012372  000513                                   BR      10$             ;LOG IT
012374  122760  177737  000003          20$:     CMPB    #ESNOMO,3(R0)   ;NO MOTOR?
012402  001003                                   BNE     1$              ;NO
012404  012704  000030                           MOV     #NOMOT,R4       ;YES-
012410  000506                                   BR      11$             ;LOG

012412  122760  177757  000003          1$:      CMPB    #ESCKS,3(R0)    ;"DATA CHECK" ERROR?
012420  001003                                   BNE     2$              ;NO
012422  052715  001000                           BIS     #BIT9,@R5       ;SET DATA-CHK-ERROR FLAG
012426  000501                                   BR      12$             ;DONT LOG

012430  126527  000100  000007          2$:      CMPB    CMDSNT(R5),#RSSSLF      ;SELF TEST?
012436  001006                                   BNE     3$              ;NOPE
012440  105760  000003                           TSTB    3(R0)           ;YES, NEG. IF ERROR
012444  100072                                   BPL     12$             ;OK

012446  012704  000044                           MOV     #SLFER,R4       ;YES-ERROR
012452  000465                                   BR      11$             ;LOG IT

012454  122760  177740  000003          3$:      CMPB    #ESSK,3(R0)     ;SEEK ERROR?
012462  001005                                   BNE     4$              ;NO
012464  012704  000024                           MOV     #SKERR,R4       ;YES-
012470  052715  040000                           BIS     #BIT14,@R5      ;SET 'DOBRK' FLAG  *** REV E *** MISSING "@"
012474  000454                                   BR      11$             ;LOG

012476  122760  177767  000003          4$:      CMPB    #ESNCRT,3(R0)   ;NO CART?
012504  001003                                   BNE     5$              ;NO
012506  012704  000054                           MOV     #NCART,R4       ;YES-
012512  000445                                   BR      11$             ;LOG
```

```
5936
5937  012514  122760  177720  000003      5$:   CMPB   #ESCMD,3(R0)     ;NO UNDERSTAND HOST?
5938  012522  001003                            BNE    6$               ;NO
5939  012524  012704  000040                    MOV    #CMNDER,R4       ;YES-
5940  012530  000436                            BR     11$              ;LOG
5941
5942  012532  122760  177770  000003      6$:   CMPB   #ESNONX,3(R0)    ;NON EXISTENT UNIT?
5943  012540  001003                            BNE    7$               ;NO
5944  012542  012704  000036                    MOV    #NOUNIT,R4       ;YES-
5945  012546  000427                            BR     11$              ;LOG
5946
5947  012550  122760  177765  000003      7$:   CMPB   #ESWLOC,3(R0)    ;WRITE LOCKED?
5948  012556  001003                            BNE    8$               ;NO
5949  012560  012704  000026                    MOV    #WRLOCK,R4       ;YES-
5950  012564  000420                            BR     11$              ;LOG
5951
5952  012566  122760  177776  000003      8$:   CMPB   #ESPART,3(R0)    ;PARTIAL OP?
5953  012574  001003                            BNE    9$               ;NO
5954  012576  012704  000034                    MOV    #PARTL,R4        ;YES-
5955  012602  000411                            BR     11$              ;LOG
5956
5957  012604  122760  177711  000003      9$:   CMPB   #ESREC,3(R0)     ;WRONG RECORD?
5958  012612  001003                            BNE    10$              ;NO
5959  012614  012704  000042                    MOV    #RECERR,R4       ;YES-
5960  012620  000402                            BR     11$              ;LOG
5961
5962  012622  012704  000046      10$:  MOV    #SUCOTL,R4       ;UNDEFINED
5963  012626  004737  012634      11$:  CALL   LOG              ;LOG ERROR
5964  012632  000207              12$:  RETURN                  ;RETURN
```

```
                        .SBTTL  LOG / TO LOG ERROR IN CORRECT PLACE

                        ;++
                        ; LOG - DETERMINE IF ERROR IS FATAL, NON-FATAL OR FATAL AFTER N TRIES
                        ;       BY INDEX (ERROR #) INTO DEVICE DATA BLOCK.  ADD THE DRIVE # TO
                        ;       INDICATE UPPER OR LOWER BYTE AND INCREMENT THAT ERROR UNLESS
                        ;       THAT BYTE WOULD OVERFLOW.  DETERMINE IF EVL FLAG SET,AND IF SO,
                        ;       CHECK THRESHOLD (EVLTHR) AND  PRINT APPROPRIATE ERROR MESSAGE
                        ;       DESCRIPTION.  ABORT THE UNIT IF INDICATED THROUGH DODROP CODE.
                        ; INPUTS:  R4 = ERROR CODE
                        ; OUTPUTS: ABNDX(R5) = ERROR CODE.
                        ;           DLV(R5) = 0
                        ;         LSLUN = UNIT NUMBER
                        ;--

012634                  LOG::   PUSH    R0
012634  010046                                 MOV     R0,-(SP)
012636                          PUSH    R1
012636  010146                                 MOV     R1,-(SP)
012640                          PUSH    R3
012640  010346                                 MOV     R3,-(SP)
012642                          PUSH    R4
012642  010446                                 MOV     R4,-(SP)

012644  011537  002074                        MOV     @R5,L$LUN       ;GET UNIT NUMBER
012650  042737  177770  002074                BIC     #177770,L$LUN   ;MASK IT OFF
012656  010465  000004                        MOV     R4,ABNDX(R5)    ;SAVE INDEX IN CASE OF ABORT MESSAGE
012662  012703  000120                        MOV     #LGOFST,R3      ;OFFSET TO LOW ORDER BYTE (DRIVE0)
012666  060403                              ADD     R4,R3           ;FORM INDEX OF PARAM. TO UPDATE
012670  060503                              ADD     R5,R3           ;FORM ABSOLUTE ADDR. THIS UNIT
012672  004737  013640                        CALL    WHCHDR          ;SEE WHICH DRIVE T'WAS
012676  103001                              BCC     2$              ;WAS DRIVE 0
012700  005203                              INC     R3              ;DRIVE 1; POINT TO UPPER BYTE
012702  122713  000377              2$:     CMPB    #255.,@R3       ;POTENTIAL OVERFLOW POSSIBLE?
012706  001005                              BNE     LOGOK           ;NO
012710                          LOGO:   ERRDF   0.,OVRFLO,ERRDES  ;YES
012710  104455                                                              TRAP    C$ERDF
012712  000000                                                              .WORD   0
012714  013534                                                              .WORD   OVRFLO
012716  013170                                                              .WORD   ERRDES
012720  000512                          BR      ABO             ;ABORT UNIT
012722  105213                  LOGOK:  INCB    @R3             ;INCREMENT THE ERROR
012724  111304                          MOVB    @R3,R4          ;TEMP'LY SAVE IT
012726  016503  000004                        MOV     ABNDX(R5),R3    ;GET INDEX AGAIN
012732  012701  002226                        MOV     #RSNTAB,R1      ;FORM ADRS OF MSG
012736  066501  000004                        ADD     ABNDX(R5),R1    ;LIKE THIS
012742  042701  000001                        BIC     #BIT0,R1        ;INSURE WORD BOUNDARY
012746  032737  000004  016752                BIT     #EVL,FLGLOC     ;EVL SELECTED?
```

```
6007  012754  001414
6008  012756  123704  002222          BEQ     LOGOK2          ;NO-CONT
6009  012762  101011                  CMPB    EVLTHR,R4       ;YES,OVER THRESHOLD?
6010  012764  010337  012776          BHI     LOGOK2          ;NO
6011  012770  011137  013000          MOV     R3,DFTL1+2      ;YES,LOAD ERROR #
6012  012774                          MOV     @R1,DFTL1+4     ;AND MESSAGE ADDR
DFTL1:  ERRDF   0,DFTL1,ERRDES        ;ERROR
  (4) 012774  104455
  (5) 012776  000000                                                    TRAP    C$ERDF
  (5) 013000  012774                                                    .WORD   0
  (5) 013002  013170                                                    .WORD   DFTL1
6013  013004  000460                                                    .WORD   ERRDES
                        BR      ABO             ;DROP IT
6014  013006  120327  000014  LOGOK2: CMPB    R3,#BDCOM       ;'NEVER FATAL' TYPE?
6015  013012  103011                  BHIS    NTSFT           ;NO
6016  013014  010337  013026          MOV     R3,LOG1+2       ;YES, ERROR CODE
6017  013020  011137  013030          MOV     @R1,LOG1+4      ;DESCRIPTION
6018  013024  LOG1:   ERRSOFT 0.,LOG1,ERRDES
  (4) 013024  104457
  (5) 013026  000000                                                    TRAP    C$ERSOFT
  (5) 013030  013024                                                    .WORD   0
  (5) 013032  013170                                                    .WORD   LOG1
6019  013034  000450                                                    .WORD   ERRDES
                        BR      LOGO            ;EXIT
6020
6021  013036  120327  000026  NTSFT:  CMPB    R3,#WRLOCK      ;ONE TRY?
6022  013042  103411                  BLO     MABEE           ;NO, MAYBE A MULTIPLE
6023  013044  010337  013056          MOV     R3,LOG2+2.      ;YES
6024  013050  011137  013060          MOV     @R1,LOG2+4
6025  013054  LOG2:   ERRHRD  0,LOG2,ERRDES  ;PRINT HARD MESSAGE
  (4) 013054  104456
  (5) 013056  000000                                                    TRAP    C$ERHRD
  (5) 013060  013054                                                    .WORD   0
  (5) 013062  013170                                                    .WORD   LOG2
6026  013064  000430                                                    .WORD   ERRDES
                        BR      ABO             ;DROP UNIT
6027
6028  013066  042704  177400  MABEE:  BIC     #177400,R4      ;NEGATE SIGN EXTEND
6029  013072  163704  003320  1$:     SUB     FTLNM,R4        ;SEE IF MULTIPLE OF
6030  013076  001413                  BEQ     HRD             ;FTLNM-YES!
6031  013100  103401                  BLO     SFT             ;NO
6032  013102  000773                  BR      1$              ;NOT THERE YET
6033
6034  013104  010337  013116  SFT:    MOV     R3,LOG3+2       ;ERROR CODE
6035  013110  011137  013120          MOV     @R1,LOG3+4      ;DESCRIPTION
6036  013114  LOG3:   ERRSOFT 0,LOG3,ERRDES
  (4) 013114  104457                                                    TRAP    C$ERSOFT
  (5) 013116  000000                                                    .WORD   0
  (5) 013120  013114                                                    .WORD   LOG3
  (5) 013122  013170                                                    .WORD   ERRDES
6037  013124  000414                  BR      LOGO            ;EXIT
6038  013126  010337  013140  HRD:    MOV     R3,LOG3B+2      ;HARD ERROR CODE
6039  013132  011137  013142          MOV     @R1,LOG3B+4     ;DESCRIPTION
6040  013136  LOG3B:  ERRHRD  0,LOG3B,ERRDES
  (4) 013136  104456                                                    TRAP    C$ERHRD
  (5) 013140  000000                                                    .WORD   0
  (5) 013142  013136                                                    .WORD   LOG3B
  (5) 013144  013170                                                    .WORD   ERRDES
6041
6042  013146  011500  ABO:    MOV     @R5,R0          ;GET UNIT NUMBER
```

```
                                                        F 6
 6043  013150  042700  177770                    BIC     #177770,R0      ;UN-SIGN EXTEND
 6044  013154                                    DODU    R0              ;USE LOGICAL # TO DROP
   (3) 013154  104451
 6045  013156                                                            TRAP    C$DODU
   (1) 013156  012604                    LOGO:   POP     R4              ;RESTORE
   (1)                                           MOV     (SP)+,R4
 6046  013160
   (1) 013160  012603                            POP     R3
   (1)                                           MOV     (SP)+,R3
 6047  013162
   (1) 013162  012601                            POP     R1
   (1)                                           MOV     (SP)+,R1
 6048  013164
   (1) 013164  012600                            POP     R0
   (1)                                           MOV     (SP)+,R0
 6049  013166  000207                            RETURN                  ;RETURN
```

```
                        ;++
                        ; ERRDES - CONTAINS CODE FOR EXTENDED ERROR INFORMATION:  DRIVE #,
                        ;          BLOCK #, ETC.
                        ;--

                                BGNMSG  ERRDES          ;ERROR DESCRIPTION
                                                                        ERRDES::
                                PUSH    R0
                                                        MOV     R0,-(SP)

                                PUSH    R2
                                                        MOV     R2,-(SP)

                                CLR     R2              ;PRESET TO DATA TYPE
                                BIT     #BIT4,@R5       ;WHAT PACK TYPE?
                                BEQ     2$              ;DATA
                                INC     R2              ;COMMAND
                        2$:     PRINTB  #UNIT,<B,DR(R5)>,R2,<B,SYSTAT+1>
                                                                        CLR     -(SP)
                                                                        BISB    SYSTAT+1
                                                                        MOV     R2,-(SP)
                                                                        CLR     -(SP)
                                                                        BISB    DR(R5),(
                                                                        MOV     #UNIT,-(
                                                                        MOV     #4,-(SP)
                                                                        MOV     SP,R0
                                                                        TRAP    C$PNTB
                                                                        ADD     #12.SP
                                MOV     REC(R5),R0      ;RECORD NUMBER
                                MOV     PATTEN(R5),R2   ;DATA EXPECTED
                                PRINTB  #RECID,R0,<B,CMDSNT(R5)>,<B,R2>,<B,SUCCS+1(R5)>
                                                                        CLR     -(SP)
                                                                        BISB    SUCCS+1(
                                                                        CLR     -(SP)
                                                                        BISB    R2,(SP)
                                                                        CLR     -(SP)
                                                                        BISB    CMDSNT(R
                                                                        MOV     R0,-(SP)
                                                                        MOV     #RECID,-
                                                                        MOV     #5,-(SP)
                                                                        MOV     SP,R0
                                                                        TRAP    C$PNTB
                                                                        ADD     #14.SP
                                TST     DLV(R5)         ;DLV ERROR?
                                BEQ     3$              ;NO
                                PRINTB  #RECID2,DLV(R5) ;YES-PRINT
                                                                        MOV     DLV(R5),
                                                                        MOV     #RECID2,
                                                                        MOV     #2,-(SP)
                                                                        MOV     SP,R0
                                                                        TRAP    C$PNTB
                                                                        ADD     #6.SP
                                CLR     DLV(R5)         ;RESET
                        3$:     POP     R2              ;RESTORE
```

```
  (1) 013354 012602                                                 MOV     (SP)+,R2
  (1)
6073  013356
  (1) 013356 012600                                         POP     R0
  (1)                                                               MOV     (SP)+,R0
6074  013360                                                ENDMSG                  ;EXIT
  (3) 013360                                                                                    L10003:
  (3) 013360 104423                                                                                     TRAP    C$MSG
6075  013362 040445 051104 053111           UNIT::  .ASCIZ  /%ADRIVE#  %01%A  PAK SENT %01%A FLAG RCVD %03%N/
6076                                                  .EVEN
6077  013442 040445 046102 041517           RECID:: .ASCIZ  /%ABLOCK#  %04%A COMMAND %02%A EXPCTD %03%A SUCCESS %03%N/
6078         013534
6079  013534 040503 023516 020124           OVRFLO: .ASCIZ  /CAN'T UPDATE ERROR OR STATISTIC:OVERFLOW PENDING/
6080         013616                                   .EVEN
6081  013616 040445 051040 042103           RECID2: .ASCIZ  /%A RCDB WAS %06%N/
6082                                                  .EVEN
```

```
                                        .SBTTL  WHCHDR / SEE WHICH DRIVE IS ACTIVE

                                        ;++
                                        ; INPUTS:  DR(R5)
                                        ; OUTPUTS:  CARRY=DRIVE (1 OR 0)
                                        ;--

013640  000241                          WHCHDR:: CLC                    ;CLEAR CARRY

013642  105765  000060                        TSTB    DR(R5)          ;DR 0?
013646  001401                          BEQ     2$              ;YES
013650  000261                          SEC                     ;NO

013652  000207                  2$:     RETURN                  ;RETURN
```

```
                                .SBTTL  CHKSUM / FORM THE PACKET CHECKSUM

                                ;++
                                ; THE CHECKSUM IS A 16 BIT CHECKSUM WITH END-AROUND CARRY.
                                ;
                                ; INPUTS:  R0 -> (POINTS TO) TOP OF PACKET
                                ;          R1 = # OF BYTES
                                ; OUTPUTS:  R0 -> WHERE TO PUT CHECKSUM
                                ;           R1 = CHECKSUM
                                ;--

013654                          CHKSUM:: PUSH   R3
013654 010346                                                   MOV     R3,-(SP)
013656                                  PUSH    R2
013656 010246                                                   MOV     R2,-(SP)
013660 042737 000001 003306             BIC     #BIT0,SYSTAT    ;"CHECKSUM IS ODD" BIT
013666 032701 000001                    BIT     #BIT0,R1        ;AN ODD # OF BYTES?
013672 001403                           BEQ     1$              ;NO
013674 052737 000001 003306             BIS     #BIT0,SYSTAT    ;YES

013702 006001                   1$:     ROR     R1              ;/2 FOR WORDS

013704 005003                   2$:     CLR     R3              ;PREP CHECKSUM WORD

013706 062003                   3$:     ADD     (R0)+,R3        ;FORM SUM
013710 005503                           ADC     R3              ;WITH CARRY
013712 005301                           DEC     R1              ;MORE WORDS?
013714 001374                           BNE     3$              ;YES

013716 032737 000001 003306             BIT     #BIT0,SYSTAT    ;WAS IT ODD
013724 001405                           BEQ     4$              ;NO
013726 112002                           MOVB    (R0)+,R2        ;YES GET NEXT BYTE
013730 042702 177400                    BIC     #177400,R2      ;UN-SIGN EXTEND
013734 060203                           ADD     R2,R3           ;ADD IT IN
013736 005503                           ADC     R3              ;AND CARRY JUST IN CASE

013740 010301                   4$:     MOV     R3,R1           ;RETURN IT IN CORRECT PLACE
013742                                  POP     R2              ;RESTORE
013742 012602                                                   MOV     (SP)+,R2
013744                                  POP     R3
013744 012603                                                   MOV     (SP)+,R3
013746 000207                           RETURN                  ;RETURN
```

```
                                .SBTTL  CKCKSM / MODULE TO CHECK THE CHKSUMS

                                ;++
                                ; MAKE SURE THE CHECKSUM RECEIVED = THE CHECKSUM CALCULATED.
                                ; INPUTS:  R4 = THE PACKET BYTE COUNT
                                ;          R0 -> THE PACKET TOP
                                ; OUTPUTS:  CARRY SET IF CHECKSUM CALC'D DOES NOT EQUAL CHECKSUM SENT
                                ;          R0 -> THE PACKET TOP
                                ;--

013750                          CKCKSM:: PUSH   R1
  (1) 013750 010146                              MOV     R1,-(SP)
  (1)
  (1)
013752                                  PUSH    R0              ;SAVE
  (1) 013752 010046                             MOV     R0,-(SP)
  (1)
  (1)
013754 010401                           MOV     R4,R1           ;COPY BYTE COUNT TO CORRECT
013756 004737 013654                    CALL    CHKSUM          ;REGISTER FOR CHKSUM AND
                                                                ;FORM CHECKSUM

                                ;HERE R0 --> XMITTED CHKSUM, R1=CHKSUM CALC'D

013762 122001                           CMPB    (R0)+,R1        ;LOWER ORDER CHECK
013764 001005                           BNE     2$              ;WRONG

013766 000301                           SWAB    R1              ;OK-PREP FOR

013770 122001                           CMPB    (R0)+,R1        ;HIGH ORDER CHECK
013772 001002                           BNE     2$              ;WRONG
013774 000241                           CLC                     ;OK-CLEAR SAILING

013776 000401                           BR      3$              ;EXIT

014000 000261                   2$:     SEC                     ;LET ERROR BE KNOWN

014002                          3$:     POP     R0
  (1) 014002 012600                             MOV     (SP)+,R0
  (1)
014004                                  POP     R1
  (1) 014004 012601                             MOV     (SP)+,R1
  (1)
014006 000207                           RETURN                  ;RETURN
```

```
                                .SBTTL  DOBRK / MODULE TO INIT TU58 AND TEST INTERRUPTS

                                ;++
                                ; DOBRK - SEND RADIAL SERIAL "BREAK" TO DEVICE:
                                ;       - SET "BREAK" ON INTERFACE.
                                ;       - SEND 8. NULLS
                                ;       - CLEAR "BREAK" ON INTERFACE
                                ;       - SET VECTORS FOR RCV AND XMIT
                                ;       - SEND 2 BYTES OF "INIT"
                                ;       - RECEIVE "CONTINUE"
                                ;       - IF RECEIVE GARBAGE OR TIMEOUT - ERROR
                                ;       - CLEAR INTERRUPTS AND VECTORS
                                ; INPUTS:  @R5 BIT14 WAS SET - (SEND BREAK)
                                ; OUTPUTS: @R5 BIT14 CLEAR IF SUCCESSFUL INIT.
                                ;          SYSTAT+1 = RECEIVED BYTE
                                ;          ERRORS R4 = ERROR CODE:
                                ;                - SEND NOT READY TIMEOUT (TOSNDB)
                                ;                - NO RESPONSE
                                ;                - DLV ERROR
                                ;                - CAN'T INIT
                                ;--

014010  105037  014543          DOBRK:: CLRB    INITWD+1        ;CLEAR BYTE RECEIVE ADDR
014014  005037  014544                  CLR     BRKTO           ;CLEAR TIME OUT CONSTANT
014020  052775  000001  000026          BIS     #BIT0,@XMSR(R5) ;SET 'BREAK'
014026  012765  000001  000100          MOV     #RSSNIT,CMDSNT(R5)  ;SAY WE SENT 'INIT'
014034  052715  000020                  BIS     #BIT4,@R5       ;PAK SENT TYPE =COMMAND, SORT OF
014040  012704  000010                  MOV     #8.,R4          ;BREAK-IT'S-BACK COUNT=8
014044                          1$:     BREAK                   ;SUPERVISOR TAKE FIVE
  (3)   014044  104422                                                  TRAP    C$BRK
014046  105775  000026                  TSTB    @XMSR(R5)       ;FOR ^C CHECK, ETC.
014052  100410                          BMI     4$              ;READY?
014054  005337  014544                  DEC     BRKTO           ;YES
014060  001371                          BNE     1$              ;NO,TIME OUT?
014062  012704  000056                  MOV     #TOSNDB,R4      ;NO
014066  004737  012634                  CALL    LOG             ;YES, SET ERROR CODE
014072  000535                          BR      3$              ;LOG IT
014074  113775  014540  000030  4$:     MOVB    BRKWD,@XMDB(R5) ;EXIT
014102  005037  014544                  CLR     BRKTO           ;SEND NULL
014106  005304                          DEC     R4              ;RESET TIME OUT
014110  001355                          BNE     1$              ;MORE NULLS TO SEND?
014112  005375  000026                  DEC     @XMSR(R5)       ;YES
014116  017500  000024                  MOV     @RCDB(R5),R0    ;NO, CLEAR 'BREAK'
014122                                  SETPRI  #PRI00          ;HEAVE 'GARBAGE' 1ST BYTE
  (3)   014122  012700  000000                                  ;SET TO INTERRUPT FO SURE
  (3)   014126  104441                                                  MOV     #PRI00,R
014130                                                                  TRAP    C$SPRI
                                        SETVEC  TUVECT(R5),#RCVINT,#PRI07 ;SET VECTO INFO
  (7)   014130  012746  000340
  (6)   014134  012746  014450                                          MOV     #PRI07,-
  (5)   014140  016546  000204                                          MOV     #RCVINT,
  (4)   014144  012746  000003                                          MOV     TUVECT(R
  (3)   014150  104437                                                  MOV     #3,-(SP)
  (2)   014152  062706  000010                                          TRAP    C$SVEC
014156  062765  000004  000204                                          ADD     #10,SP
014164                          ADD     #4,TUVECT(R5)   ;AND INC TO SND VECTOR
                                SETVEC  TUVECT(R5),#SNDINT,#PRI07;AND SET IT
```

