RH11-RS03

RH11 RS03/LA RS04 DATA RLBTY EP DZRSC DL E

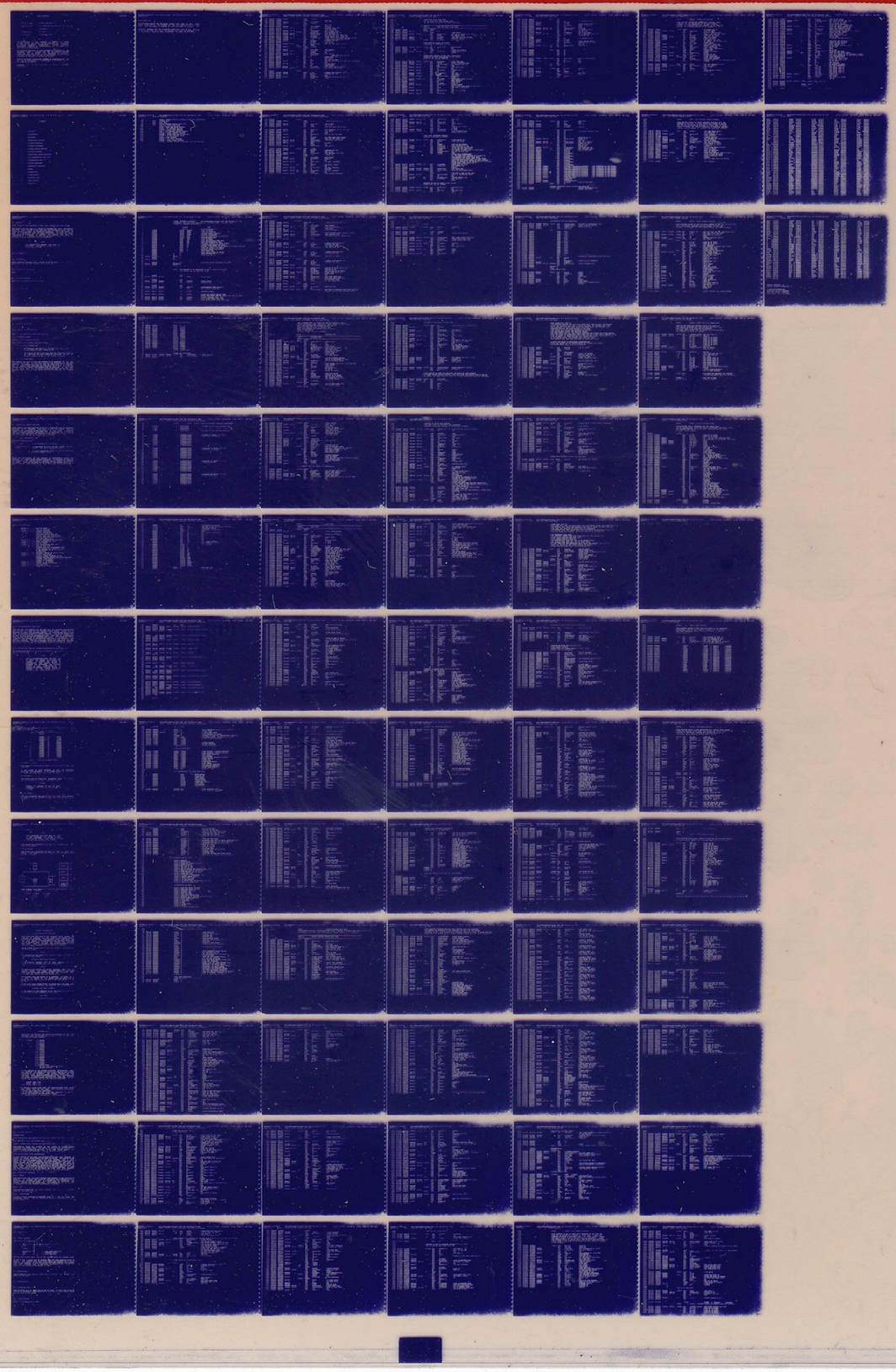
MD-11-DZRSC-E

FICHE 1 OF 1

FEB 1979

digital

MADE IN USA



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRSC-E-D

PRODUCT NAME: RH11-RS03-RS03/LA-RS04 DATA RELIABILITY DIAGNOSTIC

PROGRAM DATE: AUG 1976

MAINTAINER: DIAGNOSTIC GROUP

TABLE OF CONTENTS

CONTENTS

1. ABSTRACT

THIS DIAGNOSTIC WAS DESIGNED TO TEST RS03, RS03/LA AND RS04 DRIVES.

THE DZRSC DISK DATA TEST IS A SERIES OF ADDRESS AND DATA RELIABILITY ROUTINES WHICH VERIFY TO THE USER THAT THE CONTROLLER (RH11) AND THE DISKS (RS03/LA OR RS04) ARE OPERATING CORRECTLY. THIS TEST SHOULD BE USED IN CONJUNCTION WITH THE DZRSB DIAGNOSTIC. IF THERE IS A POWER FAIL WHILE THE DIAGNOSTIC IS RUNNING, THE PROGRAM WILL WAIT FOR APPROXIMATELY 5 MINUTES, TO GIVE ALL THE DRIVES TIME TO COME BACK UP TO SPEED, BEFORE RESTARTING THE TEST SEQUENCE.