```
                                                                                        MOV     #PRIO7,-
                                                                                        MOV     #SNDINT,
                                                                                        MOV     TUVECT(R
                                                                                        MOV     #3,-(SP)
                                                                                        TRAP    C$$VEC
                                                                                        ADD     #10,SP
 (7)  014164  012746  000340
 (6)  014170  012746  014434
 (5)  014174  016546  000204
 (4)  014200  012746  000003
 (3)  014204  104437
 (2)  014206  062706  000010
6228  014212  162765  000004  000204        SUB     #4,TUVECT(R5)   ;RESET VECTOR ADDR.
6229  014220  005037  014544                CLR     BRKTO           ;RESET TIME OUT
6230  014224  012704  014542                MOV     #INITWD,R4      ;USE ADDR. FOR SNDBYT
6231  014230  010437  014546                MOV     R4,BRKPTR       ;AND SAVE FOR "WAIT"
6232  014234  052775  000100  000026        BIS     #BIT6,@XMSR(R5) ;ENABLE INTER.
6233  014242  004737  014504                CALL    WAIT            ;AND ENTER LOOP
6234  014246  005715                        TST     @R5             ;ABORTED FROM TIME OUT?
6235  014250  100446                        BMI     3$              ;YES-EXIT
6236
6237  014252  005037  014544                CLR     BRKTO           ;RESET TIME OUT
6238  014256  012704  014542                MOV     #INITWD,R4      ;SEND SECOND INIT
6239  014262  010437  014546                MOV     R4,BRKPTR       ;SAVE POINTER AGAIN
6240  014266  052775  000100  000026        BIS     #BIT6,@XMSR(R5) ;AND THEN ENABLE INT
6241  014274  004737  014504                CALL    WAIT            ;AND WAIT
6242  014300  005715                        TST     @R5             ;IF ABORTED
6243  014302  100431                        BMI     3$              ;THEN EXIT
6244
6245  014304  012704  014543                MOV     #INITWD+1,R4    ;WHERE RESPONSE WILL GO (ADDRESS)
6246  014310  010437  014546                MOV     R4,BRKPTR       ;AND FOR 'WAIT'
6247  014314  052775  000100  000022        BIS     #BIT6,@RCSR(R5) ;ENABLE RECIEVE INT.
6248  014322  004737  014504                CALL    WAIT            ;GET ANSWER
6249  014326  005715                        TST     @R5             ;ABORTED?
6250  014330  100416                        BMI     3$              ;YES.
6251
6252  014332  123727  014543  000020        CMPB    INITWD+1,#RSCONT  ;NO. IS IT 'CONTINUE'?
6253  014340  001003                        BNE     2$              ;NOPE-ERROR
6254
6255  014342  042715  040000                BIC     #BIT14,@R5      ;SUCCESSFUL. CLEAR DOBREAK FLAG
6256  014346  000407                        BR      3$              ;EXIT
6257
6258  014350  113737  014543  003307  2$:   MOVB    INITWD+1,SYSTAT+1       ;SAVE BUM RESPONSE
6259  014356  012704  000032                MOV     #CNINIT,R4              ;CAN'T INIT CODE
6260  014362  004737  012634                CALL    LOG             ;LOG IT
6261                                                                ;SCHEDULER WILL TRY AGAIN IF NOT ABORTED
6262
6263  014366  042775  000100  000026  3$:   BIC     #BIT6,@XMSR(R5) ;CLEAR INTERRUPTS
6264  014374  042775  000100  000022        BIC     #BIT6,@RCSR(R5) ; AND FOR RECIEVE
6265  014402                                CLRVEC  TUVECT(R5)      ;RELEASE RECIEVE VECT.
 (3)  014402  016500  000204                                                    MOV     TUVECT(R
 (3)  014406  104436                                                            TRAP    C$CVEC
6266  014410  062765  000004  000204        ADD     #4,TUVECT(R5)   ;AND GET SEND ADDR.
6267  014416                                CLRVEC  TUVECT(R5)      ;AND RELEASE IT
 (3)  014416  016500  000204                                                    MOV     TUVECT(R
 (3)  014422  104436                                                            TRAP    C$CVEC
6268  014424  162765  000004  000204        SUB     #4,TUVECT(R5)   ;RESTORE POINTER
6269  014432  000207                        RETURN                  ;RETURN
```

```
                        .SBTTL  INTERRUPT SERVICE ROUTINES AND TIMER

014434                  BGNSRV  SNDINT                  ;"SEND" INTERRUPT SERVICE:
014434                                                                  SNDINT::
014434  042775  000100  000026  SNDHND: BIC     #BIT6,@XMSR(R5) ;DISABLE INTERRUPT
014442  112475  000030                  MOVB    (R4)+,@XMDB(R5);OUTPUT BYTE
014446                  ENDSRV
014446                                                          L10004:
014446  000002                                                          RTI


014450                  BGNSRV  RCVINT                  ;"RCV" INTERRUPT SERVICE:
014450                                                                  RCVINT::
014450  042775  000100  000022  RCVHND: BIC     #BIT6,@RCSR(R5) ;DISABLE INTS
014456  017565  000024  000074          MOV     @RCDB(R5),DLV(R5)       ;SAVE WORD
014464  116524  000074          MOVB    DLV(R5),(R4)+   ;BYTE TO BUFFER
014470  005765  000074          TST     DLV(R5)         ;ERROR?
014474  100402                  BMI     10$             ;YES
014476  005065  000074          CLR     DLV(R5)         ;NO CLEAR ERROR
014502                  10$:
014502                  ENDSRV
014502                                                          L10005:
014502  000002                                                          RTI


014504  000240

                        WAIT:   NOP                     ;WAIT  LOOP FOR
                                                        ;INTERRUPT SERVICING
014506  020437  014546          CMP     R4,BRKPTR       ;IF=,THEN NO INTERRUPT
014512  001011                  BNE     1$              ;GOT ONE!
014514                          BREAK                   ;SUPERVISOR BREAK
014514  104422                                                  TRAP    C$BRK
014516                          BREAK                   ;KILL SOME TIME
014516  104422                                                  TRAP    C$BRK
014520  005337  014544          DEC     BRKTO           ;TIME OUT?
014524  001367                  BNE     WAIT            ;NO...CONT.
014526  012704  000050          MOV     #TORCVB,R4      ;YES LOAD ERROR #
014532  004737  012634          CALL    LOG             ;LOG IT
014536  000207                  1$:     RETURN          ;RETURN

014540  000000          BRKWD:  .WORD   0               ;NULL
014542      004         INITWD: .BYTE   RSINIT          ;INIT COMMAND
014543      000                 .BYTE   0               ;RSCONT IS EXPECTED HERE
014544  000000          BRKTO:  .WORD   0               ;TIME OUT
014546  000000          BRKPTR: .WORD   0               ;POINTER TO INITWD
```

```
                        .SBTTL  COMPAR/DATA COMPARISON MODULE

                        ;++
                        ; COMPAR - IF "COMPARE DATA" SELECTED, COMPARE EACH DATA BYTE OF PACKET
                        ;          TO PATTEN(R5).  SAVE NUMBER OF BYTES NOT CORRECT.  IF NOT
                        ;          0, PRINT SOFT ERROR AND TOTAL # WRONG BYTES.  SET 'BAD_DATA_
                        ;          IN_PACKET' BIT (BIT6 @R5) FOR HIGHER LEVEL MODULES.
                        ; INPUTS:  - (CMPDAT) FLAG TO NOT COMPARE (=1)
                        ;          - PKPTR(R5) POINTS TO DATA PACK.
                        ; OUTPUTS: BIT6 @R5 (BAD DATA FLAG) ADJUSTED.
                        ;          LSLUN - UNIT NUMBER
                        ;          PRNSIZ - SIZE OF PACKET
                        ;--

014550                  COMPAR:: PUSH   R0              ;COMPARE DATA IS DATA PACKET
014550  010046                   MOV    R0,-(SP)
(1)
(1)
014552                          PUSH   R4               ;TO PATTERN WRITTEN
014552  010446                   MOV    R4,-(SP)
(1)
(1)
014554                          PUSH   R1               ;USING BYTE COUNT IN PACKET
014554  010146                   MOV    R1,-(SP)
(1)
(1)
014556  005037  014726          CLR    BDBYTS          ;CLEAR TOTAL WRONG
014562  016504  000104          MOV    PKPTR(R5),R4    ;GET TOP OF PACKET
014566  005737  002216          TST    CMPDAT          ;COMPARE SELECTED?
014572  001451                  BEQ    4$              ;NO-EXIT
014574  005204                  INC    R4              ;YES, LOCATE COUNT
014576  111401                  MOVB   @R4,R1          ;GET IT
014600  042701  177400          BIC    #177400,R1      ;SIGN-UNEXTEND
                                                       ;MUST TEST BYTE-WISE...
014604  005204                  INC    R4              ;-->FIRST DATA  BYTE
014606  126524  000072  1$:     CMPB   PATTEN(R5),(R4)+  ;DATA-WHAT WAS EXPECTED?
014612  001402                  BEQ    2$              ;YES
014614  005237  014726          INC    BDBYTS          ;NO, INCREMENT TOTAL WRONG
014620  005301          2$:     DEC    R1              ;MORE LEFT?
014622  001371                  BNE    1$              ;YES
014624  005737  014726          TST    BDBYTS          ;ANY WRONG?
014630  001432                  BEQ    4$              ;NO
014632  011537  002074          MOV    @R5,LSLUN       ;GET UNIT NUMBER
014636  042737  177770  002074  BIC    #177770,LSLUN   ;MASK IT OFF
014644                          ERRSOFT 0.,MSBDA,ERRDES  ;YES-PRINT 'BAD DATA IN PACKET' ERROR
014644  104457                                                   TRAP    C$ERSOFT
014646  000000                                                   .WORD   0
014650  002340                                                   .WORD   MSBDA
014652  013170                                                   .WORD   ERRDES
014654                          PRINTB  #DESC,BDBYTS
014654  013746  014726                                           MOV     BDBYTS,-
014660  012746  014730                                           MOV     #DESC,-(
014664  012746  000002                                           MOV     #2.,-(SP)
014670  010600                                                   MOV     SP,R0
014672  104414                                                   TRAP    C$PNTB
014674  062706  000006                                           ADD     #6.SP
```

```
 6351  014700  052715  000100                    BIS     #BIT6,@R5       ;LET 'EM KNOW UPSTAIRS-BAD DATA FLAG
 6352  014704  012737  000204  003336            MOV     #132.,PRNSIZ    ;SIZE IS ONE DATA PACK
 6353  014712  004737  014764                    CALL    PRNPAK          ;AND PRINT THE PACKET
 6354  014716                            4$:     POP     R1              ;RESTORE
   (1) 014716  012601                                                    MOV     (SP)+,R1
   (1)
 6355  014720                                    POP     R4
   (1) 014720  012604                                                    MOV     (SP)+,R4
   (1)
 6356  014722                                    POP     R0
   (1) 014722  012600                                                    MOV     (SP)+,R0
   (1)
 6357
 6358  014724  000207                            RETURN
 6359
 6360  014726  000000                    BDBYTS: .WORD
 6361  014730  040445  047524  040524    DESC:   .ASCIZ  /%ATOTAL BAD BYTES= %D3%A.%N/
 6362                                            .EVEN
```

```
                                .SBTTL  PRNPAK/MODULE TO PRINT DATA PACKET

                                ;++
                                ; PRNPAK - IF PRINT DATA PACK_ON_ERROR SELECTED: PRINT EACH BYTE OF PACKET
                                ;          TO BY PKPTR(R5).
                                ; INPUTS:  PRNSIZ - # OF BYTES IN PACKET.
                                ; OUTPUTS: NONE
                                ;--

014764  000240                  PRNPAK:: NOP                    ;PRINTS 1 PACKET
                                                                ;PKPTR(R5)->TOP OF PACKET
                                                                ;PRNSIZ (PASSED)=BYTE COUNT
014766                                  PUSH    R0
014766  010046                                                  MOV     R0,-(SP)

014770                                  PUSH    R4
014770  010446                                                  MOV     R4,-(SP)

014772  105737  002214                    TSTB    PRBUF           ;PRINT PACKET SELECTED?
014776  001451                          BEQ     4$              ;NO
015000  016504  000104                  MOV     PKPTR(R5),R4    ;YES-GET TOP OF PACK
015004  012737  000020  015130  1$:     MOV     #16.,LNCNT      ;16 BYTES PER LINE
015012  112437  015132  2$:     MOVB    (R4)+,PRDAT     ;AVOID SIGN EXTEND
015016                                  PRINTF  #PRFORM,<B,PRDAT>  ;PRINT BYTE
015016  005046                                                          CLR     -(SP)
015020  153716  015132                                                  BISB    PRDAT,(S
015024  012746  015134                                                  MOV     #PRFORM,
015030  012746  000002                                                  MOV     #2,-(SP)
015034  010600                                                          MOV     SP,R0
015036  104417                                                          TRAP    C$PNTF
015040  062706  000006                                                  ADD     #6,SP
015044  005337  003336                    DEC     PRNSIZ          ;ONE LESS
015050  001414                          BEQ     3$              ;NO MORE
015052  005337  015130                  DEC     LNCNT           ;NEW LINE?
015056  001355                          BNE     2$              ;NOT YET
015060                                  PRINTF  #CARLF          ;YES
015060  012746  015144                                                  MOV     #CARLF,-
015064  012746  000001                                                  MOV     #1,-(SP)
015070  010600                                                          MOV     SP,R0
015072  104417                                                          TRAP    C$PNTF
015074  062706  000004                                                  ADD     #4,SP
015100  000741                          BR      1$              ;NEXT LINE
015102                          3$:     PRINTF  #CARLF          ;FINISH UP
015102  012746  015144                                                  MOV     #CARLF,-
015106  012746  000001                                                  MOV     #1,-(SP)
015112  010600                                                          MOV     SP,R0
015114  104417                                                          TRAP    C$PNTF
015116  062706  000004                                                  ADD     #4,SP
015122                          4$:     POP     R4
015122  012604                                          MOV     (SP)+,R4

015124                                  POP     R0
015124  012600                                          MOV     (SP)+,R0
```

```
 6394  015126  000207                          RETURN                  ;RETURN
 6395
 6396  015130  000000                  LNCNT:  .WORD
 6397  015132  000000                  PRDAT:  .WORD
 6398  015134  047445  022463  020101  PRFORM: .ASCIZ  /%03%A /
 6399          015144                          .EVEN
 6400  015144  047045     000          CARLF:  .ASCIZ  /%N/
 6401          015150                          .EVEN
 6402
 6403  015150                          ENDMOD
 6404
```

```
                                                .TITLE MISCELLANEOUS SECTIONS
                                                .SBTTL  REPORT CODING SECTION

        015150                                          BGNMOD

                                                ;++
                                                ; THE REPORT CODING SECTION CONTAINS THE
                                                ; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
                                                ;
                                                ;--
        015150                                          BGNRPT
  (3)   015150                                                                                  LSRPT::
        015150                                          PUSH    R0
  (1)   015150  010046                                                          MOV     R0,-(SP)
  (1)
  (1)
        015152                                          PUSH    R1
  (1)   015152  010146                                                          MOV     R1,-(SP)
  (1)
  (1)
        015154                                          PUSH    R2
  (1)   015154  010246                                                          MOV     R2,-(SP)
  (1)
  (1)
        015156                                          PUSH    R3
  (1)   015156  010346                                                          MOV     R3,-(SP)
  (1)
  (1)
        015160                                          PUSH    R4
  (1)   015160  010446                                                          MOV     R4,-(SP)
  (1)
  (1)
        015162                                          PUSH    R5
  (1)   015162  010546                                                          MOV     R5,-(SP)
  (1)
  (1)

        015164                                          BREAK
  (3)   015164  104422                                                                          TRAP    C$BRK
        015166  012737  003350  015576                  MOV     #BLKTBL,RPTR    ;GET 1ST DEVICE BLOCK
        015174                                          PRINTS  #STATHD         ;HEADER
  (7)   015174  012746  015600                                                                  MOV     #STATHD,
  (6)   015200  012746  000001                                                                  MOV     #1,-(SP)
  (3)   015204  010600                                                                          MOV     SP,R0
  (4)   015206  104416                                                                          TRAP    C$PNTS
  (4)   015210  062706  000004                                                                  ADD     #4,SP
        015214                                          BREAK                   ;^C CHECK
  (3)   015214  104422                                                                          TRAP    C$BRK
        015216                                          PRINTS  #STHD2          ;2ND HEADER
  (7)   015216  012746  016054                                                                  MOV     #STHD2,-
  (6)   015222  012746  000001                                                                  MOV     #1,-(SP)
  (3)   015226  010600                                                                          MOV     SP,R0
  (4)   015230  104416                                                                          TRAP    C$PNTS
  (4)   015232  062706  000004                                                                  ADD     #4,SP
        015236                                  1$:     BREAK                   ;^C CHECK
  (3)   015236  104422                                                                          TRAP    C$BRK
```

```
     015240  017705  000332              MOV     @RPTR,R5        ;GET DEVICE BLOCK
     015244  032715  004000              BIT     #BIT11,@R5      ;UNIT NOT TESTED?
     015250  001131                      BNE     2$              ;TRUE, DON'T PRINT STATISTICS
                                                                 ;OK TO PRINT
     015252  011537  015574              MOV     @R5,RLUN        ;SAVE STATUS WORD
     015256  042737  177770  015574      BIC     #177770,RLUN    ;MASK UNIT NUM.
     015264  116501  000122              MOVB    SOFTR(R5),R1    ;SOFTREAD
     015270  042701  177400              BIC     #177400,R1      ;SIGN-UNEXTEND
     015274  116502  000124              MOVB    SOFTW(R5),R2    ;SOFT WRITE
     015300  042702  177400              BIC     #177400,R2      ;
     015304  116503  000136              MOVB    HARDR(R5),R3    ;HARD READ
     015310  042703  177400              BIC     #177400,R3      ;
     015314  116504  000140              MOVB    HARDW(R5),R4    ;HARD WRITE
     015320  042704  177400              BIC     #177400,R4      ;
     015324                              PRINTS  #FMO,RLUN       ;SUMMARY/UNIT #
 (8) 015324  013746  015574                                                      MOV     RLUN,-(S
 (7) 015330  012746  015712                                                      MOV     #FMO,-(S
 (6) 015334  012746  000002                                                      MOV     #2,-(SP)
 (5) 015340  010600                                                              MOV     SP,R0
 (4) 015342  104416                                                              TRAP    C$PNTS
 (4) 015344  062706  000006                                                      ADD     #6,SP
     015350                              PRINTS  #FM,#0,WRTNO(R5),RDNO(R5),<B,BDATA(R5)>,R1,R2,R3,R4
(15) 015350  010446                                                              MOV     R4,-(SP)
(14) 015352  010346                                                              MOV     R3,-(SP)
(13) 015354  010246                                                              MOV     R2,-(SP)
(12) 015356  010146                                                              MOV     R1,-(SP)
(11) 015360  005046                                                              CLR     -(SP)
(11) 015362  156516  000134                                                      BISB    BDATA(R5
(10) 015366  016546  000114                                                      MOV     RDNO(R5
 (9) 015372  016546  000110                                                      MOV     WRTNO(R5
 (8) 015376  012746  000000                                                      MOV     #0,-(SP)
 (7) 015402  012746  015730                                                      MOV     #FM,-(SP
 (6) 015406  012746  000011                                                      MOV     #11,-(SP
 (5) 015412  010600                                                              MOV     SP,R0
 (4) 015414  104416                                                              TRAP    C$PNTS
 (4) 015416  062706  000024                                                      ADD     #24,SP
     015422  116501  000123              MOVB    SOFTR+1(R5),R1  ;SAME
     015426  042701  177400              BIC     #177400,R1      ;AS
     015432  116502  000125              MOVB    SOFTW+1(R5),R2  ;ABOVE
     015436  042702  177400              BIC     #177400,R2      ;THIS
     015442  116503  000137              MOVB    HARDR+1(R5),R3  ;TIME
     015446  042703  177400              BIC     #177400,R3      ;FOR
     015452  116504  000141              MOVB    HARDW+1(R5),R4  ;DRIVE
     015456  042704  177400              BIC     #177400,R4      ;ONE

     015462                              PRINTS  #FM,#1,WRTN1(R5),RDN1(R5),<B,BDATA+1(R5)>,R1,R2,R3,R4
(15) 015462  010446                                                              MOV     R4,-(SP)
(14) 015464  010346                                                              MOV     R3,-(SP)
(13) 015466  010246                                                              MOV     R2,-(SP)
(12) 015470  010146                                                              MOV     R1,-(SP)
(11) 015472  005046                                                              CLR     -(SP)
(11) 015474  156516  000135                                                      BISB    BDATA+1(
(10) 015500  016546  000116                                                      MOV     RDN1(R5)
 (9) 015504  016546  000112                                                      MOV     WRTN1(R5
 (8) 015510  012746  000001                                                      MOV     #1,-(SP)
 (7) 015514  012746  015730                                                      MOV     #FM,-(SP
```

```
                                                                                                                    SEQ 0085

                                                                                                            MOV     #11,-(SP
                                                                                                            MOV     SP,R0
                                                                                                            TRAP    C$PNTS
                                                                                                            ADD     #24,SP
 (6)  015520  012746  000011
 (3)  015524  010600
 (4)  015526  104416
 (4)  015530  062706  000024
6494  015534  023727  015576  003366        2$:     CMP     RPTR,#LSTDEV    ;ALL UNITS DONE?
6495  015542  103005                              BHIS    3$              ;YES
6496  015544  062737  000002  015576              ADD     #2,RPTR         ;NO-DO
6497
6498  015552  000137  015236                      JMP     1$              ;MORE UNITS
6499
6500  015556                                    3$:     POP     R5
 (1)  015556  012605                                                      MOV     (SP)+,R5
 (1)
6501  015560                                            POP     R4
 (1)  015560  012604                                                      MOV     (SP)+,R4
 (1)
6502  015562                                            POP     R3
 (1)  015562  012603                                                      MOV     (SP)+,R3
 (1)
6503  015564                                            POP     R2
 (1)  015564  012602                                                      MOV     (SP)+,R2
 (1)
6504  015566                                            POP     R1
 (1)  015566  012601                                                      MOV     (SP)+,R1
 (1)
6505  015570                                            POP     R0
 (1)  015570  012600                                                      MOV     (SP)+,R0
 (1)
6506  015572                                            ENDRPT
 (3)  015572
 (3)  015572  104425                                                                L10006:
                                                                                            TRAP    C$RPT
6507  015574  000000                            RLUN:   .WORD
6508  015576  000000                            RPTR:   .WORD
6509
6510  015600  047045  040445  020040          STATHD: .ASCII  /%N%A      DR BLKS WR  BLKS RD  BDPAK  /
6511  015646  041504  045510  051057                  .ASCIZ  @DCHK/RD DCHK/WR  DCHK/RD DCHK/WR%N@
6512          015712                                  .EVEN
6513  015712  040445  047125  052111          FMO:    .ASCIZ  /%AUNIT %D1%N/
6514          015730                                  .EVEN
6515
6516  015730  040445  020040  020040          FM:     .ASCII  /%A       %D1%A %D5%A.    %D5%A.    %D3%A.    /
6517  016004  042045  022463  027101                  .ASCIZ  /%D3%A.   %D3%A.    %D3%A.    %D3%A.%N/
6518          016054                                  .EVEN
6519  016054  040445  020040  020040          STHD2:  .ASCII  /%A                                       /
6520  016121     122  041505  053117                  .ASCIZ  /RECOV   RECOV   UNRECOV UNRECOV%N/
6521          016164                                  .EVEN
6522  016164                                          ENDMOD
```

```
                        .SBTTL  INITIALIZE SECTION

                        ;++
                        ; THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
                        ; AT THE BEGINNING OF EACH PASS.
                        ;--

                                BGNINIT
                                                                        L$INIT::
                        INIT:   NOP                     ;
                                CLRB    STRT            ;FOR STATS CLEAR
                                CLR     TEST9           ;***** CLR TST 9 FLAG
                                READEF  #EF.START       ;START COMMAND?
                                                                        MOV     #EF.STAR
                                                                        TRAP    C$REFG
                                BNCOMPLETE INIT2        ;NO
                                                                        BCC     INIT2
                                INC     STRT            ;YES, SET START FLAG
                        INIT2:  MOV     #BLKTBL,DEVPTR  ;SET ALL UNITS ABORTED:
                                CLR     R4              ;UNIT NUMBER
                        1$:     MOV     @DEVPTR,R5      ;GET POINTER
                                MOV     R4,@R5          ;INSERT UNIT #
                                BIS     #BIT15,@R5        ;SET ABORTED
                                BIS     #BIT11,@R5      ;SET UNIT NOT TESTED
                                ASL     R4              ;*2 FOR LOOK-UP
                                MOV     BUFTBL(R4),RCVBUF(R5)   ;SETUP POINTER TO UNIT'S BUFFER
                                ASR     R4              ;CORRECT BACK TO UNIT #
                                CMP     DEVPTR,#LSTDEV  ;LAST DEVICE DONE?
                                BHIS    CHECK           ;YES
                                ADD     #2,DEVPTR       ;NO-GET
                                INC     R4              ;NEXT DEVICE AND
                                BR      1$              ;SERVICE

                        CHECK:  CMP     #8.,L$UNIT      ;MAKE SURE NOT
                                BHIS    GETHRD          ;TOO MANY UNITS
                                ERRSF   101.,TOMANY             ;TOMANY-REQUEST ^C
                                                                        TRAP    C$ERSF
                                                                        .WORD   101
                                                                        .WORD   TOMANY
                                                                        .WORD   0
                                DOCLN                   ;EXIT
                                                                        TRAP    C$DCLN

                        GETHRD: MOV     #BLKTBL,DEVPTR  ;INIT TABLE POINTER
                                CLR     R4              ;CLEAR DEVICE COUNTER
                        1$:     MOV     @DEVPTR,R5      ;GET STATUS WORD
                                MOV     R4,L$LUN        ;UNIT NUM. IN CASE ERROR
                                GPHARD  R4,R2           ;GET HARD INFO
                                                                        MOV     R4,R0
                                                                        TRAP    C$GPHRD
                                                                        MOV     R0,R2
                                BNCOMPLETE 3$
                                                                        BCC     3$
                                BIC     #BIT11,@R5      ;UNIT IS TESTED!
                                MOV     (R2)+,R3        ;R3=CSR
```

```
                                        MOV     (R2)+,TUVECT(R5)  ;GET VECTOR ADDRESS
                                        MOVB    (R2)+,DR+1(R5)  ;SAVE UNIT SUMMARY
                                        INC     R2              ;GET TO WORD BOUND
                                        MOV     (R2)+,PDTFLG    ;AND GET PDT FLAG
                                        BIS     #BIT14,@R5      ;SET SEND BREAK FLAG
                                        BIT     #BIT8,DR(R5)    ;DRIVE 0?
                                        BNE     13$             ;YES
                                        BIT     #BIT9,DR(R5)    ;DRIVE 1?
                                        BNE     13$             ;OK
                                        ERRSF   102.,NODRVS             ;NEITHER?!
                                                                                TRAP    CSERSF
                                                                                .WORD   102
                                                                                .WORD   NODRVS
                                                                                .WORD   0
 6581  016426                           DOCLN                   ;EXIT
    (3) 016426  104444                                                          TRAP    CSDCLN
 6582
 6583  016430  105737  016746          13$:    TSTB    STRT            ;START COMMAND?
 6584  016434  001412                          BEQ     14$             ;NO, DONT CLEAR
 6585                                                                  ;YES-CLEAR STATS
 6586  016436  012702  000202                  MOV     #BLKEND,R2      ;R2-->END OF STATS
 6587  016442  012701  000110                  MOV     #WRTNO,R1       ;FORM ADDRESS OF START:
 6588  016446  060501                          ADD     R5,R1           ;R1-->START OF STATS.
 6589  016450  162702  000110                  SUB     #WRTNO,R2       ;FORM # TO CLEAR
 6590
 6591  016454  105021                  2$:     CLRB    (R1)+           ;CLEAR 'EM
 6592  016456  005302                          DEC     R2              ;MORE?
 6593  016460  001375                          BNE     2$              ;YES
 6594  016462  042715  100000          14$:    BIC     #BIT15,@R5        ;SET NOT ABORTED
 6595  016466  010365  000022                  MOV     R3,RCSR(R5)     ;GET DEVICE REGISTERS:
 6596  016472  062703  000002                  ADD     #2,R3
 6597  016476  010365  000024                  MOV     R3,RCDB(R5)
 6598  016502  062703  000002                  ADD     #2,R3
 6599  016506  010365  000026                  MOV     R3,XMSR(R5)
 6600  016512  062703  000002                  ADD     #2,R3
 6601  016516  105737  016750                  TSTB    PDTFLG          ;UNIT A PDT?
 6602  016522  001402                          BEQ     4$              ;NO
 6603  016524  162703  000004                  SUB     #4,R3           ;YES...RCDB=XMDB
 6604  016530  010365  000030          4$:     MOV     R3,XMDB(R5)
 6605  016534  005065  000072                  CLR     PATTEN(R5)      ;ZERO DATA PATTERN
 6606  016540  005065  000002                  CLR     RETRY(R5)       ;NO RETRIES
 6607  016544  005065  000064                  CLR     REC(R5)         ;NO RECORD
 6608  016550  005065  000076                  CLR     SUCCS(R5)       ;NO SUCCESS
 6609  016554  005065  000074                  CLR     DLV(R5)         ;NO DLV ERROR
 6610  016560  005065  000210                  CLR     MRSP(R5)        ;***** CLR MRSP INDICATOR
 6611  016564  005037  003340                  CLR     ALLGON          ;OK TO PRINT STATISTICS
 6612  016570  062737  000002  003312  3$:     ADD     #2,DEVPTR       ;-->NEXT DEVICE
 6613  016576  005204                          INC     R4              ;INCREMENT UNIT NUMBER
 6614  016600  020437  002012                  CMP     R4,LSUNIT       ;MORE UNITS?
 6615  016604  001250                          BNE     1$              ;YES, GP HARD THE NEXT
 6616
 6617  016606  005037  003306                  CLR     SYSTAT          ;SYSTEM STATUS WORD
 6618  016612                                  RFLAGS  FLGLOC          ;GET USER FLAGS
    (3) 016612  104421                                                          TRAP    CSRFLA
    (3) 016614  010037  016752                                                  MOV     R0,FLGLO
 6619  016620  005037  003332          5$:     CLR     BLKER           ;NO ERROR
```

```
 6620  016624  013737  002210  003310          SETLEN: MOV     LENGTH,TAPLEN   ;GET # OF RECORDS
 6621  016632  006237  003310                          ASR     TAPLEN          ;GET # BLOCKS PER TRACK
 6622  016636  012737  000200  003334                  MOV     #200,SECREC     ;PRESET SECOND START AT 200
 6623  016644  022737  000200  003310                  CMP     #200,TAPLEN     ;# BLKS > 128.?
 6624  016652  101003                                  BHI     3$              ;NO-SWITCH TRACKS 2ND PASS
 6625  016654  012737  000400  003334                  MOV     #400,SECREC     ;YES-START AT 400
 6635
 6647
 6648  016662                                  3$:     ENDINIT
   (3) 016662
   (3) 016662  104411                                                  L10007:
                                                                               TRAP    CSINIT
 6649
 6650
 6651  016664  047524  020117  040515          TOMANY: .ASCIZ  /TOO MANY UNITS MAX.=8 /
 6652          016714                                  .EVEN
 6653  016714  042523  042514  052103          NODRVS: .ASCIZ  /SELECT AT LEAST 1 DRIVE /
 6654          016746                                  .EVEN
 6655  016746  000000                          STRT::  .WORD
 6656  016750  000000                          PDTFLG::.WORD           ;TU58 IS IN PDT
 6657  016752  000000                          FLGLOC::.WORD           ;USER FLAGS
```

```
                        ;++
                        ; THE AUTO DROP CODE IS INVOKED WHEN THE ADR FLAG IS SET AND CHECKS FOR
                        ; A VALID INTERFACE LOCATION.  DROPS UNIT IF INTERFACE IS NOT THERE.
                        ;--

                        BGNAUTO

                                NOP                     ;AUTO DROP ROUTINE                      LSAUTO::
                                SETVEC  #4,#TRPHND,#PRI07       ;GET BUS TRAP VEC.
                                                                                                MOV     #PRI07,-
                                                                                                MOV     #TRPHND,
                                                                                                MOV     #4,-(SP)
                                                                                                MOV     #3,-(SP)
                                                                                                TRAP    C$SVEC
                                                                                                ADD     #10,SP
                                MOV     #BLKTBL,TRPPTR  ;GET TOP OF DATA BLOCK TABLE
                        1$:     MOV     @TRPPTR,R5      ;GET DATA BLOCK
                                BIT     #BIT15!BIT11,@R5  ;NOT TESTED OR ABORTED?
                                BMI     2$              ;YES
                                TST     @RCSR(R5)       ;NO-VALID ADDRESS?
                                NOP                     ;YES...(TRAP IF NOT)
                        2$:     CMP     TRPPTR,#LSTDEV  ;MORE TO TRY?
                                BHIS    3$              ;NO
                                ADD     #2,TRPPTR       ;ON TO NEXT
                                BR      1$              ;GET IT
                        3$:     CLRVEC  #4              ;RESTORE
                                                                                                MOV     #4,R0
                                                                                                TRAP    C$CVEC
                        ENDAUTO                                                         L10010:
                                                                                                TRAP    CSAUTO
                        TRPPTR: .WORD


                        ;ILLEGAL ADDRESS TRAP HANDLER:

                        TRPHND: PRINTF  #MSAUTO         ;SAY "AUTO DROPPED"
                                                                                                MOV     #MSAUTO,
                                                                                                MOV     #1,-(SP)
                                                                                                MOV     SP,R0
                                                                                                TRAP    C$PNTF
                                                                                                ADD     #4,SP
                                MOV     @R5,R0          ;GET UNIT #
                                BIC     #177770,R0      ;MASK IT OFF
                                DODU    R0              ;DROP HIM
                                                                                                TRAP    C$DODU
                                RTI
                        MSAUTO: .ASCIZ  /%AAUTO DROP: %N/
```

SEQ 0090

```
                                        .SBTTL  CLEANUP CODING SECTION

                                        ;++
                                        ; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
                                        ; AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.
                                        ;--

017136                                          BGNCLN
017136
017136 005737 003340                            TST     ALLGON          ;ENTRANCE FROM ALL-UNITS-ABORTED?  L$CLEAN::
017142 001004                                   BNE     1$              ;YES-EXIT
017144 005737 002212                            TST     STAEOP          ;NO-STATS AT EOP?
017150 001401                                   BEQ     1$              ;NO
017152                                          DORPT                   ;YES
017152 104424                                                                   TRAP    C$DRPT

                                        1$:     ENDCLN
017154                                                                  L10011:
017154                                                                          TRAP    C$CLEAN
017154 104412
```

```
                        .SBTTL  DROP UNIT SECTION

                        ;++
                        ; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
                        ; TO NO LONGER BE TESTED.
                        ;--

                                BGNDU
                                                                        L$DU::
                                PUSH    R0              ;R0=UNIT NUMBER
                                                        ;SAVE IT
                                                        MOV     R0,-(SP)

                                PUSH    R5              ;SAVE PRESENT UNIT POINTER
                                                        MOV     R5,-(SP)

                                CALL    GETR5           ;GET POINTER TO UNIT
                                BIS     #BIT15,@R5      ;SET ABORTED
                                POP     R5              ;RESTORE PRESENT UNIT POINTER
                                                        MOV     (SP)+,R5

                                POP     R0              ;RETRIEVE UNIT NUMBER
                                                        MOV     (SP)+,R0

                                PRINTF  #ABOMSG,R0
                                                                        MOV     R0,-(SP)
                                                                        MOV     #ABOMSG,
                                                                        MOV     #2,-(SP)
                                                                        MOV     SP,R0
                                                                        TRAP    C$PNTF
                                                                        ADD     #6,SP

                                ENDDU
                                                                L10012:
                                                                        TRAP    C$DU

GETR5:  MOV     #BLKTBL,PTR     ;-->UNIT 0
1$:     MOV     @PTR,R5         ;GET STATUS WORD
        DEC     R0              ;CORRECT UNIT?
        BMI     2$              ;YES
        ADD     #2,PTR          ;NO,-->NEXT
        BR      1$              ;CONTINUE
2$:     RETURN
PTR:    .WORD   0

ABOMSG: .ASCIZ  /%ADROPPED UNIT %D1%N/
        .EVEN
```

```
                                        .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.
                                        ;--

017302                                          BGNAU
(3) 017302                                                                      L$AU::

                                        ;THE INIT CODE CONTAINS ALL CODE NECESSARY TO ADD A UNIT.




017302                                          ENDAU
(3) 017302
(3) 017302  104452                                                              L10013:
                                                                                        TRAP    C$AU
```

```
                        .SBTTL  TEST 1 / DEVICE SELF-DIAGNOSTIC EXECUTION

                                BGNMOD
                        .NLIST  ME,BEX

                                BGNTST

                                TSTID   #TST1                                   T1::
                                        MOV     #TST1,TSTTOP    ;SAVE ADDR OF TEST
                                        CALL    SETUP           ;INIT UNITS TSTPC
                                        CALL    SETDR           ;GET 1ST DRVS.
                                        CALL    RUN             ;DO TEST
                                        CALL    SWAPDR          ;GET NEXT DRVS.
                                        BCC     64$             ;BR NO 2ND DRVS
                                        CALL    SETUP           ;REINIT UNITS TSTPC
                                        CALL    RUN             ;REPEAT TEST
                                64$:                            ;DONE
                                EXIT TST
                                                                TRAP    C$EXIT
                                                                .WORD   L10014-.

                        TST1:   TUSELF

                                64$:    MOV     #TRBUF,R0       ;FORM COMMAND PACKET
                                        MOVB    #RSCMND,@R0     ;COMMAND FLAG
                                        MOVB    #RSMSIZ,1(R0)   ;SIZE OF MESSAGE
                                        MOVB    #RSSSLF,2(R0)   ;SELF TEST OPERATION
                                        CLRB    3(R0)           ;NO MODIFIER.
                                        CLR     4(R0)           ;NO DRIVE OR SWITCHES
                                        CLR     6(R0)           ;NO SEQUENCE NUMBER
                                        CLR     8.(R0)          ;NO BYTES
                                        CLR     10.(R0)         ;NO RECORD #
                                        MOV     #RSMSIZ,R1      ;GET SIZE
                                        TST     (R1)+           ;+2 FOR CHECKSUM
                                        MOV     #RSSNSZ,SNDCNT(R5)  ;SIZE TO SEND
                                        CALL    CHKSUM          ;FORM CHECKSUM
                                        MOV     R1,(R0)         ;INSERT INTO PACKET
                                        MOV     #RSEND,XSFLG(R5)  ;EXPECT END,
                                        MOV     #RSNDSZ,XSCNT(R5)   ;THIS BIG
                                        MOV     #1,XSPKNM(R5)   ;AND 1 PACKET
                                                                ;SEND
                                        CALL    RSVP            ;RETURN TO SCHEDULER
                                        BIT     #BIT3,@R5       ;RETRY?(BAD FLAG)
                                        BNE     64$             ;YES

                        6881    017476  005237  003322  INC     DONE
                        6882    017502  000207          RETURN
                        6883
                        6884
                        6885    017504                  ENDTST
                          (3)   017504
                          (3)   017504  104401                          L10014:
                                                                                TRAP    C$ETST
```

```
                        .SBTTL  TEST 2 / SEEK EOT,BOT
                                BGNTST
                                TSTID   #TST2                                   T2::
017506 012737 017552 003326                     MOV     #TST2,TSTTOP    ;SAVE ADDR OF TEST
017514 004737 006004                            CALL    SETUF           ;INIT UNITS TSTPC
017520 004737 005632                            CALL    SETDR           ;GET 1ST DRVS.
017524 004737 006052                            CALL    RUN             ;DO TEST
017530 004737 005530                            CALL    SWAPDR          ;GET NEXT DRVS.
017534 103004                                   BCC     64$             ;BR NO 2ND DRVS
017536 004737 006004                            CALL    SETUP           ;REINIT UNITS TSTPC
017542 004737 006052                            CALL    RUN             ;REPEAT TEST
017546                                  64$:                            ;DONE
                                EXIT TST
017546 104432                                                   TRAP    CSEXIT
017550 000206                                                   .WORD   L10015-.

017552 005004           TST2:   CLR     R4                      ;R4=INDEX INTO RECORD TABLE
017554 016465 017742 000064     1$:     MOV     RECDAT(R4),REC(R5)  ;GET THE RECORD
                                TUSEEK  REC(R5),DR(R5)  ;SEEK IT
017562                                  64$:    MOV     #TRBUF,R0       ;-->(POINT TO) XMIT BUFF
017562 012700 027722                            MOVB    #RSCMND,@R0     ;FORM COMMAND MESSAGE PA
017566 112710 000002                            MOVB    #RSMSIZ,1(R0)   ;THIS BIG
017572 112760 000012 000001                     MOVB    #RSSSEK,2(R0)   ;OP CODE IS SEEK
017600 016560 000064 000012                     MOV     REC(R5),10.(R0) ;TO THIS RECORD
017606 116560 000060 000004                     MOVB    DR(R5),4.(R0)   ;AND WHICH DRIVE
017614 105060 000003                            CLRB    3.(R0)          ;NO MODIFIER
017622 105060 000005                            CLRB    5.(R0)          ;NO SWITCHES
017626 005060 000006                            CLR     6.(R0)          ;NO SEQUENCE #
017632 005060 000010                            CLR     8.(R0)          ;NO BYTE COUNT
017636 012701 000012                            MOV     #RSMSIZ,R1      ;GET COUNT
017642 005721                                   TST     (R1)+           ;PLUS FLAG + BCNT
                                                                        ;FOR CHECKSUM CALC
017646 004737 013654                            CALL    CHKSUM          ;R0-->TOP   R1=# OF BYTE
017652 010110                                   MOV     R1,(R0)         ;INSERT INTO PACKET
                                                                        ;SET UP EXPECTATIONS:
017654 012765 000016 000070                     MOV     #RSSNSZ,SNDCNT(R5)  ;HOW MANY TO SEND
017662 112765 000002 000034                     MOVB    #RSCMND,XSFLG(R5)   ;EXPECT END PACK
017670 012765 000016 000036                     MOV     #RSNDSZ,XSCNT(R5)   ;COUNT WITH THIS
017676 012765 000001 000032                     MOV     #1.,XSPKNM(R5)  ;EXPECT ONLY 1 PACKET

017704 004737 006560                            CALL    RSVP            ;SEND
                                                                        ;AND RETURN TO SCHEDULER
017710 032715 000010                            BIT     #BIT3,@R5       ;RETRY (FLAG BYTE ERROR)
017714 001321                                   BNE     64$             ;YES

017720 062704 000002            ADD     #2,R4                   ;POINT TO NEXT RECORD
017724 026427 017742 177777     CMP     RECDAT(R4),#-1.         ;LAST ONE DONE?
017732 001310                   BNE     1$                      ;NO-LOOP
017734 005237 003322            INC     DONE                    ;YES-SET DONE FLAG
017740 000207                   RETURN
```

```
 6905
 6906 017742 000000            RECDAT: 0.        ;BOT
 6907 017744 000200                    200        ;BOT OTHER TRACK
 6908 017746 000177                    177        ;EOT
 6909 017750 000377                    377        ;EOT OTHER TRACK
 6910 017752 000400                    400        ;BOT AGAIN
 6911 017754 177777                    -1.
 6912 017756                           ENDTST
  (3) 017756                                                          L10015:
  (3) 017756 104401                                                           TRAP    C$ETST
```

```
                        .SBTTL  TEST 3 / HIGH ACTIVITY WRITE/READ (512 BYTE/BLOCK MODE)

                        ; WRITE THEN READ VARYING DATA FOR ALL PHYSICALLY ADJACENT BLOCKS AROUND
                        ; A RECORD, GO HALF-WAY INTO REMAINING TAPE REPEAT UNTIL EOT.

                                BGNTST

                                TSTID   #TST3                                           T3::
 (1) 017760 012737 020024 003326                        MOV     #TST3,TSTTOP    ;SAVE ADDR OF TEST
 (1) 017766 004737 006004                               CALL    SETUP           ;INIT UNITS TSTPC
 (1) 017772 004737 005632                               CALL    SETDR           ;GET 1ST DRVS.
 (1) 017776 004737 006052                               CALL    RUN             ;DO TEST
 (1) 020002 004737 005530                               CALL    SWAPDR          ;GET NEXT DRVS.
 (1) 020006 103004                                      BCC     64$             ;BR NO 2ND DRVS
 (1) 020010 004737 006004                               CALL    SETUP           ;REINIT UNITS TSTPC
 (1) 020014 004737 006052                               CALL    RUN             ;REPEAT TEST
 (1) 020020                                     64$:                            ;DONE
                                EXIT TST
 (3) 020020 104432                                                      TRAP    CSEXIT
 (3) 020022 001326                                                      .WORD   L10016-.

 020024 012765 000100 000066    TST3:   MOV     #100,TMP(R5)    ;INIT TO HALF OF REMAINING
 020032 005004                          CLR     R4              ;FOR INDEX INTO DATA TABLE
 020034 005065 000064                   CLR     REC(R5)         ;START AT RECORD 0
 020040 016465 022742 000072    1$:     MOV     TST3PT(R4),PATTEN(R5)   ;GET DATA
                                        TUWRIT  PATTEN(R5),REC(R5),#512.,DR(R5),#0
 020046 012700 027722                                   72$:    MOV     #TRBUF,R0       ;MAKE COMMAND PACKET:
 020052 112710 000002                                   MOVB    #RSCPKD,@R0     ;COMMAND FLAG
 020056 112760 000012 000001                            MOVB    #RSMSIZ,1(R0)   ;THIS SIZE
 020064 112760 000003 000002                            MOVB    #RSSWR,2(R0)    ;INSERT OP CODE-WRITE
 020072 112760 000000 000003                            MOVB    #0,3.(R0)       ;VERIFY (1 OR 0)
 020100 116560 000060 000004                            MOVB    DR(R5),4.(R0)   ;DRIVE #
 020106 112760 000020 000005                            MOVB    #020,5.(R0)     ;MAINTENANCE MODE SWITCH
 020114 005060 000006                                   CLR     6.(R0)          ;NO SEQUENCE #
 020120 012760 001000 000010                            MOV     #512.,8.(R0)    ;TOTAL COUNT TO WRITE
 020126 016560 000064 000012                            MOV     REC(R5),10.(R0) ;AT  RECORD N
 020134 012701 000012                                   MOV     #RSMSIZ,R1      ;THE PACKET SIZE PLUS+2
 020140 005721                                          TST     (R1)+           ;(FLAG AND COUNT) INTO R
 020142 012765 000016 000070                            MOV     #RSSNSZ,SNDCNT(R5)  ;LOAD THE  SIZE TO S
 020150 004737 013654                                   CALL    CHKSUM          ;R0 --> R1=COUNT
 020154 010110                                          MOV     R1,(R0)         ;PUT CHKSUM IN PACKET
                                                                                ;SET UP EXPECTATIONS:
 020156 012765 000020 000034                            MOV     #RSCONT,XSFLG(R5)   ;THE FLAG
 020164 012765 000001 000056                            MOV     #1,XSCNT(R5)    ;THE COUNT
 020172 012765 000001 000032                            MOV     #1,XSPKNM(R5)   ;THE # PACKETS EXPECTED
 020200 012702 001000                                   MOV     #512.,R2                ;GET # OF DATA B
 020204 004737 006360                                   CALL    RSVP            ;SEND (AND RETURN TO SCH
 020210 032715 000010                                   BIT     #BIT3,@R5       ;FLAG BYTE ERROR?
 020214 001314                                          BNE     72$             ;YES
 020216 042715 010000                                   BIC     #BIT12,@R5      ;FLAG FOR LAST PACKET
 020222 012700 027722                           64$:    MOV     #TRBUF,R0       ;POINT TO TOP OF BUFFER
 020226 020227 000200                                   CMP     R2,#128.        ;START DATA PACKET(S)
 020232 101004                                          BHI     65$             ;#512. > 128.!
 020234 010201                                          MOV     R2,R1           ;#512.<128.
 020236 052715 010000                                   BIS     #BIT12,@R5      ;SO LAST PACKET NOW
```