NOTE

THIS PROGRAM WILL DESTROY ALL DATA ON THE DISKS. TURN OFF ALL DRIVES THAT YOU DO NOT WANT TO TEST.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP11 STANDARD COMPUTER WITH A MINIMUM OF 8K OF MEMORY, AND AN RH11 CONTROLLER WITH AN RS03, RS03/LA OR AN RS04 DISK.

2.2 PRELIMINARY PROGRAMS

DZRSB

3. LOADING PROCEDURE

USE STANDARD PROCEDURE FOR ABS TAPES.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5. (ALL DOWN FOR WORST CASE TESTING)

4.2 STARTING ADDRESS

PROGRAM AND/OR OPERATOR ACTION

LOAD PROGRAM INTO MEMORY USING ABS LOADER.

1. STARTING ADDRESS 200.

A. SET SWITCHES (SEE SEC 5.). ALL DOWN FOR WORST CASE (IF SWITCH-LESS CPV SIMPLY PRESS START)

B. THE DISPLAY ON THE 11/45 WILL SHOW THE ITERATION COUNT IN THE LEFT BYTE AND TEST NUMBER IN THE RIGHT. TO USE, SET THE DATA DISPLAY SWITCH TO THE DISPLAY POSITION.

C. PRESS START.

THE PROGRAM WILL NOW MAP THE DATA BUFFERS IN 4K SEGMENTS ON -A- AND -B- PORTS UP TO 28K. IT WILL THEN TYPE OUT THE PARAMETERS OF THE DATA BUFFERS. THE PROGRAM WILL ONLY DO THIS THE FIRST TIME IT IS STARTED, FOR IT STORES THESE ADDRESSES AND CONTINUES USING THEM. TO HAVE THE PROGRAM REMAP THE SYSTEM, THE PROGRAM MUST BE RELOADED. IF YOU WISH TO GET ABOVE 28K, YOU CAN ENTER CONVERSATION MODE AND PUT THE DATA BUFFERS WHERE YOU WISH. THE SIZE OF THE DATA BUFFERS CAN NOT EXCEED 24K.

5. OPERATIONAL SWITCH SETTINGS

THIS PROGRAM HAS BEEN MODIFIED TO RUN ON A PROCESSOR WITH OR WITHOUT A HARDWARE SWITCH REGISTER. WHEN FIRST EXECUTED THE PROGRAM TESTS THE EXISTENCE OF A HARDWARE SWITCH REGISTER. IF NOT FOUND A SOFTWARE SWITCH REGISTER LOCATION (SWREG=LOC.176) IS DEFAULTED TO. IF THIS IS THE CASE, UPON EXECUTION THE CONTENTS OF THE SWREG ARE DUMPED IN OCTAL ON THE CONSOLE TTY AND ANY CHANGES ARE REQUESTED.

(I.E) SWR=XXXXXX NEW=

POSSIBLE RESPONSES ARE:

1. <CR> IF NO CHANGES ARE TO BE MADE.

2. 6 DIGITS 0-7 TO REPRESENT IN OCTAL THE NEW SWITCH REGISTER VALUE: LAST DIGIT FOLLOWED BY <CR>.

3. ^U TO ALLOW REENTERING VALUE IF ERROR IS COMMITTED KEYING IN SWREG VALUE.

BUILT INTO THE PROGRAM IS THE ABILITY TO DYNAMICALLY CHANGE THE CONTENTS OF SWREG DURING PROGRAM EXECUTION. BY STRIKING ^G (CNTL G) ON CONSOLE TTY THE OPERATOR SETS A REQUEST FLAG TO CHANGE THE CONTENTS OF SWREG, WHICH IS PROCESSED IN KEY AREAS OF THE PROGRAM CODE (I.E.) EPROR ROUTINES, AFTER HALTS END OF PASS, AND OTHER APPLICABLE AREAS.

SWITCH SETTINGS ARE:

```
          SW<15> = 1 ..... HALT ON ERROR
          SW<14> = 1 ..... LOOP ON FUNCTION
          SW<13> = 1 ..... INHIBIT PRINTOUT
          SW<12> = 1 ..... INHIBIT COMPARISON
                           WITH THIS SWITCH SET, THE
                           PROGRAM WILL NOT COMPARE THE
                           DATA IT READ FROM THE DISK WITH
                           THE KNOWN GOOD DATA.
          SW<11> = 1 ..... HALT ON COMPLETION OF TRANSFER
          SW<10> = 1 ..... ENTER CONVERSATION MODE
          SW<09> = 1 ..... LOOP ON ERROR
          SW<08> = 1 ..... DATA RELIABILITY TEST MODE
          SW<07> = 1 ..... WAIT IN WAIT MODE
                           PROGRAM RUNS IN A BACKGROUND TEST
                           WHILE WAITING FOR INTERRUPT, WITH
                           SW SET PROGRAM WAITS IN A WAIT
                           INSTRUCTION.
          SW<06> = 1 ..... OPTIONAL TYPEOUT OF RETRY ERRORS
          SW<05> = 1 ..... INHIBIT PASS COUNT
          SW<04> = 1 ..... ALLOWS 0 ERROR TYPEOUTS IN THE
                           COMPARE ROUTINE BEFORE EXECUTING NEXT READ
                           COMMAND. WHEN SWITCH IS 0, ONLY 1 ERROR
                           TYPEOUT IS RECORDED.
          SW<03> = 1 ..... TYPEOUT 0  OF ERRORS
          SW<02> = 1 ..... INHIBIT MEMORY MANAGEMENT
          SW<01> = 1 ..... DATA TEST ONLY
          SW<00> = 1 ..... DROPS DRIVE AFTER 20 ERRORS
```