```
(1)  020242  000402                          BR      66$             ;USE REMAINING COUNT
(1)  020244  012701  000200            65$:  MOV     #128.,R1        ;USE 128. BYTES
(1)  020250  110160  000001            66$:  MOVB    R1,1(R0)        ;COPY COUNT TO BUFFER
(1)  020254  010103                          MOV     R1,R3           ;R3=COUNTER TO LOAD BUFF
(1)  020256  112710  000001                  MOVB    #R$DATA,@R0     ;FLAG FIRST
(1)  020262  005720                          TST     (R0)+           ;SKIP COUNT
(1)  020264  116520  000072            67$:  MOVB    PATTEN(R5),(R0)+        ;INSERT DATA
(1)  020270  005303                          DEC     R3              ;MORE?
(1)  020272  101374                          BHI     67$             ;YES
(1)  020274  012700  027722                  MOV     #TRBUF,R0       ;-->TOP AGAIN
(1)  020300  116001  000001                  MOVB    1(R0),R1        ;GET COUNT
(1)  020304  042701  177400                  BIC     #177400,R1      ;ZERO SIGN EXTEND
(1)  020310  010165  000070                  MOV     R1,SNDCNT(R5)   ;HOW MANY TO SEND PLUS
(1)  020314  062765  000004  000070          ADD     #4,SNDCNT(R5)   ;FLAG,COUNT,CHKSUM
(1)  020322  062701  000002                  ADD     #2,R1           ;COMPENSATE FOR FLAG + C
(1)  020326  004737  013654                  CALL    CHKSUM          ;FOR CHECKSUM CALC.
(1)  020332  110120                          MOVB    R1,(R0)+        ;CHKSUM INTO PACKET
(1)  020334  000301                          SWAB    R1              ;EVEN ON AN ODD
(1)  020336  110120                          MOVB    R1,(R0)+        ;BYTE BOUNDARY
(1)  020340  032715  010000                  BIT     #BIT12,@R5      ;LAST DATA PACKET?
(1)  020344  001412                          BEQ     68$             ;NO
(1)  020346  012765  000002  000034          MOV     #RSEND,XSFLG(R5)  ;YES-EXPECT 'END'
(1)  020354  012765  000016  000036          MOV     #RSNDSZ,XSCNT(R5)  ;OF THIS SIZE
(1)  020362  012765  000001  000032          MOV     #1,XSPKNM(R5)   ;AND 1 PACKET
(1)  020370  000411                          BR      69$             ;SEND
(1)  020372  012765  000020  000034    68$:  MOV     #RSCONT,XSFLG(R5)  ;(NOT LAST), EXPECT '
(1)  020400  012765  000001  000036          MOV     #1,XSCNT(R5)    ;AND 1 BYTE
(1)  020406  012765  000001  000032          MOV     #1,XSPKNM(R5)   ;AND 1 PACKET
(1)  020414  004737  006560            69$:  CALL    RSVP            ;SEND PACKET
(1)                                                                  ;AND RETURN TO SCHEDULER
(1)  020420  032715  000010                  BIT     #BIT3,@R5       ;FLAG BYTE RETRY?
(1)  020424  001210                          BNE     72$             ;YES
(1)  020426  032715  002000                  BIT     #BIT10,@R5      ;RETRY DATA ERROR?
(1)  020432  001004                          BNE     70$             ;YES
(1)  020434  162702  000200                  SUB     #128.,R2        ;NO, MORE DATA TO SEND?
(1)  020440  101270                          BHI     64$             ;YES
(1)  020442  000502                          BR      71$             ;NO
(1)  020444                            70$:  TURTRY  REC(R5),#512.,DR(R5)     ;RETRY HERE
(2)
(2)
(2)  020444  012700  027722            76$:  MOV     #TRBUF,R0       ;FORM CMND PACK:
(2)  020450  112710  000005                  MOVB    #RSCMND,@R0     ;MESSAGE PACK TYPE
(2)  020454  112760  000015  000001          MOVB    #RSMSIZ,1(R0)   ;THIS BIG
(2)  020462  112760  000003  000002          MOVB    #RSSRD,2(R0)    ;OP CODE-READ
(2)  020470  016540  000004  000013          MOV     REC(R5),10.(R0) ;THIS RECORD
(2)  020476  116540  000000  000004          MOVB    DR(R5),4.(R0)   ;THIS DRIVE
(2)  020504  105060  000003                  CLRB    3(R0)           ;PRESET NORM THRESHOLD
(2)  020510  105715                          TSTB    @R5             ;REDUCED?
(2)  020512  100002                          BPL     77$             ;NO
(2)  020514  105260  000003                  INCB    3(R0)           ;YES-CHANGE THRESHOLD
(2)  020520  012760  001000  000010    77$:  MOV     #512.,8.(R0)    ;# BYTES DESIRED
(2)  020526  112760  000020  000005          MOVB    #020,5.(R0)     ;MAINTENANCE MODE
(2)  020534  005060  000006                  CLR     6.(R0)          ;NO SEQUENCE #
(2)  020540  012701  000012                  MOV     #RSMSIZ,R1      ;SIZE OF PACKET
(2)  020544  005721                          TST     (R1)+           ;PLUS FLAG+COUNT INTO R1
(2)  020546  012765  000016  000070          MOV     #RSSNSZ,SNDCNT(R5)  ;SET UP SIZE TO SEND
```

```
     (2)
     (2)  020554  004737  013654                       CALL    CHKSUM            ;FORM CHECKSUM R1=COUNT
     (2)  020560  010110                               MOV     R1,(R0)           ;INSERT IN PACKET
     (2)
     (2)  020562  012701  001000                       MOV     #512.,R1                  ;SET EXPECTATION
     (2)                                                                         ;CALC # OF DATA PACKETS
     (2)  020566  012703  000034                       MOV     #XSFLG,R3         ;OFFSET OF FLAG
     (2)  020572  060503                               ADD     R5,R3             ;ABS. ADDR. OF XSFLG
     (2)  020574  005002                               CLR     R2                ;PRESET
     (2)  020576  005202                       73$:    INC     R2                ;# PACKETS EXPECTED
     (2)  020600  012723  000001                       MOV     #RSDATA,(R3)+     ;LOAD XSFLG
     (2)  020604  012723  000204                       MOV     #132.,(R3)+       ;AND EXPECT COUNT
     (2)  020610  162701  000200                       SUB     #128.,R1          ;NEG RESULT LAST TIME
     (2)  020614  101401                               BLOS    75$               ;LAST TIME!
     (2)  020616  000767                               BR      73$               ;MORE TO DO
     (2)  020620  005202                       75$:    INC     R2                ;ADD ONE FOR END PACK
     (2)  020622  010265  000032                       MOV     R2,XSPKNM(R5)     ;SAVE # PACKETS TO EXPEC
     (2)  020626  012723  000002                       MOV     #RSEND,(R3)+      ;EXPECT AN END
     (2)  020632  012713  000016                       MOV     #RSNDSZ,(R3)      ;THIS BIG-14. BYTES
     (2)
     (2)  020636  004737  006560                       CALL    RSVP              ;SEND
     (2)                                                                         ;AND RETURN TO SCHEDULER
     (2)
  6930    020652                               TUREAD  REC(R5),#512.,DR(R5),#0
     (1)
     (1)
     (1)  020652  012700  027722               82$:    MOV     #TRBUF,R0         ;FORM CMND PACK:
     (1)  020656  112710  000002                       MOVB    #RSCMND,@R0       ;MESSAGE PACK TYPE
     (1)  020662  112760  000012  000001               MOVB    #RSMSIZ,1(R0)     ;THIS BIG
     (1)  020670  112760  000002  000002               MOVB    #RSSRD,2(R0)      ;OP CODE IS READ
     (1)  020676  016560  000064  000012               MOV     REC(R5),10.(R0)   ;THIS RECORD
     (1)  020704  116560  000060  000004               MOVB    DR(R5),4.(R0)     ;THIS DRIVE
     (1)  020712  112760  000000  000003               MOVB    #0,3.(R0)         ;VERIFY
     (1)  020720  012760  001000  000010               MOV     #512.,8.(R0)      ;TOTAL BYTES TO READ
     (1)  020726  112760  000020  000005               MOVB    #020,5.(R0)       ;MAINTENANCE MODE
     (1)  020734  005060  000006                       CLR     6.(R0)            ;NO SEQUENCE #
     (1)  020740  012701  000012                       MOV     #RSMSIZ,R1        ;GET SIZE OF PACKET
     (1)  020744  005721                               TST     (R1)+             ;+2 FOR CHECKSUM
     (1)  020746  012765  000016  000070               MOV     #RSSNSZ,SNDCNT(R5)   ;SIZE TO SEND
     (1)  020754  004737  013654                       CALL    CHKSUM            ;FORM CHECKSUM R1=COUNT
     (1)  020760  010110                               MOV     R1,(R0)           ;INSERT CHECKSUM
     (1)
     (1)  020762  012701  001000                       MOV     #512.,R1                  ;SET EXPECTATION
     (1)                                                                         ;CALC # OF DATA PACKETS
     (1)  020766  012703  000034                       MOV     #XSFLG,R3         ;GET OFFSET
     (1)  020772  060503                               ADD     R5,R3             ;ABS. ADDR. OF XSFLG
     (1)  020774  005002                               CLR     R2                ;PRESET AS NONE
     (1)  020776  005202                       78$:    INC     R2                ;# PACKETS EXPECTED
     (1)  021000  012723  000001                       MOV     #RSDATA,(R3)+     ;LOAD XSFLG
     (1)  021004  012723  000204                       MOV     #132.,(R3)+       ;AND EXPECTED COUNT
     (1)  021010  162701  000200                       SUB     #128.,R1          ;NEG RESULT LAST TIME
     (1)  021014  101401                               BLOS    80$               ;LAST TIME
     (1)  021016  000767                               BR      78$               ;MORE TO DO
     (1)  021020  005202                       80$:    INC     R2                ;ADD ONE FOR END PACK
     (1)  021022  010265  000032                       MOV     R2,XSPKNM(R5)     ;SAVE # PACKETS TO EXPEC
     (1)  021026  012723  000002                       MOV     #RSEND,(R3)+      ;EXPECT AN END ALSO...
```

```
(1)  021032  012713  000016                     MOV     #RSNDSZ,(R3)       ;THIS BIG-14. BYTES
(1)  021036  004737  006560                     CALL    RSVP               ;SEND
(1)                                                                        ;AND RETURN TO SCHEDULER
(1)  021042  032715  002010             81$:    BIT     #BIT10!BIT3,@R5    ;RETRY?
(1)  021046  001500                             BEQ     79$                ;NO.
(1)  021050                                     TURTRY  REC(R5),#512.,DR(R5)      ;YES
(2)
(2)
(2)  021050  012700  027722             86$:    MOV     #TRBUF,R0          ;FORM CMND PACK:
(2)  021054  112710  000002                     MOVB    #RSCMND,@R0        ;MESSAGE PACK TYPE
(2)  021060  112760  000012  000001             MOVB    #RSMSIZ,1(R0)      ;THIS BIG
(2)  021066  112760  000002  000002             MOVB    #RSSRD,2(R0)       ;OP CODE-READ
(2)  021074  016560  000064  000012             MOV     REC(R5),10.(R0)    ;THIS RECORD
(2)  021102  116560  000060  000004             MOVB    DR(R5),4.(R0)      ;THIS DRIVE
(2)  021110  105060  000003                     CLRB    3(R0)              ;PRESET NORM THRESHOLD
(2)  021114  105715                             TSTB    @R5                ;REDUCED?
(2)  021116  100002                             BPL     87$                ;NO
(2)  021120  105260  000003                     INCB    3(R0)              ;YES-CHANGE THRESHOLD
(2)  021124  012760  001000  000010     87$:    MOV     #512.,8.(R0)       ;# BYTES DESIRED
(2)  021132  112760  000020  000005             MOVB    #020,5.(R0)        ;MAINTENANCE MODE
(2)  021140  005060  000006                     CLR     6.(R0)             ;NO SEQUENCE #
(2)  021144  012701  000012                     MOV     #RSMSIZ,R1         ;SIZE OF PACKET
(2)  021150  005721                             TST     (R1)+              ;PLUS FLAG+COUNT INTO R1
(2)  021152  012765  000016  000070             MOV     #RSSNSZ,SNDCNT(R5)   ;SET UP SIZE TO SEND
(2)
(2)  021160  004737  013654                     CALL    CHKSUM             ;FORM CHECKSUM R1=COUNT
(2)  021164  010110                             MOV     R1,(R0)            ;INSERT IN PACKET
(2)
(2)  021166  012701  001000                     MOV     #512.,R1                 ;SET EXPECTATION
(2)                                                                        ;CALC # OF DATA PACKETS
(2)  021172  012703  000034                     MOV     #XSFLG,R3          ;OFFSET OF FLAG
(2)  021176  060503                             ADD     R5,R3              ;ABS. ADDR. OF XSFLG
(2)  021200  005002                             CLR     R2                 ;PRESET
(2)  021202  005202                     83$:    INC     R2                 ;# PACKETS EXPECTED
(2)  021204  012723  000001                     MOV     #RSDATA,(R3)+      ;LOAD XSFLG
(2)  021210  012723  000204                     MOV     #132.,(R3)+        ;AND EXPECT COUNT
(2)  021214  162701  000200                     SUB     #128.,R1           ;NEG RESULT LAST TIME
(2)  021220  101401                             BLOS    85$                ;LAST TIME!
(2)  021222  000767                             BR      83$                ;MORE TO DO
(2)  021224  005202                     85$:    INC     R2                 ;ADD ONE FOR END PACK
(2)  021226  010265  000032                     MOV     R2,XSPKNM(R5)      ;SAVE # PACKETS TO EXPEC
(2)  021232  012723  000002                     MOV     #RSEND,(R3)+       ;EXPECT AN END
(2)  021236  012713  000016                     MOV     #RSNDSZ,(R3)       ;THIS BIG-14. BYTES
(2)
(2)  021242  004737  006560                     CALL    RSVP               ;SEND
(2)                                                                        ;AND RETURN TO SCHEDULER
(2)
     021252  062704  000002                 ADD     #2,R4              ;POINT TO NEXT DATA
     021256  005764  022742                 TST     TST3PT(R4)         ;END?
     021262  001402                         BEQ     2$                 ;YES
     021264  000137  020040                 JMP     1$                 ;NO-WRITE, READ NEW DATA
     021270  005004                 2$:     CLR     R4                 ;POINT TO FIRST DATA
     021272  062765  000200  000064         ADD     #200,REC(R5)       ;BUT NOW USE ADJACENT RECORD
     021300  032765  001000  000064         BIT     #1000,REC(R5)      ;ALL ADJACENT RECORDS DONE?
     021306  001002                         BNE     3$                 ;YES
     021310  000137  020040                 JMP     1$                 ;NO-WRITE, READ AT NEW RECORD
```

```
 6940  021314  162765  001000  000064          3$:     SUB     #1000,REC(R5)   ;RESTORE TO NEXT RECORD
 6941  021322  066565  000066  000064                  ADD     TMP(R5),REC(R5) ;HALF INTO REST OF TAPE
 6942  021330  006265  000066                          ASR     TMP(R5)         ;HALF OF HALF FOR NEXT TIME
 6943  021334  103402                                  BCS     4$              ;DONE?
 6944  021336  000137  020040                          JMP     1$              ;NO
 6945  021342  005237  003322                  4$:     INC     DONE            ;YES-SET FLAG
 6946  021346  000207                                  RETURN
 6947  021350                                          ENDTST
   (3) 021350
   (3) 021350  104401                                                  L10016:
                                                                               TRAP    CSETST
 6948
```

```
                                        .SBTTL  TEST 4 / HIGH ACTIVITY WRITE/READ (128 BYTE/BLOCK MODE)

                                        ; WRITE THEN READ VARYING DATA FOR ALL PHYSICALLY ADJACENT BLOCKS AROUND
                                        ; A RECORD, GO HALF-WAY INTO REMAINING TAPE REPEAT UNTIL EOT.

     021352                                     BGNTST
(3)  021352                                                                             T4::
(3)  021352
     021352                                     TSTID   #TST4
(1)  021352  012737  021416  003326                             MOV     #TST4,TSTTOP    ;SAVE ADDR OF TEST
(1)  021360  004737  006004                                     CALL    SETUP           ;INIT UNITS TSTPC
(1)  021364  004737  005632                                     CALL    SETDR           ;GET 1ST DRVS.
(1)  021370  004737  006052                                     CALL    RUN             ;DO TEST
(1)  021374  004737  005530                                     CALL    SWAPDR          ;GET NEXT DRVS.
(1)  021400  103004                                             BCC     64$             ;BR NO 2ND DRVS
(1)  021402  004737  006004                                     CALL    SETUP           ;REINIT UNITS TSTPC
(1)  021406  004737  006052                                     CALL    RUN             ;REPEAT TEST
(1)  021412                                             64$:                            ;DONE
     021412                                     EXIT TST
(3)  021412  104432                                                                             TRAP    C$EXIT
(3)  021414  001340                                                                             .WORD   L10017-.


     021416  012765  000400  000066     TST4:   MOV     #400,TMP(R5)            ;INIT TO HALF OF REMAINING
     021424  005004                             CLR     R4                      ;FOR INDEX INTO DATA TABLE
     021426  005065  000064                     CLR     REC(R5)                 ;START AT RECORD 0
     021432  016465  022742  000072     1$:     MOV     TST3PT(R4),PATTEN(R5)   ;GET DATA
     021440                                     TUWRIT  PATTEN(R5),REC(R5),#128.,DR(R5),#BIT7
(1)  021440  012700  027722                             72$:    MOV     #TRBUF,R0               ;MAKE COMMAND PACKET:
(1)  021444  112710  000002                                     MOVB    #RSCMND,@R0             ;COMMAND FLAG
(1)  021450  112760  000012  000001                             MOVB    #RSMSIZ,1(R0)           ;THIS SIZE
(1)  021456  112760  000005  000002                             MOVB    #RSSWR,2(R0)            ;INSERT OP CODE-WRITE
(1)  021464  112760  000200  000003                             MOVB    #BIT7,3.(R0)            ;VERIFY (1 OR 0)
(1)  021472  116560  000060  000004                             MOVB    DR(R5),4.(R0)           ;DRIVE #
(1)  021500  112760  000020  000005                             MOVB    #020,5.(R0)             ;MAINTENANCE MODE SWITCH
(1)  021506  005060  000006                                     CLR     6.(R0)                  ;NO SEQUENCE #
(1)  021512  012760  000200  000010                             MOV     #128.,8.(R0)            ;TOTAL COUNT TO WRITE
(1)  021520  016560  000064  000012                             MOV     REC(R5),10.(R0)         ;AT  RECORD N
(1)  021526  012701  000012                                     MOV     #RSMSIZ,R1              ;THE PACKET SIZE PLUS+2
(1)  021532  005721                                             TST     (R1)+                   ;(FLAG AND COUNT) INTO R
(1)  021534  012765  000016  000070                             MOV     #RSSNSZ,SNDCNT(R5)      ;LOAD THE  SIZE TO S
(1)  021542  004737  013654                                     CALL    CHKSUM                  ;R0 --> R1=COUNT
(1)  021546  010110                                             MOV     R1,(R0)                 ;PUT CHKSUM IN PACKET
(1)                                                                                             ;SET UP EXPECTATIONS:
(1)  021550  012765  000020  000034                             MOV     #RSCONT,XSFLG(R5)       ;THE FLAG
(1)  021556  012765  000001  000036                             MOV     #1,XSCNT(R5)            ;THE COUNT
(1)  021564  012765  000001  000032                             MOV     #1,XSPKNM(R5)           ;THE # PACKETS EXPECTED
(1)  021572  012702  000200                                     MOV     #128.,R2                        ;GET # OF DATA B
(1)  021576  004737  006560                                     CALL    RSVP                    ;SEND (AND RETURN TO SCH
(1)  021602  032715  000010                                     BIT     #BIT3,@R5               ;FLAG BYTE ERROR?
(1)  021606  001314                                             BNE     72$                     ;YES
(1)  021610  042715  010000                                     BIC     #BIT12,@R5              ;FLAG FOR LAST PACKET
(1)  021614  012700  027722                             64$:    MOV     #TRBUF,R0               ;POINT TO TOP OF BUFFER
(1)  021620  020227  000200                                     CMP     R2,#128.                ;START DATA PACKET(S)
(1)  021624  101004                                             BHI     65$                     ;#128. > 128.!
(1)  021626  010201                                             MOV     R2,R1                   ;#128.<128.
(1)  021630  052715  010000                                     BIS     #BIT12,@R5              ;SO LAST PACKET NOW
```

```
(1)  021634  000402
(1)  021636  012701  000200
(1)  021642  110160  000001
(1)  021646  010103
(1)  021650  112710  000001
(1)  021654  005720
(1)  021656  116520  000072
(1)  021662  005303
(1)  021664  101374
(1)  021666  012700  027722
(1)  021672  116001  000001
(1)  021676  042701  177400
(1)  021702  010165  000070
(1)  021706  062765  000004  000070
(1)  021714  062701  000002
(1)  021720  004737  013654
(1)  021724  110120
(1)  021726  000301
(1)  021730  110120
(1)  021732  032715  010000
(1)  021736  001412
(1)  021740  012765  000002  000034
(1)  021746  012765  000016  000036
(1)  021754  012765  000001  000032
(1)  021762  000411
(1)  021764  012765  000020  000034
(1)  021772  012765  000001  000036
(1)  022000  012765  000001  000032
(1)  022006  004737  006560
(1)
(1)  022012  032715  000010
(1)  022016  001210
(1)  022020  032715  002000
(1)  022024  001004
(1)  022026  162702  000200
(1)  022032  101270
(1)  022034  000502
(1)  022036
(2)
(2)
(2)  022036  012700  027722
(2)  022042  112710  000005
(2)  022046  112760  000015  000001
(2)  022054  112760  000002  000002
(2)  022062  016560  000064  000015
(2)  022070  116560  000060  000004
(2)  022076  105060  000003
(2)  022102  105715
(2)  022104  100002
(2)  022106  105260  000003
(2)  022112  012760  000200  000010
(2)  022120  112760  000020  000005
(2)  022126  005060  000006
(2)  022132  012701  000012
(2)  022136  005721
(2)  022140  012765  000016  000070
```

```
                BR      66$             ;USE REMAINING COUNT
65$:    MOV     #128.,R1                ;USE 128. BYTES
66$:    MOVB    R1,1(R0)                ;COPY COUNT TO BUFFER
        MOV     R1,R3                   ;R3=COUNTER TO LOAD BUFF
        MOVB    #RSDATA,@R0             ;FLAG FIRST
        TST     (R0)+                   ;SKIP COUNT
67$:    MOVB    PATTEN(R5),(R0)+                ;INSERT DATA
        DEC     R3                      ;MORE?
        BHI     67$                     ;YES
        MOV     #TRBUF,R0               ;-->TOP AGAIN
        MOVB    1(R0),R1                ;GET COUNT
        BIC     #177400,R1              ;ZERO SIGN EXTEND
        MOV     R1,SNDCNT(R5)           ;HOW MANY TO SEND PLUS
        ADD     #4,SNDCNT(R5)           ;FLAG,COUNT,CHKSUM
        ADD     #2,R1                   ;COMPENSATE FOR FLAG + C
        CALL    CHKSUM                  ;FOR CHECKSUM CALC.
        MOVB    R1,(R0)+                ;CHKSUM INTO PACKET
        SWAB    R1                      ;EVEN ON AN ODD
        MOVB    R1,(R0)+                ;BYTE BOUNDARY
        BIT     #BIT12,@R5              ;LAST DATA PACKET?
        BEQ     68$                     ;NO
        MOV     #RSEND,XSFLG(R5)  ;YES-EXPECT 'END'
        MOV     #RSNDSZ,XSCNT(R5)  ;OF THIS SIZE
        MOV     #1,XSPKNM(R5)           ;AND 1 PACKET
        BR      69$                     ;SEND
68$:    MOV     #RSCONT,XSFLG(R5)  ;(NOT LAST), EXPECT '
        MOV     #1,XSCNT(R5)            ;AND 1 BYTE
        MOV     #1,XSPKNM(R5)           ;AND 1 PACKET
69$:    CALL    RSVP                    ;SEND PACKET
                                        ;AND RETURN TO SCHEDULER
        BIT     #BIT3,@R5               ;FLAG BYTE RETRY?
        BNE     72$                     ;YES
        BIT     #BIT10,@R5              ;RETRY DATA ERROR?
        BNE     70$                     ;YES
        SUB     #128.,R2                ;NO, MORE DATA TO SEND?
        BHI     64$                     ;YES
        BR      71$                     ;NO
70$:    TURTRY  REC(R5),#128.,DR(R5)    ;RETRY HERE

76$:    MOV     #TRBUF,R0               ;FORM CMND PACK:
        MOVB    #RSCMND,@R0             ;MESSAGE PACK TYPE
        MOVB    #RSMSIZ,1(R0)           ;THIS BIG
        MOVB    #RSSRD,2(R0)            ;OP CODE-READ
        MOV     REC(R5),10.(R0)         ;THIS RECORD
        MOVB    DR(R5),4.(R0)           ;THIS DRIVE
        CLRB    3(R0)                   ;PRESET NORM THRESHOLD
        TSTB    @R5                     ;REDUCED?
        BPL     77$                     ;NO
        INCB    3(R0)                   ;YES-CHANGE THRESHOLD
77$:    MOV     #128.,8.(R0)            ;# BYTES DESIRED
        MOVB    #020,5.(R0)             ;MAINTENANCE MODE
        CLR     6.(R0)                  ;NO SEQUENCE #
        MOV     #RSMSIZ,R1              ;SIZE OF PACKET
        TST     (R1)+                   ;PLUS FLAG+COUNT INTO R1
        MOV     #RSSNSZ,SNDCNT(R5)  ;SET UP SIZE TO SEND
```