F1

5.1 DATA RELIABILITY TEST MODE

WITH SW8 SET, THE PROGRAM WILL SET THE "BAI" BIT IN RHCS2 AND TRANSFER 64K OF DATA AT A TIME FOR ALL PATTERNS EXCEPT RANDOM. RANDOM WILL BE EXECUTED AS USUAL WITH STANDARD BUFFERS. NO COMPARES ARE DONE IN THIS MODE OF OPERATION EXCEPT ON RANDOM PATTERNS. THIS OPTION SHOULD ONLY BE USED IN DATA TEST OR CONVERSATION MODE. WHEN USED IN CONVERSATION MODE IT OVER RIDES THE NON STANDARD WORD COUNT. YOU SHOULD NOT SELECT A DESIRED DISK ADDRESS IN CONVERSATION MODE FOR IT CAN PRODUCE A DISK ADDRESS OVERFLOW ERROR FOR THIS DATA RELIABILITY TEST MODE ONLY DOES 64K WORD TRANSFERS. IF SW8 IS CHANGED, WHILE THE PROGRAM IS RUNNING, THE PROGRAM WILL FINISH ITS PASS BEFORE EXECUTING THE SWITCH CHANGE.

5.2 CONVERSATION MODE FOR PROGRAM PARAMETERS FOR DATA TEST ONLY

IN CONVERSATION MODE THE OPERATOR CAN SPECIFY ANY ONE OR ALL OF THE PROGRAM PARAMETERS.

NOTE

ONCE IN CONVERSATION MODE, THE ONLY WAY TO REMAP THE SYSTEM IS TO RELOAD THE PROGRAM. TO RESTART THE PROGRAM IN CONVERSATION MODE WITHOUT HAVING TO REANSWER THE QUESTIONS, THE STARTING ADDRESS IS 210. RESET SWITCH 10. TO RESTART THE PROGRAM WITHOUT HAVING TO REANSWER THE PORT SIZING QUESTIONS, RESTART AT 220. RESET SWITCH 10.

THE PROGRAM WILL NOW ASK SEVERAL QUESTIONS, THE TABLE BELOW WILL HELP YOU ANSWER THE QUESTIONS.

TYPE TO START AT		TYPE TO START AT	
0	000000		
1	020000	20	400000
2	040000	21	420000
3	060000	22	440000
4	100000	23	460000
5	120000	24	500000
6	140000	25	520000
7	160000	26	540000
10	200000	27	560000
11	220000	30	600000
12	240000	31	620000
13	260000	32	640000
14	300000	33	660000
15	320000	34	700000
16	340000	35	720000
17	360000	36	740000

NOTE: TYPE ONLY NUMBERS SHOWN!!!

1. -A- PORT? (Y OR N)

   THIS GIVES YOU THE OPTION TO TEST -A- OR -B- PORT USING THE DATA TEST.

   IF THE ANSWER TO THIS QUESTION IS YES, THE FOLLOWING QUESTIONS WILL BE ASKED. IF THE ANSWER IS NO, -B- PORT WILL BE TESTED AND QUESTIONS 4 AND 5 WILL BE ASKED.

2. 1ST 4K BANK #

   THIS NUMBER THAT IS TYPED WILL DETERMINE WHERE THE BUFFER AREA WILL START ON -A- PORT. USE TABLE ABOVE

   NOTE:

   PROGRAM IS LOCATED IN 1ST 4K BANK. THEREFORE, THIS BANK CAN NOT BE USED AS A BUFFER.

3. # OF 4K BANKS ?

   THIS WILL DETERMINE THE SIZE OF THE -A- PORT DATA BUFFER. THE SIZE OF THE DATA BUFFER CAN NOT EXCEED 24K AND MUST BE IN "OCTAL".

4. 1ST 4K BANK #

NOTE

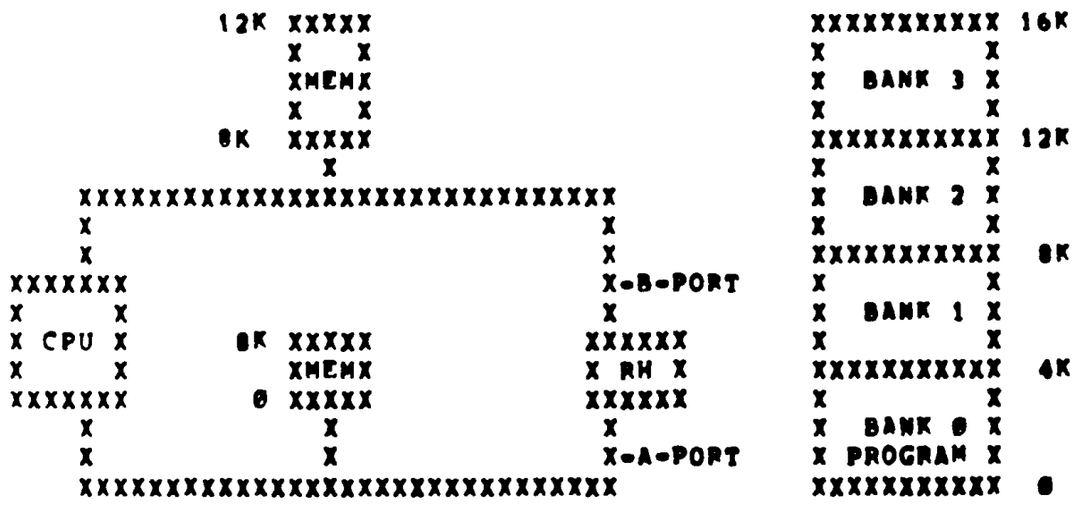
THIS DIAGNOSTIC WILL ONLY TEST -B- PORT IF THE PROCESSOR HAS ACCESS TO THAT MEMORY ON -B- PORT. THIS MEMORY MUST HAVE THE SAME ADDRESS ON ALL PORTS.

THIS NUMBER WILL DETERMINE WHERE THE DATA BUFFER AREA WILL START ON -B- PORT.

5. # OF 4K BANKS?

THIS NUMBER WILL DETERMINE THE SIZE OF THE -B- PORT DATA BUFFER. THE SIZE OF THE DATA BUFFER CAN NOT EXCEED 24K AND MUST BE IN "OCTAL".

EXAMPLE:



THESE ANSWERS GIVEN BELOW WILL TEST THE CONFIGURATION IN THE GIVEN EXAMPLE. ANSWERS:

TO TEST	-A- PORT	-B- PORT
	1) Y	1) N
	2) 1	4) 2
	3) 1	5) 1

PROGRAM CONVERSATION

MULTI DRIVE MODE? (YES-NO)

MULTI DISK MODE IS A MODE IN THE PROGRAM WHICH ALLOWS THE OPERATOR TO EXERCISE ALL THE DISKS ON THE SYSTEM WITHOUT RE-STARTING THE PROGRAM. THE PROGRAM, AFTER EXERCISING ONE DISK WILL REPORT A MESSAGE TELLING THE OPERATOR WHICH DISK WILL BE SELECTED NEXT, AND THEN THE PROGRAM WILL EXERCISE THAT DISK. WHEN A COMPLETE PASS IS ACCOMPLISHED, A PASS COMPLETE WILL BE REPORTED AND THE TEST WILL RECYCLE.

IF THE ANSWER TO THE MULTI DRIVE MODE WAS "NO", THE FOLLOWING QUESTION IS ASKED.

UNIT #

THE OPERATOR CAN NOW SELECT THE UNIT HE WISHES TO TEST BY TYPING THE UNIT NUMBER.

OPTIONAL WORD COUNT (YES-NO)

IF THE OPERATOR ANSWERS "NO" TO THIS QUESTION THE NEXT QUESTION WILL BE DELETED FROM THE CONVERSATION.

WD CT

THE OPERATOR CAN SPECIFY ANY LENGTH TRANSFER FROM 1(8) TO 60000(8) WORDS. THE NORMAL TRANSFER LENGTH 'S N(8) WORDS WHERE N IS THE MAXIMUM BUFFER SIZE FOR THE AVAILABLE CORE. IN EITHER CASE, BUFFER WILL NOT EXCEED 24 K.

THIS PROGRAM MAPS THE SYSTEM IN 4K SEGMENTS. IF THERE IS A 1K BLOCK OF MEMORY ON THE SYSTEM THAT YOU WOULD LIKE TO REACH, YOU CAN TYPE IN THAT 4K BANK # AND THEN SPECIFY A WC OF 2000.

IF THE WORD COUNT NUMBER TYPED, IS LARGER THAN THE CORE SIZE GIVEN IN THE SETUP ROUTINE, THE QUESTION WILL BE REPEATED.

OPTIONAL DSK ADDR (YES-NO)

IF THE ANSWER TO THIS QUESTION IS NO, THE WHOLE DISK WILL BE WRITTEN AND THE NEXT QUESTION IS NOT ASKED.

DSK ADDR

THE OPERATOR CAN NOW SPECIFY THE STARTING SECTOR

PATTERN NO.?

THIS GIVES THE OPERATOR THE OPTION OF SELECTING ALL THE DATA PATTERNS (#22) OR ANY ONE DATA PATTERN, SIMPLY BY TYPING THE DATA PATTERN NUMBER DESIRED.

PATTERN	0	= 000000
=	1	= 177777
=	2	= 031463
=	3	= 066666
=	4	= 100001
=	5	= 107070
=	6	= 070707
=	7	= 052525
=	10	= 125252
=	11	= 177737
=	12	= 146314
=	13	= 136363
=	14	= 063636
=	15	= 000001
=	16	= 100005
=	17	= 155555
=	20	= 133333
=	21	= RANDOM DATA
=	22	= RUN ALL DATA PATTERNS UNDER

PROGRAM CONTROL

IN THIS SECTION OF THE PROGRAM PARAMETER CONVERSATION MODE, THE OPERATOR CAN SELECT ANY ONE OR ALL THREE OF THE CONTROL FUNCTIONS TO BE EXECUTED. THE NORMAL SEQUENCE OF DISK FUNCTIONS UNDER PROGRAM CONTROL ARE WRITE, WRITE CHECK, AND THEN READ. BY ENTERING THE CONVERSATION MODE THE OPERATOR HAS GAINED COMPLETE CONTROL OVER THE DISK FUNCTIONS. HE MUST SPECIFY YES OR NO TO ALL OF THE FOLLOWING QUESTIONS.

WRITE? (YES - NO)
READ? (YES - NO)
WRITE CHECK? (YES - NO)

TO PERFORM A WRITE CHECK ONLY, THE OPERATOR MUST FIRST WRITE SOME KNOWN DATA ON THE DISK. THIS COURSE OF ACTION ALSO PREVAILS FOR A READ ONLY OPERATION.
* IF AN ERROR OCCURS IN THE LINE THE OPERATOR IS TYPING, DEPRESS THE RUB-OUT KEY AND RETYPE ANSWER.
ALL ANSWERS SHOULD BE FOLLOWED BY A CARRIAGE-RETURN

5.3 ROUTINE ABSTRACTS

ADDRESS TEST

THIS TEST WRITES EACH SECTOR WITH ITS OWN ADDRESS THEN READS IT BACK

AND COMPARES IT FOR THE CORRECT DATA.