```
    (2)
    (2)  022146  004737  013654                    CALL    CHKSUM          ;FORM CHECKSUM R1=COUNT
    (2)  022152  010110                            MOV     R1,(R0)         ;INSERT IN PACKET
    (2)
    (2)  022154  012701  000200                    MOV     #128.,R1                ;SET EXPECTATION
    (2)  022160  012703  000034                                            ;CALC # OF DATA PACKETS
    (2)  022164  060503                            MOV     #XSFLG,R3       ;OFFSET OF FLAG
    (2)  022166  005002                            ADD     R5,R3           ;ABS. ADDR. OF XSFLG
    (2)  022170  005202                            CLR     R2              ;PRESET
    (2)  022172  012723  000001            73$:    INC     R2              ;# PACKETS EXPECTED
    (2)  022176  012723  000204                    MOV     #RSDATA,(R3)+   ;LOAD XSFLG
    (2)  022202  162701  000200                    MOV     #132.,(R3)+     ;AND EXPECT COUNT
    (2)  022206  101401                            SUB     #128.,R1        ;NEG RESULT LAST TIME
    (2)  022210  000767                            BLOS    75$             ;LAST TIME!
    (2)  022212  005202                            BR      73$             ;MORE TO DO
    (2)  022214  010265  000032            75$:    INC     R2              ;ADD ONE FOR END PACK
    (2)  022220  012723  000002                    MOV     R2,XSPKNM(R5)   ;SAVE # PACKETS TO EXPEC
    (2)  022224  012713  000016                    MOV     #RSEND,(R3)+    ;EXPECT AN END
    (2)                                            MOV     #RSNDSZ,(R3)    ;THIS BIG-14. BYTES
    (2)
    (2)  022230  004737  006560                    CALL    RSVP            ;SEND
    (2)                                                                    ;AND RETURN TO SCHEDULER
    (2)
 6965  022244                        TUREAD  REC(R5),#128.,DR(R5),#BIT7
    (1)
    (1)
    (1)  022244  012700  027722            82$:    MOV     #TRBUF,R0       ;FORM CMND PACK:
    (1)  022250  112710  000002                    MOVB    #RSCMND,@R0     ;MESSAGE PACK TYPE
    (1)  022254  112760  000012  000001            MOVB    #RSMSIZ,1(R0)   ;THIS BIG
    (1)  022262  112760  000002  000002            MOVB    #RSSRD,2(R0)    ;OP CODE IS READ
    (1)  022270  016560  000064  000012            MOV     REC(R5),10.(R0) ;THIS RECORD
    (1)  022276  116560  000060  000004            MOVB    DR(R5),4.(R0)   ;THIS DRIVE
    (1)  022304  112760  000200  000003            MOVB    #BIT7,3.(R0)    ;VERIFY
    (1)  022312  012760  000200  000010            MOV     #128.,8.(R0)    ;TOTAL BYTES TO READ
    (1)  022320  112760  000020  000005            MOVB    #020,5.(R0)     ;MAINTENANCE MODE
    (1)  022326  005060  000006                    CLR     6.(R0)          ;NO SEQUENCE #
    (1)  022332  012701  000012                    MOV     #RSMSIZ,R1      ;GET SIZE OF PACKET
    (1)  022336  005721                            TST     (R1)+           ;+2 FOR CHECKSUM
    (1)  022340  012765  000016  000070            MOV     #RSSNSZ,SNDCNT(R5)   ;SIZE TO SEND
    (1)  022346  004737  013654                    CALL    CHKSUM          ;FORM CHECKSUM R1=COUNT
    (1)  022352  010110                            MOV     R1,(R0)         ;INSERT CHECKSUM
    (1)
    (1)  022354  012701  000200                    MOV     #128.,R1                ;SET EXPECTATION
    (1)                                                                    ;CALC # OF DATA PACKETS
    (1)  022360  012703  000034                    MOV     #XSFLG,R3       ;GET OFFSET
    (1)  022364  060503                            ADD     R5,R3           ;ABS. ADDR. OF XSFLG
    (1)  022366  005002                            CLR     R2              ;PRESET AS NONE
    (1)  022370  005203                    78$:    INC     R2              ;# PACKETS EXPECTED
    (1)  022372  012723  000001                    MOV     #RSDATA,(R3)+   ;LOAD XSFLG
    (1)  022376  012723  000204                    MOV     #132.,(R3)+     ;AND EXPECTED COUNT
    (1)  022402  162701  000200                    SUB     #128.,R1        ;NEG RESULT LAST TIME
    (1)  022406  101401                            BLOS    80$             ;LAST TIME
    (1)  022410  000767                            BR      78$             ;MORE TO DO
    (1)  022412  005202                    80$:    INC     R2              ;ADD ONE FOR END PACK
    (1)  022414  010265  000032                    MOV     R2,XSPKNM(R5)   ;SAVE # PACKETS TO EXPEC
    (1)  022420  012725  000002                    MOV     #RSEND,(R3)+    ;EXPECT AN END ALSO...
```

```
(1)  022424  012713  000016                          MOV     #RSNDSZ,(R3)       ;THIS BIG-14. BYTES
(1)  022430  004737  006560                          CALL    RSVP               ;SEND
(1)                                                                             ;AND RETURN TO SCHEDULER
(1)  022434  032715  002010                   81$:   BIT     #BIT10!BIT3,@R5    ;RETRY?
(1)  022440  001500                                  BEQ     79$                ;NO.
(1)  022442                                          TURTRY  REC(R5),#128.,DR(R5)      ;YES
(2)
(2)
(2)  022442  012700  027722                   86$:   MOV     #TRBUF,R0          ;FORM CMND PACK:
(2)  022446  112710  000002                          MOVB    #RSCMND,@R0        ;MESSAGE PACK TYPE
(2)  022452  112760  000012  000001                  MOVB    #RSMSIZ,1(R0)      ;THIS BIG
(2)  022460  112760  000002  000002                  MOVB    #RSSRD,2(R0)       ;OP CODE-READ
(2)  022466  016560  000064  000012                  MOV     REC(R5),10.(R0)    ;THIS RECORD
(2)  022474  116560  000060  000004                  MOVB    DR(R5),4.(R0)      ;THIS DRIVE
(2)  022502  105060  000003                          CLRB    3(R0)              ;PRESET NORM THRESHOLD
(2)  022506  105715                                  TSTB    @R5                ;REDUCED?
(2)  022510  100002                                  BPL     87$                ;NO
(2)  022512  105260  000003                          INCB    3(R0)              ;YES-CHANGE THRESHOLD
(2)  022516  012760  000200  000010           87$:   MOV     #128.,8.(R0)       ;# BYTES DESIRED
(2)  022524  112760  000020  000005                  MOVB    #020,5.(R0)        ;MAINTENANCE MODE
(2)  022532  005060  000006                          CLR     6.(R0)             ;NO SEQUENCE #
(2)  022536  012701  000012                          MOV     #RSMSIZ,R1         ;SIZE OF PACKET
(2)  022542  005721                                  TST     (R1)+              ;PLUS FLAG+COUNT INTO R1
(2)  022544  012765  000016  000070                  MOV     #RSSNSZ,SNDCNT(R5)   ;SET UP SIZE TO SEND
(2)
(2)  022552  004737  013654                          CALL    CHKSUM             ;FORM CHECKSUM R1=COUNT
(2)  022556  010110                                  MOV     R1,(R0)            ;INSERT IN PACKET
(2)
(2)  022560  012701  000200                          MOV     #128.,R1                ;SET EXPECTATION
(2)                                                                             ;CALC # OF DATA PACKETS
(2)  022564  012703  000034                          MOV     #XSFLG,R3          ;OFFSET OF FLAG
(2)  022570  060503                                  ADD     R5,R3              ;ABS. ADDR. OF XSFLG
(2)  022572  005002                                  CLR     R2                 ;PRESET
(2)  022574  005202                           83$:   INC     R2                 ;# PACKETS EXPECTED
(2)  022576  012723  000001                          MOV     #RSDATA,(R3)+      ;LOAD XSFLG
(2)  022602  012723  000204                          MOV     #132.,(R3)+        ;AND EXPECT COUNT
(2)  022606  162701  000200                          SUB     #128.,R1           ;NEG RESULT LAST TIME
(2)  022612  101401                                  BLOS    85$                ;LAST TIME!
(2)  022614  000767                                  BR      83$                ;MORE TO DO
(2)  022616  005202                           85$:   INC     R2                 ;ADD ONE FOR END PACK
(2)  022620  010265  000032                          MOV     R2,XSPKNM(R5)      ;SAVE # PACKETS TO EXPEC
(2)  022624  012723  000002                          MOV     #RSEND,(R3)+       ;EXPECT AN END
(2)  022630  012713  000016                          MOV     #RSNDSZ,(R3)       ;THIS BIG-14. BYTES
(2)
(2)  022634  004737  006560                          CALL    RSVP               ;SEND
(2)                                                                             ;AND RETURN TO SCHEDULER
(2)
     022644  062704  000002                   ADD     #2,R4              ;POINT TO NEXT DATA
     022650  005764  022742                   TST     TST3PT(R4)         ;END?
     022654  001402                           BEQ     2$                 ;YES
     022656  000137  021432                   JMP     1$                 ;NO-WRITE, READ NEW DATA
     022662  005004                   2$:     CLR     R4                 ;POINT TO FIRST DATA
     022664  062765  001000  000064           ADD     #1000,REC(R5)      ;BUT NOW USE ADJACENT RECORD
     022672  032765  004000  000064           BIT     #4000,REC(R5)      ;ALL ADJACENT RECORDS DONE?
     022700  001002                           BNE     3$                 ;YES
     022702  000137  021432                   JMP     1$                 ;NO-WRITE, READ AT NEW RECORD
```

```
 6975  022706  162765  004000  000064          3$:     SUB     #4000,REC(R5)   ;RESTORE TO NEXT RECORD
 6976  022714  066565  000066  000064                  ADD     TMP(R5),REC(R5) ;HALF INTO REST OF TAPE
 6977  022722  006265  000066                          ASR     TMP(R5)         ;HALF OF HALF FOR NEXT TIME
 6978  022726  103402                                  BCS     4$              ;DONE?
 6979  022730  000137  021432                          JMP     1$              ;NO
 6980  022734  005237  003322                  4$:     INC     DONE            ;YES-SET FLAG
 6981  022740  000207                                  RETURN
 6982  022742  000000                          TST3PT: .WORD   000000
 6983  022744  125252                                  .WORD   125252
 6984  022746  177777                                  .WORD   177777
 6985  022750  052525                                  .WORD   052525
 6986  022752  000000                                  .WORD   000000
 6987
 6988
 6989  022754                                          ENDTST
   (3) 022754
   (3) 022754  104401                                                  L10017:
                                                                               TRAP    CSETST
```

```
 6992
 6993
 6994
 6995  022756                                .SBTTL  TEST 5 / WRITE SELECTED NUMBER OF BLOCKS
   (3) 022756                                BGNTST
 6996  022756                                                                        T5::
   (1) 022756  012737  023022  003326        TSTID   #TST5
   (1) 022764  004737  006004                                    MOV     #TST5,TSTTOP    ;SAVE ADDR OF TEST
   (1) 022770  004737  005632                                    CALL    SETUP           ;INIT UNITS TSTPC
   (1) 022774  004737  006052                                    CALL    SETDR           ;GET 1ST DRVS.
   (1) 023000  004737  005530                                    CALL    RUN             ;DO TEST
   (1) 023004  103004                                            CALL    SWAPDR          ;GET NEXT DRVS.
   (1) 023006  004737  006004                                    BCC     64$             ;BR NO 2ND DRVS
   (1) 023012  004737  006052                                    CALL    SETUP           ;REINIT UNITS TSTPC
   (1) 023016                                                    CALL    RUN             ;REPEAT TEST
 6997  023016                                                64$:                        ;DONE
   (3) 023016  104432                        EXIT TST
   (3) 023020  000724                                                            TRAP    C$EXIT
 6998                                                                            .WORD   L10020-.
 6999
 7000  023022  005065  000064
 7001  023026  013765  003310  000066  TST5:   CLR     REC(R5)         ;START AT REC 0
 7002  023034  005065  000062                  MOV     TAPLEN,TMP(R5)  ;GET THE # OF BLOCKS PER TRACK
 7003  023040  016565  000064  000072          CLR     TRK(R5)         ;TRK(R5)=1ST OR 2ND PASS COUNTER
 7004  023046  005737  002220        1$:       MOV     REC(R5),PATTEN(R5)  ;USE RECORD NO. FOR DATA
 7005  023052  001403                          TST     DRVCHK                  ;ADD DR #?
 7006  023054  066565  000060  000072          BEQ     10$                     ;NO
 7007  023062                                  ADD     DR(R5),PATTEN(R5)       ;YES, ADD DRIVE ID
   (1) 023062  012700  027722        10$:      TUWRIT  PATTEN(R5),REC(R5),#512.,DR(R5),#0
   (1) 023066  112710  000002                                  72$:     MOV     #TRBUF,R0       ;MAKE COMMAND PACKET:
   (1) 023072  112760  000012  000001                          MOVB    #RSCMND,@R0     ;COMMAND FLAG
   (1) 023100  112760  000003  000002                          MOVB    #RSMSIZ,1(R0)   ;THIS SIZE
   (1) 023106  112760  000000  000003                          MOVB    #RSSWR,2(R0)    ;INSERT OP CODE-WRITE
   (1) 023114  116560  000060  000004                          MOVB    #0,3.(R0)       ;VERIFY (1 OR 0)
   (1) 023122  112760  000020  000005                          MOVB    DR(R5),4.(R0)   ;DRIVE #
   (1) 023130  005060  000006                                  MOVB    #020,5.(R0)     ;MAINTENANCE MODE SWITCH
   (1) 023134  012760  001000  000010                          CLR     6.(R0)          ;NO SEQUENCE #
   (1) 023142  016560  000064  000012                          MOV     #512.,8.(R0)    ;TOTAL COUNT TO WRITE
   (1) 023150  012701  000012                                  MOV     REC(R5),10.(R0) ;AT  RECORD N
   (1) 023154  005721                                          MOV     #RSMSIZ,R1      ;THE PACKET SIZE PLUS+2
   (1) 023156  012765  000016  000070                          TST     (R1)+           ;(FLAG AND COUNT) INTO R
   (1) 023164  004737  013654                                  MOV     #RSSNSZ,SNDCNT(R5)  ;LOAD THE  SIZE TO S
   (1) 023170  010110                                          CALL    CHKSUM          ;R0 --> R1=COUNT
   (1)                                                         MOV     R1,(R0)         ;PUT CHKSUM IN PACKET
   (1) 023172  012765  000020  000034                                                  ;SET UP EXPECTATIONS:
   (1) 023200  012765  000001  000036                          MOV     #RSCONT,XSFLG(R5)   ;THE FLAG
   (1) 023206  012765  000001  000032                          MOV     #1,XSCNT(R5)    ;THE COUNT
   (1) 023214  012702  001000                                  MOV     #1,XSPKNM(R5)   ;THE # PACKETS EXPECTED
   (1) 023220  004737  006560                                  MOV     #512.,R2                ;GET # OF DATA B
   (1) 023224  032715  000010                                  CALL    RSVP            ;SEND (AND RETURN TO SCH
   (1) 023230  001314                                          BIT     #BIT3,@R5       ;FLAG BYTE ERROR?
   (1) 023232  042715  010000                                  BNE     72$             ;YES
   (1) 023236  012700  027722                                  BIC     #BIT12,@R5      ;FLAG FOR LAST PACKET
   (1) 023242  020227  000200                          64$:    MOV     #TRBUF,R0       ;POINT TO TOP OF BUFFER
   (1) 023246  101004                                          CMP     R2,#128.        ;START DATA PACKET(S)
   (1) 023250  010201                                          BHI     65$             ;#512. > 128.!
                                                               MOV     R2,R1           ;#512.<128.
```

```
(1)  023252  052715  010000                        BIS     #BIT12,@R5      ;SO LAST PACKET NOW
(1)  023256  000402                                BR      66$             ;USE REMAINING COUNT
(1)  023260  012701  000200                  65$:  MOV     #128.,R1        ;USE 128. BYTES
(1)  023264  110160  000001                  66$:  MOVB    R1,1(R0)        ;COPY COUNT TO BUFFER
(1)  023270  010103                                MOV     R1,R3           ;R3=COUNTER TO LOAD BUFF
(1)  023272  112710  000001                        MOVB    #RSDATA,@R0     ;FLAG FIRST
(1)  023276  005720                                TST     (R0)+           ;SKIP COUNT
(1)  023300  116520  000072                  67$:  MOVB    PATTEN(R5),(R0)+        ;INSERT DATA
(1)  023304  005303                                DEC     R3              ;MORE?
(1)  023306  101374                                BHI     67$             ;YES
(1)  023310  012700  027722                        MOV     #TRBUF,R0       ;-->TOP AGAIN
(1)  023314  116001  000001                        MOVB    1(R0),R1        ;GET COUNT
(1)  023320  042701  177400                        BIC     #177400,R1      ;ZERO SIGN EXTEND
(1)  023324  010165  000070                        MOV     R1,SNDCNT(R5)   ;HOW MANY TO SEND PLUS
(1)  023330  062765  000004  000070                ADD     #4,SNDCNT(R5)   ;FLAG,COUNT,CHKSUM
(1)  023336  062701  000002                        ADD     #2,R1           ;COMPENSATE FOR FLAG + C
(1)  023342  004737  013654                        CALL    CHKSUM          ;FOR CHECKSUM CALC.
(1)  023346  110120                                MOVB    R1,(R0)+        ;CHKSUM INTO PACKET
(1)  023350  000301                                SWAB    R1              ;EVEN ON AN ODD
(1)  023352  110120                                MOVB    R1,(R0)+        ;BYTE BOUNDARY
(1)  023354  032715  010000                        BIT     #BIT12,@R5      ;LAST DATA PACKET?
(1)  023360  001412                                BEQ     68$             ;NO
(1)  023362  012765  000002  000034                MOV     #RSEND,XSFLG(R5)  ;YES-EXPECT 'END'
(1)  023370  012765  000016  000036                MOV     #RSNDSZ,XSCNT(R5)  ;OF THIS SIZE
(1)  023376  012765  000001  000032                MOV     #1,XSPKNM(R5)   ;AND 1 PACKET
(1)  023404  000411                                BR      69$             ;SEND
(1)  023406  012765  000020  000034          68$:  MOV     #RSCONT,XSFLG(R5)  ;(NOT LAST), EXPECT '
(1)  023414  012765  000001  000036                MOV     #1,XSCNT(R5)    ;AND 1 BYTE
(1)  023422  012765  000001  000032                MOV     #1,XSPKNM(R5)   ;AND 1 PACKET
(1)  023430  004737  006560                  69$:  CALL    RSVP            ;SEND PACKET
(1)                                                                        ;AND RETURN TO SCHEDULER
(1)  023434  032715  000010                        BIT     #BIT3,@R5       ;FLAG BYTE RETRY?
(1)  023440  001210                                BNE     72$             ;YES
(1)  023442  032715  002000                        BIT     #BIT10,@R5      ;RETRY DATA ERROR?
(1)  023446  001004                                BNE     70$             ;YES
(1)  023450  162702  000200                        SUB     #128.,R2        ;NO, MORE DATA TO SEND?
(1)  023454  101270                                BHI     64$             ;YES
(1)  023456  000502                                BR      71$             ;NO
(1)  023460                                  70$:  TURTRY  REC(R5),#512.,DR(R5)    ;RETRY HERE
(2)
(2)
(2)  023460  012700  027722                  76$:  MOV     #TRBUF,R0       ;FORM CMND PACK:
(2)  023464  112710  000002                        MOVB    #RSCMND,@R0     ;MESSAGE PACK TYPE
(2)  023470  112760  000012  000001                MOVB    #RSMSIZ,1(R0)   ;THIS BIG
(2)  023476  112760  000002  000002                MOVB    #RSSRD,2(R0)    ;OP CODE-READ
(2)  023504  016560  000064  000012                MOV     REC(R5),10.(R0) ;THIS RECORD
(2)  023512  116560  000060  000004                MOVB    DR(R5),4.(R0)   ;THIS DRIVE
(2)  023520  105060  000003                        CLRB    3(R0)           ;PRESET NORM THRESHOLD
(2)  023524  105715                                TSTB    @R5             ;REDUCED?
(2)  023526  100002                                BPL     77$             ;NO
(2)  023530  105260  000003                        INCB    3(R0)           ;YES-CHANGE THRESHOLD
(2)  023534  012760  001000  000010          77$:  MOV     #512.,8.(R0)    ;# BYTES DESIRED
(2)  023542  112760  000020  000005                MOVB    #020,5.(R0)     ;MAINTENANCE MODE
(2)  023550  005060  000006                        CLR     6.(R0)          ;NO SEQUENCE #
(2)  023554  012701  000012                        MOV     #RSMSIZ,R1      ;SIZE OF PACKET
(2)  023560  005721                                TST     (R1)+           ;PLUS FLAG+COUNT INTO R1
```

```
    (2)  023562  012765  000016  000070                          MOV     #RSSNSZ,SNDCNT(R5)  ;SET UP SIZE TO SEND
    (2)
    (2)  023570  004737  013654                                  CALL    CHKSUM              ;FORM CHECKSUM R1=COUNT
    (2)  023574  010110                                          MOV     R1,(R0)             ;INSERT IN PACKET
    (2)
    (2)  023576  012701  001000                                  MOV     #512.,R1                    ;SET EXPECTATION
    (2)                                                                                      ;CALC # OF DATA PACKETS
    (2)  023602  012703  000034                                  MOV     #XSFLG,R3           ;OFFSET OF FLAG
    (2)  023606  060503                                          ADD     R5,R3               ;ABS. ADDR. OF XSFLG
    (2)  023610  005002                                          CLR     R2                  ;PRESET
    (2)  023612  005202                                  73$:    INC     R2                  ;# PACKETS EXPECTED
    (2)  023614  012723  000001                                  MOV     #RSDATA,(R3)+       ;LOAD XSFLG
    (2)  023620  012723  000204                                  MOV     #132.,(R3)+         ;AND EXPECT COUNT
    (2)  023624  162701  000200                                  SUB     #128.,R1            ;NEG RESULT LAST TIME
    (2)  023630  101401                                          BLOS    75$                 ;LAST TIME!
    (2)  023632  000767                                          BR      73$                 ;MORE TO DO
    (2)  023634  005202                                  75$:    INC     R2                  ;ADD ONE FOR END PACK
    (2)  023636  010265  000032                                  MOV     R2,XSPKNM(R5)       ;SAVE # PACKETS TO EXPEC
    (2)  023642  012723  000002                                  MOV     #RSEND,(R3)+        ;EXPECT AN END
    (2)  023646  012713  000016                                  MOV     #RSNDSZ,(R3)        ;THIS BIG-14. BYTES
    (2)
    (2)  023652  004737  006560                                  CALL    RSVP                ;SEND
    (2)                                                                                      ;AND RETURN TO SCHEDULER
    (2)

 7008  023666  005365  000066                  DEC     TMP(R5)         ;DO ALL RECORDS FOR THIS TRACK?
 7009  023672  001404                          BEQ     2$              ;YES-GET OTHER TRACK
 7010  023674  005265  000064                  INC     REC(R5)         ;NO-ONTO NEXT RECORD
 7011  023700  000137  023040                  JMP     1$              ;EXECUTE THE WRITE
 7012  023704  005765  000062          2$:     TST     TRK(R5)         ;DONE 2 TRACKS?
 7013  023710  001012                          BNE     TST5EX          ;YES-EXIT
 7014  023712  005265  000062                  INC     TRK(R5)         ;NO-SET FLAG FOR NEXT PASS
 7015  023716  013765  003334  000064          MOV     SECREC,REC(R5)  ;GET NEW STARTING BLOCK #
 7016  023724  013765  003310  000066          MOV     TAPLEN,TMP(R5)  ;RESET # OF BLOCKS
 7017  023732  000137  023040                  JMP     1$              ;AND EXECUTE
 7018  023736  005237  003322          TST5EX: INC     DONE            ;DONE
 7019  023742  000207                          RETURN                  ;RETURN
 7020
 7021  023744                                  ENDTST
   (3)  023744
   (3)  023744  104401                                          L10020: TRAP    CSETST
```

```
                        .SBTTL  TEST 6 / READ SELECTED NUMBER OF BLOCKS

                        BGNTST

                                                                        T6::
                        TSTID   #TST6
                                                MOV     #TST6,TSTTOP    ;SAVE ADDR OF TEST
                                                CALL    SETUF           ;INIT UNITS TSTPC
                                                CALL    SETDR           ;GET 1ST DRVS.
                                                CALL    RUN             ;DO TEST
                                                CALL    SWAPDR          ;GET NEXT DRVS.
                                                BCC     64$             ;BR NO 2ND DRVS
                                                CALL    SETUP           ;REINIT UNITS TSTPC
                                                CALL    RUN             ;REPEAT TEST
                                        64$:                            ;DONE
                        EXIT TST
                                                                        TRAP    C$EXIT
                                                                        .WORD   L10021-.