RANEX - RANDOM DATA, ADDRESS AND WORD COUNT TEST

THIS ROUTINE TESTS THE ABILITY OF THE SYSTEM TO ACCESS RANDOM ADDRESSES WITH RANDOM DATA. ONE SECTOR OF RANDOM DATA IS WRITTEN AT A STARTING RANDOM ADDRESS ON THE DISK. IT IS THEN WRITE CHECKED AND READ. ALL ERRORS ARE REPORTED. THIS IS REPEATED 1000 TIMES.

DATA RELIABILITY - DATA PATTERN TEST

IN THIS PORTION OF THE TEST, THE RELIABILITY OF THE DISK SURFACE IS TESTED BY WRITE, WRITE CHECK, AND READ FUNCTIONS. THE ROUTINE FIRST WRITES THE COMPLETE SURFACE WITH A SET DATA PATTERN, THEN A WRITE CHECK OF THE COMPLETE SURFACE IS ACCOMPLISHED, THUS REPORTING ALL ERRORS BETWEEN THE DATA WRITTEN AND THE DATA IN MEMORY. THE DISK IS THEN READ. THE DATA READ FROM THE DISK IS COMPARED AGAINST THE KNOWN DATA PATTERN. THIS COMPARE IS TAKING PLACE THE SAME TIME THE DISK IS BEING READ. THE BUFFER IS CLEARED AS IT IS BEING COMPARED. IF THERE ARE DATA BUFFERS ON -A- AND -B- PORTS, THE DATA TEST WILL TRANSFER DATA OVER -A- PORT ON ODD PASSES AND OVER -B- PORT ON EVEN PASSES.

5.4 SUBROUTINE ABSTRACTS

5.4.1 SCOPE

THIS SUBROUTINE CALL IS PLACED BETWEEN EACH SUBTEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH SUBTEST AS IT IS BEING ENTERED IN LOCATION "LAD". IF A SCOPE LOOP IS REQUESTED, THE CURRENT SUBTEST WILL BE LOOPED UPON. THE CONTENTS OF LAD MAY BE USED TO DETERMINE THE LAST SUBTEST SUCCESSFULLY COMPLETED.

5.4.2 HLT

THIS ROUTINE PRINTS OUT AN ERROR MESSAGE (SEE 6.1). TO INHIBIT TYPEOUTS, PUT SW<13> ON A 1.

5.4.3 TRAPCATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM 0 - 776 TO CATCH ANY UNEXPECTED TRAPS. THUS ANY UNEXPECTED TRAPS OR INTERRUPTS WILL HALT AT THE VECTOR + 2.

6. ERRORS

6.1 ERROR PRINTOUT

THE FORMAT IS AS FOLLOWS;

```
ADR     CS1 = ----- CS2 = ----- ER = -----
GOOD        = ----- BAD = -----
```

WHERE;

```
CS1,CS2,ER ETC.         = RS11 DISK REGISTERS.
GOOD                    = EXPECTED DATA.
BAD                     = DATA RECEIVED.
```

TO FIND THE FAILING TEST, LOOK AT THE LISTING ABOVE THE ADDRESS TYPED.

IF SW0 IS SET, A DRIVE WILL BE DROPPED FROM THE TEST SEQUENCE AFTER 20 ERRORS. THE PROGRAM WILL STATE WHICH DRIVE WAS DROPPED AND ON WHICH PASS IT WAS DROPPED. IF ALL THE DRIVES HAVE BEEN DROPPED, THE PROGRAM WILL TYPE "TESTING UNIT 0" AND HALT", INDICATING THAT IT COULD NOT FIND ANY MORE DRIVES ON THE SYSTEM TO TEST.

7. RESTRICTIONS

THIS DIAGNOSTIC WILL TEST -B- PORT, ONLY IF THE CPU CAN ACCESS THAT MEMORY ON -B- PORT.

8. MISCELLANEOUS

8.1 EXECUTION TIME

PASS COMPLETE WILL BE TYPED OUT AT END OF PASS. IT WILL TAKE 10 TO 20 MINUTES TO COMPLETE A PASS DEPENDING ON THE TYPE OF DRIVE BEING TESTED AND THE SIZE OF THE SYSTEM.

8.2 STACK POINTER

STACK IS INITIALLY SET TO 500

8.3 POWER FAIL

THE STARTING ADDRESS FOR THE WRITE POWER FAIL TEST IS 244. WHEN ASKED, ENTER UNIT 0. THE PROGRAM WILL TELL THE OPERATOR WHEN TO POWER DOWN. WHEN THE SYSTEM IS POWERED UP, ONLY ONE ERROR IS ALLOWED. THE