TST6:   CLR     REC(R5)                 ;START AT REC 0
        MOV     TAPLEN,TMP(R5)          ;GET THE # OF BLOCKS PER TRACK
        CLR     TRK(R5)                 ;TRK(R5)=1ST OR 2ND PASS
1$:     MOV     REC(R5),PATTEN(R5)      ;USE RECORD NO. AS DATA
        TST     DRVCHK                  ;ADD DR #?
        BEQ     10$                     ;NO
        ADD     DR(R5),PATTEN(R5)       ;ADD IN DRIVE ID
10$:    TUREAD  REC(R5),#512.,DR(R5),#0

68$:    MOV     #TRBUF,R0               ;FORM CMND PACK:
        MOVB    #RSCMND,@R0             ;MESSAGE PACK TYPE
        MOVB    #RSMSIZ,1(R0)           ;THIS BIG
        MOVB    #RSSRD,2(R0)            ;OP CODE IS READ
        MOV     REC(R5),10.(R0)         ;THIS RECORD
        MOVB    DR(R5),4.(R0)           ;THIS DRIVE
        MOVB    #0,3.(R0)               ;VERIFY
        MOV     #512.,8.(R0)            ;TOTAL BYTES TO READ
        MOVB    #020,5.(R0)             ;MAINTENANCE MODE
        CLR     6.(R0)                  ;NO SEQUENCE #
        MOV     #RSMSIZ,R1              ;GET SIZE OF PACKET
        TST     (R1)+                   ;+2 FOR CHECKSUM
        MOV     #RSSNSZ,SNDCNT(R5)      ;SIZE TO SEND
        CALL    CHKSUM                  ;FORM CHECKSUM R1=COUNT
        MOV     R1,(R0)                 ;INSERT CHECKSUM

        MOV     #512.,R1                        ;SET EXPECTATION
                                        ;CALC # OF DATA PACKETS
        MOV     #XSFLG,R3               ;GET OFFSET
        ADD     R5,R3                   ;ABS. ADDR. OF XSFLG
        CLR     R2                      ;PRESET AS NONE
64$:    INC     R2                      ;# PACKETS EXPECTED
        MOV     #RSDATA,(R3)+           ;LOAD XSFLG
        MOV     #132.,(R3)+             ;AND EXPECTED COUNT
        SUB     #128.,R1                ;NEG RESULT LAST TIME
        BLOS    66$                     ;LAST TIME
        BR      64$                     ;MORE TO DO

7024
7025
7026  023746
 (3)  023746
7027  023746
 (1)  023746  012737  024012  003326
 (1)  023754  004737  006004
 (1)  023760  004737  005632
 (1)  023764  004737  006052
 (1)  023770  004737  005530
 (1)  023774  103004
 (1)  023776  004737  006004
 (1)  024002  004737  006052
 (1)  024006
7028  024006
 (3)  024006  104432
 (3)  024010  000520
7029
7030
7031  024012  005065  000064
7032  024016  013765  003310  000066
7033  024024  005065  000062
7034  024030  016565  000064  000072
7035  024036  005737  002220
7036  024042  001403
7037  024044  066565  000060  000072
7038  024052
 (1)
 (1)
 (1)  024052  012700  027722
 (1)  024056  112710  000002
 (1)  024062  112760  000012  000001
 (1)  024070  112760  000002  000002
 (1)  024076  016560  000064  000012
 (1)  024104  116560  000060  000004
 (1)  024112  112760  000000  000003
 (1)  024120  012760  001000  000010
 (1)  024126  112760  000020  000005
 (1)  024134  005060  000006
 (1)  024140  012701  000012
 (1)  024144  005721
 (1)  024146  012765  000016  000070
 (1)  024154  004737  013654
 (1)  024160  010110
 (1)
 (1)  024162  012701  001000
 (1)
 (1)  024166  012703  000034
 (1)  024172  060503
 (1)  024174  005002
 (1)  024176  005205
 (1)  024200  012723  000001
 (1)  024204  012723  000204
 (1)  024210  162701  000200
 (1)  024214  101401
 (1)  024216  000767
```

```
    (1)  024220  005202
    (1)  024222  010265  000032
    (1)  024226  012723  000002
    (1)  024232  012713  000016
    (1)  024236  004737  006560
    (1)
    (1)  024242  032715  002010
    (1)  024246  001500
    (1)  024250
    (2)
    (2)
    (2)  024250  012700  027722
    (2)  024254  112710  000002
    (2)  024260  112760  000012  000001
    (2)  024266  112760  000002  000002
    (2)  024274  016560  000064  000012
    (2)  024302  116560  000060  000004
    (2)  024310  105060  000003
    (2)  024314  105715
    (2)  024316  100002
    (2)  024320  105260  000003
    (2)  024324  012760  001000  000010
    (2)  024332  112760  000020  000005
    (2)  024340  005060  000006
    (2)  024344  012701  000012
    (2)  024350  005721
    (2)  024352  012765  000016  000070
    (2)
    (2)  024360  004737  013654
    (2)  024364  010110
    (2)
    (2)  024366  012701  001000
    (2)
    (2)  024372  012703  000034
    (2)  024376  060503
    (2)  024400  005002
    (2)  024402  005202
    (2)  024404  012723  000001
    (2)  024410  012723  000204
    (2)  024414  162701  000200
    (2)  024420  101401
    (2)  024422  000767
    (2)  024424  005202
    (2)  024426  010265  000032
    (2)  024432  012723  000002
    (2)  024436  012713  000016
    (2)
    (2)  024442  004737  006560
    (2)
    (2)
7039  024452  005365  000066
7040  024456  001404
7041  024460  005265  000064
7042  024464  000137  024030
7043  024470  005765  000062
7044  024474  001012

66$:    INC     R2                      ;ADD ONE FOR END PACK
        MOV     R2,XSPKNM(R5)           ;SAVE # PACKETS TO EXPEC
        MOV     #RSEND,(R3)+            ;EXPECT AN END ALSO...
        MOV     #RSNDSZ,(R3)            ;THIS BIG-14. BYTES
        CALL    RSVP                    ;SEND
                                        ;AND RETURN TO SCHEDULER
67$:    BIT     #BIT10!BIT3,@R5         ;RETRY?
        BEQ     65$                     ;NO.
        TURTRY  REC(R5),#512.,DR(R5)    ;YES

72$:    MOV     #TRBUF,R0               ;FORM CMND PACK:
        MOVB    #RSCMND,@R0             ;MESSAGE PACK TYPE
        MOVB    #RSMSIZ,1(R0)           ;THIS BIG
        MOVB    #RSSRD,2(R0)            ;OP CODE-READ
        MOV     REC(R5),10.(R0)         ;THIS RECORD
        MOVB    DR(R5),4.(R0)           ;THIS DRIVE
        CLRB    3(R0)                   ;PRESET NORM THRESHOLD
        TSTB    @R5                     ;REDUCED?
        BPL     73$                     ;NO
        INCB    3(R0)                   ;YES-CHANGE THRESHOLD
73$:    MOV     #512.,8.(R0)            ;# BYTES DESIRED
        MOVB    #020,5.(R0)             ;MAINTENANCE MODE
        CLR     6.(R0)                  ;NO SEQUENCE #
        MOV     #RSMSIZ,R1              ;SIZE OF PACKET
        TST     (R1)+                   ;PLUS FLAG+COUNT INTO R1
        MOV     #RSSNSZ,SNDCNT(R5)      ;SET UP SIZE TO SEND

        CALL    CHKSUM                  ;FORM CHECKSUM R1=COUNT
        MOV     R1,(R0)                 ;INSERT IN PACKET

        MOV     #512.,R1                        ;SET EXPECTATION
                                        ;CALC # OF DATA PACKETS
        MOV     #XSFLG,R3               ;OFFSET OF FLAG
        ADD     R5,R3                   ;ABS. ADDR. OF XSFLG
        CLR     R2                      ;PRESET
69$:    INC     R2                      ;# PACKETS EXPECTED
        MOV     #RSDATA,(R3)+           ;LOAD XSFLG
        MOV     #132.,(R3)+             ;AND EXPECT COUNT
        SUB     #128.,R1                ;NEG RESULT LAST TIME
        BLOS    71$                     ;LAST TIME!
        BR      69$                     ;MORE TO DO
71$:    INC     R2                      ;ADD ONE FOR END PACK
        MOV     R2,XSPKNM(R5)           ;SAVE # PACKETS TO EXPEC
        MOV     #RSEND,(R3)+            ;EXPECT AN END
        MOV     #RSNDSZ,(R3)            ;THIS BIG-14. BYTES

        CALL    RSVP                    ;SEND
                                        ;AND RETURN TO SCHEDULER

        DEC     TMP(R5)                 ;DO ALL RECORDS THIS TRACK?
        BEQ     2$                      ;YES-GET OTHER TRACK
        INC     REC(R5)                 ;NO-NEXT RECORD
        JMP     1$                      ;EXECUTE THE READ
2$:     TST     TRK(R5)                 ;DONE 2 TRACKS?
        BNE     TST6EX                  ;YES-EXIT
```

```
                                                                H  9
 7045  024476  005265  000062                    INC     TRK(R5)             ;NO-SET FLAG FOR NEXT PASS
 7046  024502  013765  003334  000064            MOV     SECREC,REC(R5)      ;GET NEW STARTING BLOCK #
 7047  024510  013765  003310  000066            MOV     TAPLEN,TMP(R5)      ;RESET # OF BLOCKS
 7048  024516  000137  024030                    JMP     1$                  ;AND EXECUTE
 7049  024522  005237  003322              TST6EX: INC     DONE                ;DONE
 7050  024526  000207                            RETURN                      ;RETURN
 7051
 7052  024530                                    ENDTST
   (3)  024530
   (3)  024530  104401                                              L10021:
                                                                            TRAP    CSETST
```

```
                        .SBTTL  TEST 7 / WRITE-VERIFY SELECTED NUMBER OF BLOCKS

                                BGNTST

                                                                        T7::
                                TSTID   #TST7
                                                        MOV     #TST7,TSTTOP    ;SAVE ADDR OF TEST
                                                        CALL    SETUF           ;INIT UNITS TSTPC
                                                        CALL    SETDR           ;GET 1ST DRVS.
                                                        CALL    RUN             ;DO TEST
                                                        CALL    SWAPDR          ;GET NEXT DRVS.
                                                        BCC     64$             ;BR NO 2ND DRVS
                                                        CALL    SETUP           ;REINIT UNITS TSTPC
                                                        CALL    RUN             ;REPEAT TEST
                                                64$:                            ;DONE
                                EXIT TST

                                                                        TRAP    C$EXIT
                                                                        .WORD   L10022-.

                        TST7:   CLR     REC(R5)                 ;START AT REC 0
                                MOV     TAPLEN,TMP(R5)          ;GET THE # OF BLOCKS PER TRACK
                                CLR     TRK(R5)                 ;TRK(R5)=1ST OR 2ND PASS
                        1$:     MOV     REC(R5),PATTEN(R5)      ;USE RECORD NO. FOR DATA
                                TST     DRVCHK                  ;ADD DR #?
                                BEQ     10$                     ;NO
                                ADD     DR(R5),PATTEN(R5)               ;ADD DRIVE ID
                        10$:    TUWRIT  PATTEN(R5),REC(R5),#512.,DR(R5),#1
                                                72$:    MOV     #TRBUF,R0       ;MAKE COMMAND PACKET:
                                                        MOVB    #RSCMND,@R0     ;COMMAND FLAG
                                                        MOVB    #RSMSIZ,1(R0)   ;THIS SIZE
                                                        MOVB    #RSSWR,2(R0)    ;INSERT OP CODE-WRITE
                                                        MOVB    #1,3.(R0)       ;VERIFY (1 OR 0)
                                                        MOVB    DR(R5),4.(R0)   ;DRIVE #
                                                        MOVB    #020,5.(R0)     ;MAINTENANCE MODE SWITCH
                                                        CLR     6.(R0)          ;NO SEQUENCE #
                                                        MOV     #512.,8.(R0)    ;TOTAL COUNT TO WRITE
                                                        MOV     REC(R5),10.(R0) ;AT  RECORD N
                                                        MOV     #RSMSIZ,R1      ;THE PACKET SIZE PLUS+2
                                                        TST     (R1)+           ;(FLAG AND COUNT) INTO R
                                                        MOV     #RSSNSZ,SNDCNT(R5)  ;LOAD THE  SIZE TO S
                                                        CALL    CHKSUM          ;R0 --> R1=COUNT
                                                        MOV     R1,(R0)         ;PUT CHKSUM IN PACKET
                                                                                ;SET UP EXPECTATIONS:
                                                        MOV     #RSCONT,XSFLG(R5)   ;THE FLAG
                                                        MOV     #1,XSCNT(R5)    ;THE COUNT
                                                        MOV     #1,XSPKNM(R5)   ;THE # PACKETS EXPECTED
                                                        MOV     #512.,R2                ;GET # OF DATA B
                                                        CALL    RSVP            ;SEND (AND RETURN TO SCH
                                                        BIT     #BIT3,@R5       ;FLAG BYTE ERROR?
                                                        BNE     72$             ;YES
                                                        BIC     #BIT12,@R5      ;FLAG FOR LAST PACKET
                                                64$:    MOV     #TRBUF,R0       ;POINT TO TOP OF BUFFER
                                                        CMP     R2,#128.        ;START DATA PACKET(S)
                                                        BHI     65$             ;#512. > 128.!
                                                        MOV     R2,R1           ;#512.<128.
                                                        BIS     #BIT12,@R5      ;SO LAST PACKET NOW
```

```
7055
7056
7057  024532
 (3)  024532
7058  024532
 (1)  024532  012737  024576  003326      MOV     #TST7,TSTTOP    ;SAVE ADDR OF TEST
 (1)  024540  004737  006004              CALL    SETUF           ;INIT UNITS TSTPC
 (1)  024544  004737  005632              CALL    SETDR           ;GET 1ST DRVS.
 (1)  024550  004737  006052              CALL    RUN             ;DO TEST
 (1)  024554  004737  005530              CALL    SWAPDR          ;GET NEXT DRVS.
 (1)  024560  103004                      BCC     64$             ;BR NO 2ND DRVS
 (1)  024562  004737  006004              CALL    SETUP           ;REINIT UNITS TSTPC
 (1)  024566  004737  006052              CALL    RUN             ;REPEAT TEST
 (1)  024572                      64$:                            ;DONE
7059  024572
 (3)  024572  104432                      TRAP    C$EXIT
 (3)  024574  000724                      .WORD   L10022-.
7060
7061
7062  024576  005065  000064      TST7:   CLR     REC(R5)         ;START AT REC 0
7063  024602  013765  003310  000066      MOV     TAPLEN,TMP(R5)  ;GET THE # OF BLOCKS PER TRACK
7064  024610  005065  000062              CLR     TRK(R5)         ;TRK(R5)=1ST OR 2ND PASS
7065  024614  016565  000064  000072  1$: MOV     REC(R5),PATTEN(R5)  ;USE RECORD NO. FOR DATA
7066  024622  005737  002220              TST     DRVCHK          ;ADD DR #?
7067  024626  001403                      BEQ     10$             ;NO
7068  024630  066565  000060  000072      ADD     DR(R5),PATTEN(R5)       ;ADD DRIVE ID
7069  024636                      10$:    TUWRIT  PATTEN(R5),REC(R5),#512.,DR(R5),#1
 (1)  024636  012700  027722                      MOV     #TRBUF,R0       ;MAKE COMMAND PACKET:
 (1)  024642  112710  000002                      MOVB    #RSCMND,@R0     ;COMMAND FLAG
 (1)  024646  112760  000012  000001              MOVB    #RSMSIZ,1(R0)   ;THIS SIZE
 (1)  024654  112760  000003  000002              MOVB    #RSSWR,2(R0)    ;INSERT OP CODE-WRITE
 (1)  024662  112760  000001  000003              MOVB    #1,3.(R0)       ;VERIFY (1 OR 0)
 (1)  024670  116560  000060  000004              MOVB    DR(R5),4.(R0)   ;DRIVE #
 (1)  024676  112760  000020  000005              MOVB    #020,5.(R0)     ;MAINTENANCE MODE SWITCH
 (1)  024704  005060  000006                      CLR     6.(R0)          ;NO SEQUENCE #
 (1)  024710  012760  001000  000010              MOV     #512.,8.(R0)    ;TOTAL COUNT TO WRITE
 (1)  024716  016560  000064  000012              MOV     REC(R5),10.(R0) ;AT  RECORD N
 (1)  024724  012701  000012                      MOV     #RSMSIZ,R1      ;THE PACKET SIZE PLUS+2
 (1)  024730  005721                              TST     (R1)+           ;(FLAG AND COUNT) INTO R
 (1)  024732  012765  000016  000070              MOV     #RSSNSZ,SNDCNT(R5)  ;LOAD THE  SIZE TO S
 (1)  024740  004737  013654                      CALL    CHKSUM          ;R0 --> R1=COUNT
 (1)  024744  010110                              MOV     R1,(R0)         ;PUT CHKSUM IN PACKET
 (1)                                              ;SET UP EXPECTATIONS:
 (1)  024746  012765  000020  000034              MOV     #RSCONT,XSFLG(R5)   ;THE FLAG
 (1)  024754  012765  000001  000036              MOV     #1,XSCNT(R5)    ;THE COUNT
 (1)  024762  012765  000001  000032              MOV     #1,XSPKNM(R5)   ;THE # PACKETS EXPECTED
 (1)  024770  012702  001000                      MOV     #512.,R2                ;GET # OF DATA B
 (1)  024774  004737  006560                      CALL    RSVP            ;SEND (AND RETURN TO SCH
 (1)  025000  032715  000010                      BIT     #BIT3,@R5       ;FLAG BYTE ERROR?
 (1)  025004  001314                              BNE     72$             ;YES
 (1)  025006  042715  010000                      BIC     #BIT12,@R5      ;FLAG FOR LAST PACKET
 (1)  025012  012700  027722              64$:    MOV     #TRBUF,R0       ;POINT TO TOP OF BUFFER
 (1)  025016  020227  000200                      CMP     R2,#128.        ;START DATA PACKET(S)
 (1)  025022  101004                              BHI     65$             ;#512. > 128.!
 (1)  025024  010201                              MOV     R2,R1           ;#512.<128.
 (1)  025026  052715  010000                      BIS     #BIT12,@R5      ;SO LAST PACKET NOW
```

```
(1)  025032  000402                        BR      66$             ;USE REMAINING COUNT
(1)  025034  012701  000200          65$:  MOV     #128.,R1        ;USE 128. BYTES
(1)  025040  110160  000001          66$:  MOVB    R1,1(R0)        ;COPY COUNT TO BUFFER
(1)  025044  010103                        MOV     R1,R3           ;R3=COUNTER TO LOAD BUFF
(1)  025046  112710  000001                MOVB    #RSDATA,@R0     ;FLAG FIRST
(1)  025052  005720                        TST     (R0)+           ;SKIP COUNT
(1)  025054  116520  000072          67$:  MOVB    PATTEN(R5),(R0)+        ;INSERT DATA
(1)  025060  005303                        DEC     R3              ;MORE?
(1)  025062  101374                        BHI     67$             ;YES
(1)  025064  012700  027722                MOV     #TRBUF,R0       ;-->TOP AGAIN
(1)  025070  116001  000001                MOVB    1(R0),R1        ;GET COUNT
(1)  025074  042701  177400                BIC     #177400,R1      ;ZERO SIGN EXTEND
(1)  025100  010165  000070                MOV     R1,SNDCNT(R5)   ;HOW MANY TO SEND PLUS
(1)  025104  062765  000004  000070        ADD     #4,SNDCNT(R5)   ;FLAG,COUNT,CHKSUM
(1)  025112  062701  000002                ADD     #2,R1           ;COMPENSATE FOR FLAG + C
(1)  025116  004737  013654                CALL    CHKSUM          ;FOR CHECKSUM CALC.
(1)  025122  110120                        MOVB    R1,(R0)+        ;CHKSUM INTO PACKET
(1)  025124  000301                        SWAB    R1              ;EVEN ON AN ODD
(1)  025126  110120                        MOVB    R1,(R0)+        ;BYTE BOUNDARY
(1)  025130  032715  010000                BIT     #BIT12,@R5      ;LAST DATA PACKET?
(1)  025134  001412                        BEQ     68$             ;NO
(1)  025136  012765  000002  000034        MOV     #RSEND,XSFLG(R5)  ;YES-EXPECT 'END'
(1)  025144  012765  000016  000036        MOV     #RSNDS2,XSCNT(R5)  ;OF THIS SIZE
(1)  025152  012765  000001  000032        MOV     #1,XSPKNM(R5)   ;AND 1 PACKET
(1)  025160  000411                        BR      69$             ;SEND
(1)  025162  012765  000020  000034  68$:  MOV     #RSCONT,XSFLG(R5)  ;(NOT LAST), EXPECT '
(1)  025170  012765  000001  000036        MOV     #1,XSCNT(R5)    ;AND 1 BYTE
(1)  025176  012765  000001  000032        MOV     #1,XSPKNM(R5)   ;AND 1 PACKET
(1)  025204  004737  006560          69$:  CALL    RSVP            ;SEND PACKET
(1)                                                                ;AND RETURN TO SCHEDULER
(1)  025210  032715  000010                BIT     #BIT3,@R5       ;FLAG BYTE RETRY?
(1)  025214  001210                        BNE     72$             ;YES
(1)  025216  032715  002000                BIT     #BIT10,@R5      ;RETRY DATA ERROR?
(1)  025222  001004                        BNE     70$             ;YES
(1)  025224  162702  000200                SUB     #128.,R2        ;NO, MORE DATA TO SEND?
(1)  025230  101270                        BHI     64$             ;YES
(1)  025232  000502                        BR      71$             ;NO
(1)  025234                          70$:  TURTRY  REC(R5),#512.,DR(R5)     ;RETRY HERE
(2)
(2)
(2)  025234  012700  027722          76$:  MOV     #TRBUF,R0       ;FORM CMND PACK:
(2)  025240  112710  000002                MOVB    #RSCMND,@R0     ;MESSAGE PACK TYPE
(2)  025244  112760  000012  000001        MOVB    #RSMSIZ,1(R0)   ;THIS BIG
(2)  025252  112760  000002  000002        MOVB    #RSSRD,2(R0)    ;OP CODE-READ
(2)  025260  016560  000064  000012        MOV     REC(R5),10.(R0) ;THIS RECORD
(2)  025266  116560  000060  000004        MOVB    DR(R5),4.(R0)   ;THIS DRIVE
(2)  025274  105060  000003                CLRB    3(R0)           ;PRESET NORM THRESHOLD
(2)  025300  105715                        TSTB    @R5             ;REDUCED?
(2)  025302  100002                        BPL     77$             ;NO
(2)  025304  105260  000003                INCB    3(R0)           ;YES-CHANGE THRESHOLD
(2)  025310  012760  001000  000010  77$:  MOV     #512.,8.(R0)    ;# BYTES DESIRED
(2)  025316  112760  000020  000005        MOVB    #020,5.(R0)     ;MAINTENANCE MODE
(2)  025324  005060  000006                CLR     6.(R0)          ;NO SEQUENCE #
(2)  025330  012701  000012                MOV     #RSMSIZ,R1      ;SIZE OF PACKET
(2)  025334  005721                        TST     (R1)+           ;PLUS FLAG+COUNT INTO R1
(2)  025336  012765  000016  000070        MOV     #RSSNSZ,SNDCNT(R5)  ;SET UP SIZE TO SEND
```