STARTING ADDRESS FOR THE WRITECHECK POWER FAIL TEST IS 250. HERE AS IN THE WRITE POWER FAIL TEST, THE PROGRAM WILL TELL THE OPERATOR WHEN TO POWER DOWN. WHEN THE POWER COMES BACK, NO ERRORS SHOULD OCCUR.

```
                                        .TITLE  MAINDEC-11-DIRSC-E      RH11-RS03-RS03/LA-RS04 DATA AND RELIABILITY TEST
                                        ;COPYRIGHT 1973,1974,1975,1976 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
                                        ;PROGRAM BY STANLEY HARACKIEWICZ

                                        ;          SWITCH                        USE
                                        ;          ------                ------------------
                    100000                      SW15=   100000          ;HALT ON ERROR
                    040000                      SW14=   40000           ;LOOP ON FUNCTION
                    020000                      SW13=   20000           ;INHIBIT ERROR TYPEOUTS
                    010000                      SW12=   10000           ;INHIBIT COMPARISON
                    004000                      SW11=   4000            ;HALT ON COMPLETION OF TRANSFER
                    002000                      SW10=   2000            ;CONVERSATION MODE
                    001000                      SW9=    1000            ;LOOP ON ERROR
                    000400                      SW8=    400             ;DATA RELIABILITY TEST MODE
                    000200                      SW7=    200             ;WAIT IN BACKGROUND TEST
                    000100                      SW6=    100             ;OPTIONAL TYPEOUT OF RETRY ERRORS
                    000040                      SW5=    40              ;INHIBIT PASS COUNT AND UNIT #
                    000020                      SW4=    20              ;ALLOWS 8 LOCATIONS TO BE TESTED IN COMPARE ROUTINE
                    000010                      SW3=    10              ;TYPE OUT TOTAL # OF ERRORS
                    000004                      SW2=    4               ;INHIBIT MEMORY MANAGEMENT
                    000002                      SW1=    2               ;DATA TEST ONLY
                    000001                      SW0=    1               ;DROP DRIVE AFTER 20 ERRORS
                    000000              .=      0                       ;TRAP CATCHER FROM 0 - 776
                    000046              .=      46                      ;HOOKS FOR ACT 11
 000046   014070                      $ENDAD
                    000052              .=      52
 000052   040000                      BIT14

                    000174              .=      174                     ;SOFTWARE SWITCH REGISTER LOCATION
 000174   000000                      DISPREG:0
 000176   000000                      SWREG:  0


                                        ;NOTE:  FOR PROGRAM TO BE RESTARTED AT 200 IT MUST BE RELOADED
                                        ;       INTO MEMORY DUE TO ONCE ONLY CODE.


                    000200              .=      200
 000200   000137   001230                      JMP     SETSWI          ;START TEST

                    000210              .=      210
 000210   012706   000500                      MOV     #500,SP         ;SETUP STACK
 000214   000137   003224                      JMP     @#ADTST         ;RESTART ADDR

                    000220              .=      220
 000220   012706   000500                      MOV     #500,SP         ;CONVERSATION MODE WITHOUT
 000224   000137   002374                      JMP     @#A1            ;DATA BUFFER QUESTIONS

 000230   000137   015154                      JMP     @#RLDR          ;RESTORE LOADER

 000234   000137   003316                      JMP     @#ADTL          ;TRACK AND SECTOR SELECT TEST
                                                                      ;WRITE EACH WORD ADDR ON ITSELF AND READ IT BACK
 000240   000137   005050                      JMP     @#RANEL         ;RANDOM ADDRESS, DATA TEST
 000244   000137   012466                      JMP     @#PFT1          ;DISK WRITE POWER FAIL TEST
 000250   000137   013014                      JMP     @#PFT2          ;DISK WRITE CHECK POWER FAIL TEST
```

```
                                        ;RH11 DATA PATTERNS
                                        PAT0:   0
000254  000000                          PAT1:   177777
000256  177777                          PAT2:   031463
000260  031463                          PAT3:   066666
000262  066666                          PAT4:   100001
000264  100001                          PAT5:   107070
000266  107070                          PAT6:   070707
000270  070707                          PAT7:   052525
000272  052525                          PAT10:  125252
000274  125252                          PAT11:  177737
000276  177737                          PAT12:  146314
000300  146314                          PAT13:  136363
000302  136363                          PAT14:  063636
000304  063636                          PAT15:  000001
000306  000001                          PAT16:  100005
000310  100005                          PAT17:  155555
000312  155555                          PAT20:  133333
000314  133333                          ;PAT21  RANDOM DATA

                                        ;CLEAR ALL REGISTERS
000316  012777  000040  000510  .CLRDV: MOV     #40,@#RSCS2     ;CLEAR ALL REG
000324  013777  001160  000502           MOV     UNNUM,@#RSCS2   ;GET UNIT #
000332  000002                          RTI
```

```
                                .SBTTL          $KMMR - KERNAL MEMORY MANAGEMENT REGISTER ASSIGNMENTS

            177572                      SR0=177572              ;ADDRESS OF MEM MGMT REGISTER SR0
            177574                      SR1=177574              ;    "    "       "          "   SR1
            177576                      SR2=177576              ;    "    "       "          "   SR2
            172516                      SR3=172516              ;ADDRESS OF MEM MGMT REGISTER SR3

            172300                      KIPDR0=172300           ;ADDRESS OF KERNEL 'I' PAGE
            172302                      KIPDR1=172302           ;DESCRIPTOR REGISTERS
            172304                      KIPDR2=172304
            172306                      KIPDR3=172306
            172310                      KIPDR4=172310
            172312                      KIPDR5=172312
            172314                      KIPDR6=172314