```
    (2)
    (2)  025344  004737  013654                          CALL    CHKSUM          ;FORM CHECKSUM R1=COUNT
    (2)  025350  010110                                  MOV     R1,(R0)         ;INSERT IN PACKET
    (2)
    (2)  025352  012701  001000                          MOV     #512.,R1                ;SET EXPECTATION
    (2)                                                                          ;CALC # OF DATA PACKETS
    (2)  025356  012703  000034                          MOV     #XSFLG,R3       ;OFFSET OF FLAG
    (2)  025362  060503                                  ADD     R5,R3           ;ABS. ADDR. OF XSFLG
    (2)  025364  005002                                  CLR     R2              ;PRESET
    (2)  025366  005202                              73$:INC     R2              ;# PACKETS EXPECTED
    (2)  025370  012725  000001                          MOV     #RSDATA,(R3)+   ;LOAD XSFLG
    (2)  025374  012725  000204                          MOV     #132.,(R3)+     ;AND EXPECT COUNT
    (2)  025400  162701  000200                          SUB     #128.,R1        ;NEG RESULT LAST TIME
    (2)  025404  101401                                  BLOS    75$             ;LAST TIME!
    (2)  025406  000767                                  BR      73$             ;MORE TO DO
    (2)  025410  005202                              75$:INC     R2              ;ADD ONE FOR END PACK
    (2)  025412  010265  000032                          MOV     R2,XSPKNM(R5)   ;SAVE # PACKETS TO EXPEC
    (2)  025416  012725  000002                          MOV     #RSEND,(R3)+    ;EXPECT AN END
    (2)  025422  012713  000016                          MOV     #RSNDSZ,(R3)    ;THIS BIG-14. BYTES
    (2)
    (2)  025426  004737  006560                          CALL    RSVP            ;SEND
    (2)                                                                          ;AND RETURN TO SCHEDULER
    (2)
 7070  025442  005365  000066                      DEC     TMP(R5)         ;DO ALL RECORDS FOR THIS TRACK?
 7071  025446  001404                              BEQ     2$              ;YES-GET OTHER TRACK
 7072  025450  005265  000064                      INC     REC(R5)         ;NO-NEXT RECORD
 7073  025454  000137  024614                      JMP     1$              ;EXECUTE THE WRITE
 7074  025460  005765  000062                  2$: TST     TRK(R5)         ;DONE 2 TRACKS?
 7075  025464  001012                              BNE     TST7EX          ;YES-EXIT
 7076  025466  005265  000062                      INC     TRK(R5)         ;NO-SET FLAG FOR NEXT PASS
 7077  025472  013765  003334  000064              MOV     SECREC,REC(R5)  ;GET NEW STARTING BLOCK #
 7078  025500  013765  003310  000066              MOV     TAPLEN,TMP(R5)  ;RESET # OF BLOCKS
 7079  025506  000137  024614                      JMP     1$              ;AND EXECUTE
 7080  025512  005237  003322                  TST7EX: INC     DONE            ;DONE
 7081  025516  000207                              RETURN                  ;RETURN
 7082
 7083  025520                                      ENDTST
  (3)  025520
  (3)  025520  104401                                                  L10022:
                                                                               TRAP    CSETST
```

```
                                .SBTTL  TEST 8 / READ-REDUCED THRESHOLD SELECTED NUMBER OF BLOCKS
                                        BGNTST
                                                                                T8::
                                        TSTID   #TST8
025522                                                  MOV     #TST8,TSTTOP    ;SAVE ADDR OF TEST
025522                                                  CALL    SETUF           ;INIT UNITS TSTPC
025522                                                  CALL    SETDR           ;GET 1ST DRVS.
025522                                                  CALL    RUN             ;DO TEST
025522  012737  025566  003326                          CALL    SWAPDR          ;GET NEXT DRVS.
025530  004737  006004                                  BCC     64$             ;BR NO 2ND DRVS
025534  004737  005632                                  CALL    SETUP           ;REINIT UNITS TSTPC
025540  004737  006052                                  CALL    RUN             ;REPEAT TEST
025544  004737  005530                                                          ;DONE
025550  103004                                  64$:
025552  004737  006004                                  EXIT TST
025556  004737  006052
025562                                                                  TRAP    C$EXIT
025562                                                                  .WORD   L10023-.
025564  104432
025564  000520

025566  005065  000064  TST8:   CLR     REC(R5)         ;START AT REC 0
025572  013765  003310  000066          MOV     TAPLEN,TMP(R5)  ;GET THE # OF BLOCKS PER TRACK
025600  005065  000062          CLR     TRK(R5)         ;TRK(R5)=1ST OR 2ND PASS
025604  016565  000064  000072  1$:     MOV     REC(R5),PATTEN(R5)  ;USE RECORD NO. FOR DATA
025612  005737  002220          TST     DRVCHK          ;ADD DR #?
025616  001403                  BEQ      10$            ;NO
025620  066565  000060  000072          ADD     DR(R5),PATTEN(R5)       ;ADD DRIVE ID
025626                  10$:    TUREAD  REC(R5),#512.,DR(R5),#1
(1)
(1)
(1)     025626  012700  027722  68$:    MOV     #TRBUF,R0       ;FORM CMND PACK:
(1)     025632  112710  000002          MOVB    #RSCMND,@R0     ;MESSAGE PACK TYPE
(1)     025636  112760  000012  000001  MOVB    #RSMSIZ,1(R0)   ;THIS BIG
(1)     025644  112760  000002  000002  MOVB    #RSSRD,2(R0)    ;OP CODE IS READ
(1)     025652  016560  000064  000012  MOV     REC(R5),10.(R0) ;THIS RECORD
(1)     025660  116560  000060  000004  MOVB    DR(R5),4.(R0)   ;THIS DRIVE
(1)     025666  112760  000001  000003  MOVB    #1,3.(R0)       ;VERIFY
(1)     025674  012760  001000  000010  MOV     #512.,8.(R0)    ;TOTAL BYTES TO READ
(1)     025702  112760  000020  000005  MOVB    #020,5.(R0)     ;MAINTENANCE MODE
(1)     025710  005060  000006          CLR     6.(R0)          ;NO SEQUENCE #
(1)     025714  012701  000012          MOV     #RSMSIZ,R1      ;GET SIZE OF PACKET
(1)     025720  005721                  TST     (R1)+           ;+2 FOR CHECKSUM
(1)     025722  012765  000016  000070  MOV     #RSSNSZ,SNDCNT(R5)  ;SIZE TO SEND
(1)     025730  004737  013654          CALL    CHKSUM          ;FORM CHECKSUM R1=COUNT
(1)     025734  010110                  MOV     R1,(R0)         ;INSERT CHECKSUM
(1)
(1)     025736  012701  001000          MOV     #512.,R1                ;SET EXPECTATION
(1)                                                             ;CALC # OF DATA PACKETS
(1)     025742  012703  000034          MOV     #XSFLG,R3       ;GET OFFSET
(1)     025746  060503                  ADD     R5,R3           ;ABS. ADDR. OF XSFLG
(1)     025750  005002                  CLR     R2              ;PRESET AS NONE
(1)     025752  005202          64$:    INC     R2              ;# PACKETS EXPECTED
(1)     025754  012723  000001          MOV     #RSDATA,(R3)+   ;LOAD XSFLG
(1)     025760  012723  000204          MOV     #132.,(R3)+     ;AND EXPECTED COUNT
(1)     025764  162701  000200          SUB     #128.,R1        ;NEG RESULT LAST TIME
(1)     025770  101401                  BLOS    66$             ;LAST TIME
(1)     025772  000767                  BR      64$             ;MORE TO DO
```

```
(1)  025774  005202
(1)  025776  010265  000032
(1)  026002  012723  000002
(1)  026006  012713  000016
(1)  026012  004737  006560
(1)
(1)  026016  032715  002010
(1)  026022  001500
(1)  026024
(2)
(2)  026024  012700  027722
(2)  026030  112710  000003
(2)  026034  112760  000012  000001
(2)  026042  112760  000002  000002
(2)  026050  016560  000064  000012
(2)  026056  116560  000060  000004
(2)  026064  105060  000003
(2)  026070  105715
(2)  026072  100002
(2)  026074  105260  000003
(2)  026100  012760  001000  000010
(2)  026106  112760  000020  000005
(2)  026114  005060  000006
(2)  026120  012701  000012
(2)  026124  005721
(2)  026126  012765  000016  000070
(2)
(2)  026134  004737  013654
(2)  026140  010110
(2)
(2)  026142  012701  001000
(2)
(2)  026146  012703  000034
(2)  026152  060503
(2)  026154  005002
(2)  026156  005202
(2)  026160  012723  000001
(2)  026164  012723  000204
(2)  026170  162701  000200
(2)  026174  101401
(2)  026176  000767
(2)  026200  005202
(2)  026202  010265  000032
(2)  026206  012723  000002
(2)  026212  012713  000016
(2)
(2)  026216  004737  006560
(2)
(2)
```

```
66$:    INC     R2                  ;ADD ONE FOR END PACK
        MOV     R2,XSPKNM(R5)       ;SAVE # PACKETS TO EXPEC
        MOV     #RSEND,(R3)+        ;EXPECT AN END ALSO...
        MOV     #RSNDSZ,(R3)        ;THIS BIG-14. BYTES
        CALL    RSVP                ;SEND
                                    ;AND RETURN TO SCHEDULER
67$:    BIT     #BIT10!BIT3,@R5     ;RETRY?
        BEQ     65$                 ;NO.
        TURTRY  REC(R5),#512.,DR(R5)    ;YES

72$:    MOV     #TRBUF,R0           ;FORM CMND PACK:
        MOVB    #RSCMND,@R0         ;MESSAGE PACK TYPE
        MOVB    #RSMSIZ,1(R0)       ;THIS BIG
        MOVB    #RSSRD,2(R0)        ;OP CODE-READ
        MOV     REC(R5),10.(R0)     ;THIS RECORD
        MOVB    DR(R5),4.(R0)       ;THIS DRIVE
        CLRB    3(R0)               ;PRESET NORM THRESHOLD
        TSTB    @R5                 ;REDUCED?
        BPL     73$                 ;NO
        INCB    3(R0)               ;YES-CHANGE THRESHOLD
73$:    MOV     #512.,8.(R0)        ;# BYTES DESIRED
        MOVB    #020,5.(R0)         ;MAINTENANCE MODE
        CLR     6.(R0)              ;NO SEQUENCE #
        MOV     #RSMSIZ,R1          ;SIZE OF PACKET
        TST     (R1)+               ;PLUS FLAG+COUNT INTO R1
        MOV     #RSSNSZ,SNDCNT(R5)  ;SET UP SIZE TO SEND

        CALL    CHKSUM              ;FORM CHECKSUM R1=COUNT
        MOV     R1,(R0)             ;INSERT IN PACKET

        MOV     #512.,R1                    ;SET EXPECTATION
                                    ;CALC # OF DATA PACKETS
        MOV     #XSFLG,R3           ;OFFSET OF FLAG
        ADD     R5,R3               ;ABS. ADDR. OF XSFLG
        CLR     R2                  ;PRESET
69$:    INC     R2                  ;# PACKETS EXPECTED
        MOV     #RSDATA,(R3)+       ;LOAD XSFLG
        MOV     #132.,(R3)+         ;AND EXPECT COUNT
        SUB     #128.,R1            ;NEG RESULT LAST TIME
        BLOS    71$                 ;LAST TIME!
        BR      69$                 ;MORE TO DO
71$:    INC     R2                  ;ADD ONE FOR END PACK
        MOV     R2,XSPKNM(R5)       ;SAVE # PACKETS TO EXPEC
        MOV     #RSEND,(R3)+        ;EXPECT AN END
        MOV     #RSNDSZ,(R3)        ;THIS BIG-14. BYTES

        CALL    RSVP                ;SEND
                                    ;AND RETURN TO SCHEDULER
```

```
7101  026226  005365  000066        DEC     TMP(R5)         ;DO ALL RECORDS THIS TRACK?
7102  026232  001404                BEQ     2$              ;YES-GET OTHER TRACK
7103  026234  005265  000064        INC     REC(R5)         ;NO-NEXT RECORD
7104  026240  000137  025604        JMP     1$              ;EXECUTE THE READ
7105  026244  005765  000062  2$:   TST     TRK(R5)         ;DONE 2 TRACKS?
7106  026250  001012                BNE     TST8EX          ;YES-EXIT
```

```
                                                    N  9
 7107  026252  005265  000062
 7108  026256  013765  003334  000064          INC     TRK(R5)             ;NO-SET FLAG FOR NEXT PASS
 7109  026264  013765  003310  000066          MOV     SECREC,REC(R5)      ;GET NEW STARTING BLOCK #
 7110  026272  000137  025604                  MOV     TAPLEN,TMP(R5)      ;RESET # OF BLOCKS
 7111  026276  005237  003322                  JMP     1$                  ;AND EXECUTE
 7112  026302  000207                TST8EX:   INC     DONE                ;DONE
 7113                                          RETURN                      ;RETURN
 7114  026304
   (3) 026304                                 ENDTST
   (3) 026304  104401                                                     L10023:
                                                                                TRAP    CSETST
```

```
7117
7118                                    .SBTTL  TEST 9 / TESTS MODIFIED RADIAL SERIAL PROTOCOL
7119  026306                                    BGNTST
 (3)  026306
7120                                                                        T9::
7121  026306  012737  026330  003326            MOV     #TST9,TSTTOP      ;SAVE ADDR OF TEST
7122  026314  004737  006004                    CALL    SETUP             ;INIT UNITS TSTPC
7123  026320  004737  006052                    CALL    RUN               ;DO TEST
7124                                                                      ;DONE
7125
7126  026324                                    EXIT TST
 (3)  026324  104432
 (3)  026326  000662                                                      TRAP    CSEXIT
7127                                                                      .WORD   L10024-.
7128  026330  012737  000001  003342  TST9:    MOV     #1,TEST9          ;INDICATES 1ST PART OF TST 8
7129  026336  012700  027722  64$:     MOV     #TRBUF,R0         ;FORM COMMAND PACKET
7130  026342  112710  000002                  MOVB    #RSCMND,@R0       ;COMMAND FLAG
7131  026346  112760  000012  000001        MOVB    #RSMSIZ,1(R0)     ;SIZE OF MESSAGE
7132  026354  112760  000012  000002        MOVB    #RSSGET,2(R0)     ;GET CHARACTERISTICS
7133  026362  105060  000003                  CLRB    3(R0)             ;NO MODIFIER.
7134  026366  005060  000004                  CLR     4(R0)             ;NO DRIVE OR SWITCHES
7135  026372  005060  000006                  CLR     6(R0)             ;NO SEQUENCE NUMBER
7136  026376  005060  000010                  CLR     8.(R0)            ;NO BYTES
7137  026402  005060  000012                  CLR     10.(R0)           ;NO RECORD #
7138  026406  012701  000012                  MOV     #RSMSIZ,R1        ;GET SIZE
7139  026412  005721                          TST     (R1)+             ;+2 FOR CHECKSUM
7140  026414  012765  000016  000070          MOV     #RSSNSZ,SNDCNT(R5)   ;SIZE TO SEND
7141  026422  004737  013654                  CALL    CHKSUM            ;FORM CHECKSUM
7142  026426  010110                          MOV     R1,(R0)           ;INSERT INTO PACKET
7143  026430  012765  000001  000034          MOV     #RSDATA,XSFLG(R5)   ;EXPECT DATA PACKET
7144  026436  012765  000034  000036          MOV     #RSGCDP,XSCNT(R5)   ;THIS BIG
7145  026444  012765  000001  000032          MOV     #1,XSPKNM(R5)     ;AND 1 PACKET
7146                                                                      ;SEND
7147  026452  004737  006560                  CALL    RSVP              ;RETURN TO SCHEDULER
7148
7149  026456  004737  014010                  CALL    DOBRK             ;CLR POTENTIAL INTERFACE ERROR
7150
7151  026462  032715  000010                  BIT     #BIT3,@R5         ;RETRY?(BAD FLAG)
7152  026466  001323                          BNE     64$               ;YES
7153
7154  026470  012737  000002  003342          MOV     #2,TEST9          ;INDICATE 2ND PART OF TST 8
7155
7156  026476  012700  027722  65$:     MOV     #TRBUF,R0         ;-->(POINT TO) XMIT BUFFER
7157  026502  112710  000002                  MOVB    #RSCMND,@R0       ;FORM COMMAND MESSAGE PACK
7158  026506  112760  000012  000001        MOVB    #RSMSIZ,1(R0)     ;THIS BIG
7159  026514  112760  000001  000002        MOVB    #RSSNIT,2(R0)     ;OP CODE IS INITIALIZE
7160  026522  013760  000064  000012        MOV     REC,10.(R0)       ;TO THIS RECORD
7161  026530  105060  000003                  CLRB    3.(R0)            ;NO MODIFIER
7162  026534  105060  000004                  CLRB    4.(R0)            ;NO DRIVE
7163  026540  112760  000010  000005        MOVB    #BIT03,5.(R0)     ;SET MRSP SWITCH
7164  026546  005060  000006                  CLR     6.(R0)            ;NO SEQUENCE #
7165  026552  005060  000010                  CLR     8.(R0)            ;NO BYTE COUNT
7166  026556  012701  000012                  MOV     #RSMSIZ,R1        ;GET COUNT
7167  026562  005721                          TST     (R1)+             ;PLUS FLAG + BCNT
7168                                                                      ;FOR CHECKSUM CALC
7169  026564  004737  013654                  CALL    CHKSUM            ;R0-->TOP   R1=# OF BYTES
```

```
 7170  026570  010110                          MOV     R1,(R0)          ;INSERT INTO PACKET
 7171                                                                   ;SET UP EXPECTATIONS:
 7172  026572  012765  000016  000070          MOV     #RSSNSZ,SNDCNT(R5)   ;HOW MANY TO SEND
 7173  026600  112765  000002  000034          MOVB    #RSCMND,XSFLG(R5)   ;EXPECT END PACK
 7174  026606  012765  000016  000036          MOV     #RSNDSZ,XSCNT(R5)   ;COUNT WITH THIS
 7175  026614  012765  000001  000032          MOV     #1.,XSPKNM(R5)   ;EXPECT ONLY 1 PACKET
 7176
 7177  026622  004737  006560                  CALL    RSVP             ;SEND
 7178                                                                   ;AND RETURN TO SCHEDULER
 7179
 7180  026626  032715  000010                  BIT     #BIT3,@R5        ;RETRY (FLAG BYTE ERROR)?
 7181  026632  001321                          BNE     65$              ;YES
 7182
 7183  026634  012700  027722          66$:    MOV     #TRBUF,R0        ;-->(POINT TO) XMIT BUFFER
 7184  026640  112710  000002                  MOVB    #RSCMND,@R0      ;FORM COMMAND MESSAGE PACK
 7185  026644  112760  000012  000001          MOVB    #RSMSIZ,1(R0)    ;THIS BIG
 7186  026652  112760  000000  000002          MOVB    #RSSNOP,2(R0)    ;OP CODE IS NO-OPERATION
 7187  026660  013760  000064  000012          MOV     REC,10.(R0)      ;TO THIS RECORD
 7188  026666  105060  000003                  CLRB    3.(R0)           ;NO MODIFIER
 7189  026672  105060  000004                  CLRB    4.(R0)           ;NO DRIVE
 7190  026676  112760  000010  000005          MOVB    #BIT03,5.(R0)    ;SET MRSP SWITCH
 7191  026704  005060  000006                  CLR     6.(R0)           ;NO SEQUENCE #
 7192  026710  005060  000010                  CLR     8.(R0)           ;NO BYTE COUNT
 7193  026714  012701  000012                  MOV     #RSMSIZ,R1       ;GET COUNT
 7194  026720  005721                          TST     (R1)+            ;PLUS FLAG + BCNT
 7195                                                                   ;FOR CHECKSUM CALC
 7196  026722  004737  013654                  CALL    CHKSUM           ;R0-->TOP   R1=# OF BYTES
 7197  026726  010110                          MOV     R1,(R0)          ;INSERT INTO PACKET
 7198                                                                   ;SET UP EXPECTATIONS:
 7199  026730  012765  000016  000070          MOV     #RSSNSZ,SNDCNT(R5)   ;HOW MANY TO SEND
 7200  026736  112765  000002  000034          MOVB    #RSCMND,XSFLG(R5)   ;EXPECT END PACK
 7201  026744  012765  000016  000036          MOV     #RSNDSZ,XSCNT(R5)   ;COUNT WITH THIS
 7202  026752  012765  000001  000032          MOV     #1.,XSPKNM(R5)   ;EXPECT ONLY 1 PACKET
 7203
 7204  026760  004737  006560                  CALL    RSVP             ;SEND
 7205                                                                   ;AND RETURN TO SCHEDULER
 7206
 7207  026764  032715  000010                  BIT     #BIT3,@R5        ;RETRY (FLAG BYTE ERROR)?
 7208  026770  001321                          BNE     66$              ;YES
 7209
 7210  026772  005237  003322                  INC     DONE
 7211  026776  005037  003342                  CLR     TEST9
 7212
 7213  027002  005737  002224                  TST     PPSOT9           ;PROTOCOL SUMMARY @ END OF PASS
 7214  027006  001477                          BEQ     ENDT9            ;NO
 7215  027010  005037  027366                  CLR     UNITNO           ;SET UNIT # TO ZERO
 7216  027014                                  PRINTF  #MSAGE1          ;PRINT HEADER
  (7)  027014  012746  027212                                                  MOV     #MSAGE1,
  (6)  027020  012746  000001                                                  MOV     #1.,-(SP)
  (3)  027024  010600                                                          MOV     SP,R0
  (4)  027026  104417                                                          TRAP    C$PNTF
  (4)  027030  062706  000004                                                  ADD     #4,SP
 7217  027034  012737  003350  003312          MOV     #BLKTBL,DEVPTR   ;SET ALL UNITS
 7218  027042  017705  154244          1$:     MOV     @DEVPTR,R5       ;GET POINTER
 7219  027046  005765  000000                  TST     STATUS(R5)       ;IS UNIT ABORTED
 7220  027052  100431                          BMI     3$               ;YES
```

```
                                                                D 10
 7221  027054  005765  000210                TST     MRSP(R5)          ;IS UNIT MODIFIED
 7222  027060  001413                        BEQ     2$                ;NO
 7223  027062                                PRINTF  #MSAGE2,UNITNO    ;MESSAGE FOR MODIFIED UNIT
   (8) 027062  013746  027366                                                  MOV     UNITNO,-
   (7) 027066  012746  027253                                                  MOV     #MSAGE2,
   (6) 027072  012746  000002                                                  MOV     #2,-(SP)
   (3) 027076  010600                                                          MOV     SP,R0
   (4) 027100  104417                                                          TRAP    C$PNTF
   (4) 027102  062706  000006                                                  ADD     #6,SP
 7224  027106  000425                        BR      4$                ;SEE IF LAST UNIT
 7225  027110                          2$:   PRINTF  #MSAGE3,UNITNO    ;MESSAGE FOR NON-MODIFIED UNIT
   (8) 027110  013746  027366                                                  MOV     UNITNO,-
   (7) 027114  012746  027307                                                  MOV     #MSAGE3,
   (6) 027120  012746  000002                                                  MOV     #2,-(SP)
   (3) 027124  010600                                                          MOV     SP,R0
   (4) 027126  104417                                                          TRAP    C$PNTF
   (4) 027130  062706  000006                                                  ADD     #6,SP
 7226  027134  000412                        BR      4$                ;SEE IF LAST UNIT
 7227  027136                          3$:   PRINTF  #MSAGE4,UNITNO    ;MESSAGE FOR ABORTED UNIT
   (8) 027136  013746  027366                                                  MOV     UNITNO,-
   (7) 027142  012746  027336                                                  MOV     #MSAGE4,
   (6) 027146  012746  000002                                                  MOV     #2,-(SP)
   (3) 027152  010600                                                          MOV     SP,R0
   (4) 027154  104417                                                          TRAP    C$PNTF
   (4) 027156  062706  000006                                                  ADD     #6,SP
 7228  027162  023727  003312  003366    4$:   CMP     DEVPTR,#LSTDEV    ;IS THIS THE LAST DEVICE
 7229  027170  103006                        BHIS    ENDT9             ;YES
 7230  027172  062737  000002  003312        ADD     #2,DEVPTR         ;GET NEXT UNIT
 7231  027200  005237  027366                INC     UNITNO            ;INC UNIT #
 7232  027204  000716                        BR      1$
 7233
 7234  027206  000207              ENDT9:    RETURN
 7235
 7236  027210                              ENDTST
   (3) 027210                                                                L10024:
   (3) 027210  104401                                                                TRAP    C$ETST
 7237
 7238  027212  047045  051445  022470  MSAGE1: .ASCIZ  /%N%S8%AUNIT NO%S9%S6%APROTOCOL%N/
 7239  027253     045  022516  034523  MSAGE2: .ASCIZ  !%N%S9%S2%01%S9%S9%ARSP/MRSP!
 7240  027307     045  022516  034523  MSAGE3: .ASCIZ  /%N%S9%S2%01%S9%S9%ARSP/
 7241  027336  047045  051445  022471  MSAGE4: .ASCIZ  /%N%S9%S2%01%S9%S9%A---/
 7242          027366                          .EVEN
 7243  027366  000000              UNITNO: .WORD
```

```
 7246
 7247                                   .SBTTL  PATCH AREA
 7248  000144                                   .REPT   100.
 7249                                           .WORD
                                                .ENDR
```

```
7252
7253
7254
7255
7256  027700  030760
7257  027702  032016
7258  027704  033054
7259  027706  034112
7260  027710  035150
7261  027712  036206
7262  027714  037244
7263  027716  040302
7264
7265
7266
7267
7268
7269  027720  000023
7270  027721  000023
7271
7272  027722  001036
7273
7274
7275
7276  030760  001036
7277  032016  001036
7278  033054  001036
7279  034112  001036
7280  035150  001036
7281  036206  001036
7282  037244  001036
7283  040302  001036
7284
7285
7286
7287  041340

                        .SBTTL  I/O BUFFER AREAS:

                        ;WHO-GETS-WHAT-SPACE TABLE

                        BUFTBL: .WORD   BUF0
                                .WORD   BUF1
                                .WORD   BUF2
                                .WORD   BUF3
                                .WORD   BUF4
                                .WORD   BUF5
                                .WORD   BUF6
                                .WORD   BUF7

                        ;-----------------------------------------------------------------------
                        ;ONLY 1 TRANSMIT BUFFER NECESSARY:

                                        .BYTE RSXOFF
                                        .BYTE RSXOFF    ;SEND XOFF BEFORE EVERY PACKET

                                        TRBUF:  .BLKB RCBFSZ
                        ;-----------------------------------------------------------------------

                        BUF0:   .BLKB   RCBFSZ
                        BUF1:   .BLKB   RCBFSZ
                        BUF2:   .BLKB   RCBFSZ
                        BUF3:   .BLKB   RCBFSZ
                        BUF4:   .BLKB   RCBFSZ
                        BUF5:   .BLKB   RCBFSZ
                        BUF6:   .BLKB   RCBFSZ
                        BUF7:   .BLKB   RCBFSZ

                        ;-----------------------------------------------------------------------

                                ENDMOD
```

```
                                        .TITLE PARAMETER CODING
                                        .SBTTL  HARDWARE PARAMETER CODING SECTION
041340                                          BGNMOD

                                        ;++
                                        ; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
                                        ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
                                        ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
                                        ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
                                        ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
                                        ; WITH THE OPERATOR.
                                        ;--

041340                                          BGNHRD
(3)  041340  000021                                                     .WORD L10025-L$H
(3)  041342                                                     L$HARD::
041342                                          GPRMA   MSG1,0,0,160000,177777,YES
(4)  041342  000031                                                     .WORD   T$CODE
(4)  041344  041404                                                     .WORD   MSG1
(4)  041346  160000                                                     .WORD   T$LOLIM
(4)  041350  177777                                                     .WORD   T$HILIM
041352                                          GPRMA   MSG1B,2,0,0,776,YES
(4)  041352  001031                                                     .WORD   T$CODE
(4)  041354  041415                                                     .WORD   MSG1B
(4)  041356  000000                                                     .WORD   T$LOLIM
(4)  041360  000776                                                     .WORD   T$HILIM
041362                                          GPRML   MSG1C,6,1,YES
(4)  041362  003130                                                     .WORD   T$CODE
(4)  041364  041432                                                     .WORD   MSG1C
(4)  041366  000001                                                     .WORD   1
041370                                          GPRML   MSG2,4,1,YES
(4)  041370  002130                                                     .WORD   T$CODE
(4)  041372  041450                                                     .WORD   MSG2
(4)  041374  000001                                                     .WORD   1
041376                                          GPRML   MSG3,4,2,YES
(4)  041376  002130                                                     .WORD   T$CODE
(4)  041400  041465                                                     .WORD   MSG3
(4)  041402  000002                                                     .WORD   2

041404                                          ENDHRD
(2)
(3)  041404                                                     .EVEN
                                                        L10025:

041404  052524  034065  041440   MSG1:   .ASCIZ  /TU58 CSR/
041415     126  041505  047524   MSG1B:  .ASCIZ  /VECTOR ADDR./
041432  042120  020124  047111   MSG1C:  .ASCIZ  /PDT INTERFACE/
041450  042524  052123  042040   MSG2:   .ASCIZ  /TEST DRIVE 0/
041465     124  051505  020124   MSG3:   .ASCIZ  /TEST DRIVE 1/
                                        .EVEN
```

```
                                        .SBTTL  SOFTWARE PARAMETER CODING SECTION
                                        ;++
                                        ; THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
                                        ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
                                        ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
                                        ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
                                        ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
                                        ; WITH THE OPERATOR.
                                        ;--

                                                BGNSFT

                                                                                L$SOFT::        .WORD L10026-L$S
                                                GPRMD   MSG4,0,D,1777,8.,512.,YES
                                                                                                .WORD   T$CODE
                                                                                                .WORD   MSG4
                                                                                                .WORD   1777
                                                                                                .WORD   T$LOLIM
                                                                                                .WORD   T$HILIM
                                                GPRML   MSG4B,10,1,YES
                                                                                                .WORD   T$CODE
                                                                                                .WORD   MSG4B
                                                                                                .WORD   1
                                                GPRML   MSG5,2,1,YES
                                                                                                .WORD   T$CODE
                                                                                                .WORD   MSG5
                                                                                                .WORD   1
                                                GPRML   MSG6,6,1,YES
                                                                                                .WORD   T$CODE
                                                                                                .WORD   MSG6
                                                                                                .WORD   1
                                                GPRML   MSG7,4,1,YES
                                                                                                .WORD   T$CODE
                                                                                                .WORD   MSG7
                                                                                                .WORD   1
                                                GPRMD   MSG8,10.,D,377,1,254.,YES
                                                                                                .WORD   T$CODE
                                                                                                .WORD   MSG8
                                                                                                .WORD   377
                                                                                                .WORD   T$LOLIM
                                                                                                .WORD   T$HILIM
                                                GPRML   MSG9,12.,1,YES
                                                                                                .WORD   T$CODE
                                                                                                .WORD   MSG9
                                                                                                .WORD   1
                                        SFTOUT: ENDSFT
                                                                                                .EVEN
                                                                                L10026:
052516 041115 051105    MSG4:   .ASCIZ  'NUMBER OF BLOCKS:TEST 5-8 (8 TO 512)'
   101 042104 042040    MSG4B:  .ASCIZ  /ADD DR # TO DATA PATTERN:TEST 5-8/
   123 040524 044524    MSG5:   .ASCIZ  /STATISTICS PRINTED AT EOP/
   103 046517 040520    MSG6:   .ASCIZ  /COMPARE DATA ON READ/
051120 047111 020124    MSG7:   .ASCIZ  /PRINT PACKET ON ERROR/
020043 051105 047522    MSG8:   .ASCIZ  /# ERRORS = DVC FATAL IF 'EVL'SET/
   120 044522 052116    MSG9:   .ASCIZ  /PRINT UNIT PROTOCOL SUMMARY (TEST 9)/
                                        .EVEN
```

```
7431            000016                          .REPT   14.             ;LASTAD CORRECTION
7432                                            .WORD
7433                                            .ENDR
7440  042144                                    LASTAD
  (2)
  (2)  042144  042164                                                   .EVEN
  (2)  042146  000006                                                   .WORD TSFREE
  (3)  042150                                                           .WORD TSSIZE
                                        L$LAST::
7441  042150                                    ENDMOD
7442
7443  042150                                    BGNSETUP        1
7444  042150                                    BGNPTAB
  (4)  042150  000000                                                   .WORD   0
  (3)  042152  000004                                                   .WORD   L10031-.
  (3)  042154                                                   L10027:
7445  042154  176500                           176500
7446  042156  000300                           300
7447  042160  000003                           3
7448  042162  000000                           0
7449  042164                                    ENDPTAB
  (3)  042164                                                   L10031:
7450  042164                                    ENDSETUP
7451            000001                .END
```

```
ABNDX = 000004 G    BUF7   040302      C$GMAN= 000043      ESABO = 177720 G    GTAGIN  007464
ABO     013146      CARLF  015144      C$GPHR= 000042      ESCKS = 177757 G    GTBYTE  010346 G
ABOMSG  017254      CHECK  016274      C$GPLO= 000030      ESCKSM= 177757      GTDOWN  010104
ABONM   006532      CHKANR 010702      C$GPRI= 000040      ESCMD = 177720 G    GTOK    007730
ADR   = 000020 G    CHKANS 010576 G    C$INIT= 000011      ESNCRT= 177767 G    GTPKS1  007220 G
ALLGON  003340 G    CHKEND 011302 G    C$INLP= 000020      ESNOMO= 177737 G    GTPKS8  007430 G
ASSEMB= 000010      CHKERR 011406      C$MANI= 000050      ESNONX= 177770 G    GTPTR   010344
BDATA = 000134 G    CHKPKS 010706 G    C$MEM = 000031      ESOK  = 000000 G    GTUM    007672
BDBYTS  014726      CHKPTR 010704      C$MSG = 000023      ESPART= 177776 G    GSCNTO= 000200
BDCHK = 000022 G    CHKREE 011460      C$OPEN= 000034      ESRD  = 177757      GSDELM= 000377
BDCOM = 000014 G    CHKRET 012024      C$PNTB= 000014      ESREC = 177711 G    GSDISP= 000003
BIT0  = 000001 G    CHKSUC 012320 G    C$PNTF= 000017      ESSK  = 177740 G    GSEXCP= 000400
BIT00 = 000001 G    CHKSUM 013654 G    C$PNTS= 000016      ESSLF = 177777 G    GSHILI= 000002
BIT01 = 000002 G    CHK8   010616      C$PNTX= 000015      ESTRY = 000001 G    GSLOL!= 000001
BIT02 = 000004 G    CKCKSM 013750 G    C$QIO = 000377      ESWLOC= 177765 G    GSNO  = 000000
BIT03 = 000010 G    CLRALL 005710 G    C$RDBU= 000007      ESWR  = 177757      GSOFFS= 000400
BIT04 = 000020 G    CLRBUF 005750 G    C$REFG= 000047      EVL   = 000004 G    GSOFSI= 000376
BIT05 = 000040 G    CLRPTR 006002      C$RESE= 000033      EVLTHR  002222      GSPRMA= 000001
BIT06 = 000100 G    CMDSNT= 000100 G   C$REVI= 000003      EXOFF   007427      GSPRMD= 000002
BIT07 = 000200 G    CMNDER= 000040 G   C$RFLA= 000021      EXON    007426      GSPRML= 000000
BIT08 = 000400 G    CMPDAT  002216     C$RPT = 000025      ESEND = 002100      GSRADA= 000140
BIT09 = 001000 G    CNINIT= 000032 G   C$SEFG= 000046      ESLOAD= 000035      GSRADB= 000000
BIT1  = 000002 G    COMPAR  014550 G   C$SPRI= 000041      FLGLOC  016752 G    GSRADD= 000040
BIT10 = 002000 G    CSNRDY  003344 G   C$SVEC= 000037      FM      015730      GSRADL= 000120
BIT11 = 004000 G    CSRCVB  003346 G   C$TPRI= 000013      FMO     015712      GSRADO= 000020
BIT12 = 010000 G    C$AU  = 000052     DESC    014730      FTLNM   003320      GSXFER= 000004
BIT13 = 020000 G    C$AUTO= 000061     DEVPTR  003312 G    F$AU  = 000015      GSYES = 000010
BIT14 = 040000 G    C$BRK = 000022     DEV0    003370      F$AUTO= 000020      HARDR = 000136 G
BIT15 = 100000 G    C$BSEG= 000004     DEV1    003602      F$BGN = 000040      HARDW = 000140 G
BIT2  = 000004 G    C$BSUB= 000002     DEV2    004014      F$CLEA= 000007      HELP  = 000000
BIT3  = 000010 G    C$CEFG= 000045     DEV3    004226      F$DU  = 000016      HOE   = 100000 G
BIT4  = 000020 G    C$CLCK= 000062     DEV4    004440      F$END = 000041      HRD     013126
BIT5  = 000040 G    C$CLEA= 000012     DEV5    004652      F$HARD= 000004      HRDRD = 000016 G
BIT6  = 000100 G    C$CLOS= 000035     DEV6    005064      F$HW  = 000013      HRDWR = 000020 G
BIT7  = 000200 G    C$CLP1= 000006     DEV7    005276      F$INIT= 000006      HRD1    011746
BIT8  = 000400 G    C$CVEC= 000036     DFPTBL  002176 G    F$JMP = 000050      IBE   = 010000 G
BIT9  = 001000 G    C$DCLN= 000044     DFTL1   012774      F$MOD = 000000      IDPTR   003324 G
BLKEND= 000202 G    C$DODU= 000051     DIAGMC= 000000      F$MSG = 000011      IDU   = 000040 G
BLKER   003332 G    C$DRPT= 000024     DLV   = 000074 G    F$PROT= 000021      IER   = 020000 G
BLKSIZ= 000212 G    C$DU  = 000053     DOBRK   014010 G    F$PWR = 000017      INIT    016164
BLKTBL  003350 G    C$EDIT= 000003     DONE    003322 G    F$RPT = 000012      INITWD  014542
BOE   = 000400 G    C$ERDF= 000055     DR    = 000060 G    F$SEG = 000003      INIT2   016212
BRKPTR  014546      C$ERHR= 000056     DRVCHK  002220      F$SOFT= 000005      ISR   = 000100 G
BRKTO   014544      C$ERRO= 000060     EF.CON= 000036 G    F$SRV = 000010      IXE   = 004000 G
BRKWD   014540      C$ERSF= 000054     EF.NEW= 000035 G    F$SUB = 000002      I$AU  = 000041
BUFTBL  027700      C$ERSO= 000057     EF.PWR= 000034 G    F$SW  = 000014      I$AUTO= 000041
BUF0    030760      C$ESCA= 000010     EF.RES= 000037 G    F$TEST= 000001      I$CLN = 000041
BUF1    032016      C$ESEG= 000005     EF.STA= 000040 G    GBTMP   010572      I$DU  = 000041
BUF2    033054      C$ESUB= 000003     ENDGPB  010146      GBTMP2  010574      I$HRD = 000041
BUF3    034112      C$ETST= 000001     ENDRSP  007100      GETANS  007152 G    I$INIT= 000041
BUF4    035150      C$EXIT= 000032     ENDT9   027206      GETHRD  016316      I$MOD = 000041
BUF5    036206      C$GETB= 000026     ERRDES  013170 G    GETPTR  007216      I$MSG = 000041
BUF6    037244      C$GETW= 000027     ERRMOD  010150      GETR5   017222      I$PROT= 000040
```

```
I$PTAB= 000041          L$HPTP  002022 G      MRSP   = 000210 G      NXTST2  006352          RETERR  012246
I$PWR  = 000041          L$HW    002176 G      MSAGE1  027212          ONEFIL= 000001          RETRY  = 000002 G
I$RPT  = 000041          L$ICP   002104 G      MSAGE2  027255          OTL    = 000010 G       RLUN    015574
I$SEG  = 000041          L$INIT  016164 G      MSAGE3  027307          OVRFLO  013534          RPTR    015576
I$SETU= 000041          L$LADP  002026 G      MSAGE4  027336          OVRN   = 000012 G       RSCMND= 000002 G
I$SFT  = 000041          L$LAST  042150 G      MSAUTO  017116          O$APTS= 000000          RSCONT= 000020 G
I$SRV  = 000041          L$LOAD  002100 G      MSBDA   002340 G        O$AU   = 000001          RSDASZ= 000204 G
I$SUB  = 000041          L$LUN   002074 G      MSCMD   002704 G        O$BGNR= 000001          RSDATA= 000001 G
I$TST  = 000041          L$MREV  002050 G      MSCOM   002404 G        O$BGNS= 000001          RSDNSZ= 000222 G
J$JMP  = 000167          L$NAME  002000 G      MSG1    041404          O$DU   = 000001          RSEND  = 000003 G
LENGTH  002210           L$PRIO  002042 G      MSG1B   041415          O$ERRT= 000000          RSGCDP= 000034 G
LGOFST= 000120 G         L$PROT  002142 G      MSG1C   041432          O$GNSW= 000001          RSINIT= 000004 G
LNCNT   015130           L$PRT   002112 G      MSG2    041450          O$POIN= 000001          RSMSIZ= 000012 G
LOE    = 040000 G        L$REPP  002062 G      MSG3    041465          O$SETU= 000001          RSNDSZ= 000016 G
LOG     012634 G         L$REV   002010 G      MSG4    041566          PARTL  = 000034 G       RSNTAB  002226
LOGO    013156           L$RPT   015150 G      MSG4B   041633          PATTEN= 000072 G        RSSEND= 000100 G
LOGOK   012722           L$SOFT  041504 G      MSG5    041675          PDTFLG  016750 G        RSSGET= 000012 G
LOGOK2  013006           L$SPC   002056 G      MSG6    041727          PERDEV  006364          RSSNIT= 000001 G
LOGO    012710           L$SPCP  002020 G      MSG7    041754          PKPTR  = 000104 G       RSSNOP= 000000 G
LOG1    013024           L$SPTP  002024 G      MSG8    042002          PNT    = 001000 G       RSSNSZ= 000016 G
LOG2    013054           L$STA   002030 G      MSG9    042043          PPSOT9  002224          RSSRD  = 000002 G
LOG3    013114           L$SW    002210 G      MSHCHK  002556 G        PRBUF   002214          RSSSEK= 000005 G
LOG3B   013136           L$TEST  002114 G      MSHDRD  003154 G        PRDAT   015132          RSSSLF= 000007 G
LOT    = 000010 G        L$TIML  002014 G      MSHDWR  003216 G        PRFORM  015134          RSSWR  = 000003 G
LSTDEV  003366 G         L$UNIT  002012 G      MSNIT   002620 G        PRI    = 002000 G       RSVP    006560 G
L$ACP   002110 G         L10001  002206        MSNLOG  002322 G        PRI00  = 000000 G       RSXOFF= 000023 G
L$APT   002036 G         L10002  002226        MSNOMO  002446 G        PRI01  = 000040 G       RSXON  = 000020 G
L$AU    017302 G         L10003  013360        MSNOTP  002464 G        PRI02  = 000100 G       RTRYN   012204
L$AUT   002070 G         L10004  014446        MSNRSP  002764 G        PRI03  = 000140 G       RUN     006052 G
L$AUTO  016754 G         L10005  014502        MSOVRN  003260 G        PRI04  = 000200 G       SAVCNT= 000206 G
L$CCP   002106 G         L10006  015572        MSPART  002634 G        PRI05  = 000240 G       SECREC  003334 G
L$CLEA  017136 G         L10007  016662        MSQRSP  003000 G        PRI06  = 000300 G       SERVST  010342
L$CO    002032 G         L10010  017060        MSREC   002720 G        PRI07  = 000340 G       SETDR   005632 G
L$DEPO  002011 G         L10011  017154        MSRNIT  002536 G        PRNPAK  014764 G        SETLEN  016624
L$DESC  002122 G         L10012  017220        MSSELF  002364 G        PRNSIZ  003336 G        SETPTR  005706
L$DESP  002076 G         L10013  017302        MSSFRD  003054 G        PTR     017252          SETSRV  010256
L$DEVP  002060 G         L10014  017504        MSSFWR  003114 G        RCBCNT  003316          SETUP   006004 G
L$DISP  002152 G         L10015  017756        MSSKER  002306 G        RCBFSZ= 001036 G        SFPTBL  002210 G
L$DLY   002116 G         L10016  021350        MSTOSN  003032 G        RCDB   = 000024 G       SFT     013104
L$DTP   002040 G         L10017  022754        MSUNIT  002656 G        RCFLG   003314 G        SFTOUT  041566
L$DTYP  002034 G         L10020  023744        MSWPRO  002514 G        RCINIT= 000006 G        SFTRD  = 000002 G
L$DU    017156 G         L10021  024530        MSWRSP  002740 G        RCSR   = 000022 G       SFTWR  = 000004 G
L$DUT   002072 G         L10022  025520        MXRTRY  003330 G        RCVBUF= 000102 G        SKERR  = 000024 G
L$DVTY  005510 G         L10023  026304        NCART  = 000054 G       RCVHND  014450          SLFER  = 000044 G
L$EF    002052 G         L10024  027210        NODRVS  016714          RCVINT  014450 G        SND     006650
L$ENVI  002044 G         L10025  041404        NOMOR   006534          RDN0   = 000114 G       SNDBYT  007102 G
L$ETP   002102 G         L10026  041566        NOMOT  = 000030 G       RDN1   = 000116 G       SNDCNT= 000070 G
L$EXP1  002046 G         L10027  042154        NOREE   011334          REC    = 000064 G       SNDHND  014434
L$EXP4  002064 G         L10031  042164        NOUNIT= 000036 G        RECDAT  017742          SNDINT  014434 G
L$EXP5  002066 G         MABEE   013066        NOXOFF  006632          RECERR= 000043 G        SOFTR  = 000122 G
L$HARD  041342 G         MESMRS  010200        NTSFT   013036          RECID   013442 G        SOFTW  = 000124 G
L$HIME  002120 G         MODRSP  010252        NXTRET  006530          RECID2  013616          SRVTBL  010322
L$HPCP  002016 G         MRSDLY  010254        NXTST   006102 G        RECOV   012044          STAEOP  002212
```

```
STATHD  015600
STATUS= 000000 G
STHD2   016054
STRT    016746 G
SUCCS = 000076 G
SUCOTL= 000046 G
SVCGBL= 000000
SVCINS= 000001
SVCSUB= 000001
SVCTAG= 000001
SVCTST= 000001
SWAPDR  005530 G
SWPTR   005630
SYSTAT  003306 G
S$LSYM= 010000
TAPLEN  003310 G
TEST9   003342 G
THRSHI  012152
THRSLO  012124
TMP   = 000066 G
TOMANY  016664
TORCVB= 000050 G
TOSNDB= 000056 G
TRBUF   027722
TRK   = 000062 G
TRPHND  017064
TRPPTR  017062
TSTPC = 000020 G
TSTTOP  003326
TST1    017550
TST2    017552
TST3    020024
TST3PT  022742
TST4    021416
TST5    023022
TST5EX  023736
TST6    024012
TST6EX  024522
TST7    024576
TST7EX  025512
TST8    025566
TST8EX  026276
TST9    026330
TUVECT= 000204 G
T$ARGC= 000002
T$CODE= 006130
T$ERRN= 000146
T$EXCP= 000000
T$FLAG= 000040
T$FREE= 042164
T$GMAN= 000000
T$HILI= 000376
T$LAST= 000001
T$LOLI= 000001
T$LSYM= 010000
T$LTNO= 000011
T$NEST= 177777
T$NSO = 000000
T$NS1 = 000005
T$PCNT= 000000
T$PTAB= 010030
T$PTHV= 000001
T$PTNU= 000001
T$SAVL= 177777
T$SEGL= 177777
T$SIZE= 000006
T$SUBN= 000000
T$TAGL= 177777
T$TAGN= 010032
T$TEMP= 000000
T$TEST= 000011
T$TSTM= 177777
T$TSTS= 000001
T$$AU = 010013
T$$AUT= 010010
T$$CLE= 010011
T$$DAT= 010031
T$$DU = 010012
T$$HAR= 010025
T$$HW = 010001
T$$INI= 010007
T$$MSG= 010003
T$$PC = 000001
T$$PRO= 010000
T$$PTA= 010030
T$$RPT= 010006
T$$SOF= 010026
T$$SRV= 010005
T$$SW = 010002
T$$TES= 010024
T1      017304 G
T1TRY = 000146 G
T2      017506 G
T3      017760 G
T4      021352 G
T4TRY = 000132 G
T5      022756 G
T6      023746 G
T7      024532 G
T8      025522 G
T9      026306 G
UAM   = 000200 G
UNIT    013562 G
UNITNO  027366
UNREC   012224
UNSUC   011622
UNXPCT  007634
WAIT    014504
WHCHDR  013640 G
WRLOCK= 000026 G
WRTN0 = 000110 G
WRTN1 = 000112 G
XFNSND  006640
XMDB  = 000030 G
XMSR  = 000026 G
XSCNT = 000036 G
XSFLG = 000034 G
XSPKNM= 000032 G
XSPTR = 000106 G
XSALWA= 000000
XSFALS= 000040
XSOFFS= 000400
XSTRUE= 000020
.     = 042164

. ABS.  042164      000

 ERRORS DETECTED:  0

CZTUUE.BIN/EN:AMA:ABS,CZTUUE=SVC.SML,CZTUUE.P11
RUN-TIME:  19 24  .6 SECONDS
RUN-TIME RATIO: 51/45=1.1
CORE USED:  23K  (46 PAGES)
```