            172316                      KIPDR7=172316

            172320                      KDPDR0=172320           ;ADDRESSES OF KERNEL 'D' PAGE
            172322                      KDPDR1=172322           ;DESCRIPTOR REGISTERS
            172324                      KDPDR2=172324
            172326                      KDPDR3=172326
            172330                      KDPDR4=172330
            172332                      KDPDR5=172332
            172334                      KDPDR6=172334
            172336                      KDPDR7=172336

            172340                      KIPAR0=172340           ;ADDRESSES OF KERNEL 'I' PAGE
            172342                      KIPAR1=172342           ;ADRESS REGISTERS
            172344                      KIPAR2=172344
            172346                      KIPAR3=172346
            172350                      KIPAR4=172350
            172352                      KIPAR5=172352
            172354                      KIPAR6=172354
            172356                      KIPAR7=172356

            172360                      KDPAR0=172360           ;ADDRESSES OF KERNEL 'D' PAGE
            172362                      KDPAR1=172362           ;ADDRESS REGISTERS
            172364                      KDPAR2=172364
            172366                      KDPAR3=172366
            172370                      KDPAR4=172370
            172372                      KDPAR5=172372
            172374                      KDPAR6=172374
            172376                      KDPAR7=172376
```

```
                        
000001                  N=      1               ;INITALIZE FOR NEWTST
104000                  HLT=    EMT             ;SET HLT TO EMT FOR ERROR TYPEOUTS
177776                  PS=     177776          ;PROCESSOR STATUS
177776                  PSW=    PS              ;PROCESSOR STATUS WORD
000007                  BELL=   7               ;BELL
000000                  R0=     %0              ;R0 - DEFINE REGISTERS
000001                  R1=     %1              ;R1
000002                  R2=     %2              ;R2
000003                  R3=     %3              ;R3
000004                  R4=     %4              ;R4
000005                  R5=     %5              ;R5
000006                  SP=     %6              ;R6 - STACK POINTER
000007                  PC=     %7              ;R7 - PROGRAM COUNTER
000001                  BIT0=   1               ;BIT EQUATES
000002                  BIT1=   2
000004                  BIT2=   4
000010                  BIT3=   10
000020                  BIT4=   20
000040                  BIT5=   40
000100                  BIT6=   100
000200                  BIT7=   200
000400                  BIT8=   400
001000                  BIT9=   1000
002000                  BIT10=  2000
004000                  BIT11=  4000
010000                  BIT12=  10000
020000                  BIT13=  20000
040000                  BIT14=  40000
100000                  BIT15=  100000
000001          GOOD=   R1                      ;FOR GOOD DATA
000000          BAD=    R0                      ;FOR BAD DATA
```

```
                    000510                          .=      510
000510  005015  047125  052111  LOADSW: .ASCIZ  <15><12>"UNIT # "
000516  021440  000040
000522  005015  040506  040524  UNRECO: .ASCIZ  <15><12>"FATAL"<15><12><12>
000530  006514  005012     000
                                
                                
000535     015  047412  052120  .OPDR:  .ASCIZ  <15><12>"OPT DSK ADDR"
000542  042040  045523  040440
000550  042104  000122
000554  005015  040504  040524  DATA:   .ASCIZ  <15><12>"DATA "
000562  000040
                                
000564  051127  020124  051105  WRTERR: .ASCIZ  "WRT ERR"
000572  000122
                                
000574  051127  020124  045503  WCKERR: .ASCIZ  "WRT CK ERR"
000602  042440  051122     000
                                
000607     122  020104  051105  RDERR:  .ASCIZ  "RD ERR"
000614  000122
                                
000616  005015  042522  020103  RECOV:  .ASCIZ  <15><12>"REC RETRY CT "
000624  042522  051124  020131
000632  052103  000040
                                
000636  005015     000          CRLFLF: .ASCIZ  <15><12>
                                
000641     040  054450  047440  YORN:   .ASCIZ  " (Y OR N)"
000646  020122  024516     000
                                
000653     015  030412  052123  STABUF: .ASCIZ  <15><12>"1ST 4K BANK # "
000660  032040  020113  040502
000666  045516  021440  000040
                                
000674  005015  020043  043117  BUFSIZ: .ASCIZ  <15><12>"# OF 4K BANKS? "
000702  032040  020113  040502
000710  045516  037523  000040
                                
000716  005015  040522  042116  RANDM:  .ASCIZ  <15><12>"RANDOM "
000724  046517  000040
000730  005015  042524  052123  TSTNG:  .ASCIZ  <15><12>"TESTING "
000736  047111  020107     000
000743     015  050012  053517  PDOWN:  .ASCIZ  <15><12>"POWER DOWN"
000750  051105  042040  053517
000756  000116
000760  005015  040520  044522  PARERR: .ASCIZ  <15><12>"PARITY ERR"
000766  054524  042440  051122
000774     000
                                
                    000776                      .EVEN
```

```
                    001000                  .=      1000

                                            ICNT:   0               ;LH = ITERATION COUNT  ;RH = TEST NO.
                                            ERRORS: 0               ;ERROR COUNT
                                            PCNT:   0,0             ;2 WORD PASS COUNT
                                            LAD:    0               ;LOOP ADDRESS FOR SCOPE
                                            HLTADR: 0               ;ADDRESS OF LAST HLT INSTRUCTION  EXECUTED
                                            FILCHR: 1000            ;FILCHR+0 (CHAR)  ;FILCHR+1=2 (COUNT)
                                            TPS:    177564          ;OUTPUT STATUS REGISTER
                                            TKS:    177560
                                            TKB:    177562
                                            TPB:    177566          ;OUTPUT BUFFER
                                            SWR:    177570          ;SWITCH REGISTER
                                            DISPLAY:177570          ;DISPLAY REGISTER

                                            ;DISK I/O REGISTERS

                                            RSCS1:  172040          ;DISK CONTROL & STATUS REGISTER
                                            RSCS2:  172050          ;DISK CONTROL & STATUS REGISTER
                                            RSWC:   172042          ;WORD COUNT REGISTER
                                            RSBA:   172044          ;BUS ADDRESS
                                            RSDA:   172046          ;DISK ADDRESS (DESIRED ADDRESS)
                                            RSDS:   172052          ;DRIVE STATUS
                                            RSER:   172054          ;ERROR REG.
                                            RSAS:   172056          ;ATTENTION SUMMARY
                                            RSLA:   172060          ;LOOK AHEAD
                                            RSDB:   172062          ;DATA BUFFER REGISTER
                                            RSMR:   172064          ;MAINTENANCE REGISTER
                                            RSDT:   172066          ;DRIVE TYPE REGISTER
                                            RSVEC:  204             ;INTERUPT RSVEC

                                            ;BIT ASSIGNMENTS FOR ERROR TYPE OUTS

                    000002                          DB=2            ;DATA BUFFER
                    000004                          DA=4            ;DESIRED ADD
                    000010                          WC=10           ;WORD COUNT
                    000020                          BA=20           ;BUS ADDRESS
                    000040                          DS=40           ;DRIVE STATUS
                    000100                          AS=100          ;ATTENTION SUMMARY
                    000200                          LA=200          ;LOOK AHEAD
                    000220                          MR=220          ;MAINTENANCE
                    000240                          DT=240          ;DRIVE TYPE

001064  000206                              STATUS: 206                     ;DISK INTERRUPT STATUS
001066  000200                              PRIORITY:BIT7                   ;DISK PRIORITY LEVEL
```

```
                000006                          RW=6                    ;R/W IN PDR REG
                000000                          UP=0                    ;UP BITY IN PDR REG
                000250                          MMVEC=250               ;ADDR OF MEM MGMT ERROR TRAP
001070  000000                                  STAMEM: 0               ;STARTING LOC FOR -A- PORT
001072  000000                                  SAVAST: 0               ;SAVE LOC FOR STAMEM
001074  000000                                  STBCOM: 0               ;STARTING LOC FOR -B- PORT

001076  000000                                  SAVCPU: 0               ;SAVE LOC FOR CPUBM
001100  000000                                  SAVMGA: 0               ;STARTING ADDR FOR -A- PORT  WITH MEM MGMT
001102  000000                                  SAVMGB: 0               ;STARTING ADDR FOR B PORT W/MEM MGMT
001104  000000                                  SAVMGC: 0               ;STARTING LOC FOR CPU W/MEM MGMT
001106  000000                                  SIZEAP: 0               ;SIZE OF A PORT
001110  000000                                  SIZEBP: 0               ;SIZE OF B PORT
001112  000000                                  WDCTB:  0               ;WC FOR A PORT
001114  000000                                  A0B1:   0               ;FLAG FOR PORT BEING TESTED
001116  000000                                  VADDR:  0               ;VIRTUAL ADDR
001120  000000                                  PHADDR: 0               ;PHYSICAL ADDR
001122  000000                                  FLAG2:  0               ;FLAG FOR RESTART AND FOUND DRIVE
001124  000000                                  DROP:   0               ;BAD UNITS ON SYSTEM THAT GET DUMPED

                                        ;DISCRIPTION OF FLAG2

                                        ;BIT0 = RESTART
                                        ;BIT1 = FOUND DRIVE
                                        ;BIT2 = ERROR DO A CRLF FOR UNIT #
                                        ;BIT3 = DOING COMPARE
                                        ;BIT4 = SET A16 IN CSI
                                        ;BIT5 = SET A17 IN CSI
                                        ;BIT6 = SET IF MEMORY HAS ALREADY BEEN FOUND
                                        ;BIT7 = WHEN SET MAKE WC UP TO 28K
                                        ;BIT8 = FOUND MEMORY ON -B- PORT
                                        ;BIT9 = POWER DID FAIL
                                        ;BIT10 = WAITING IN BACKGROUND TEST
                                        ;BIT11 = PARITY ERROR ROUTINE
                                        ;BIT12 = POWER FAIL TEST
                                        ;BIT13 = IN MAP ROUTINE
                                        ;BIT14 = IN POWER FAIL OR CONVERSATION MODE
                                        ;BIT15 = ERROR IN POWER FAIL

                                        ;DISCRIPTION OF FLAG

                                        ;BIT0 = USED FOR WRITE COUNTER
                                        ;BIT1 = USED FOR WRITE COUNTER
                                        ;BIT2 = TRANSFER MODE 64K
                                        ;BIT5 = OPTIONAL DMA
                                        ;BIT6 = TEST -B- PORT
                                        ;BIT7 = LAST DISK BUFFER FLAG
                                        ;BIT8 = PROGRAM IS IN ADDRESS OR RANDOM TEST
                                        ;BIT9 = ERROR DURING TRANSFER
                                        ;BIT10 = DATA TEST ONLY
                                        ;BIT11 = MULTIPORT
                                        ;BIT12 = READ
                                        ;BIT13 = WRITE CHECK
                                        ;BIT14 = WRITE
                                        ;BIT15 = PROGRAM CONTROL MODE
```

```
                                ;RH11 DEDICATE REGISTERS (MEMORY)
                                FLAG:   0               ;TEST REGISTER
                                WRDCT:  0               ;WORKING WORD COUNT
                                TRACK:  0               ;WORKING DAE
                                DMA:    0               ;WORKING DAR
                                PATNU:  0               ;DATA PATTERN INDEX
                                BUF:    0               ;WORKING DATA BUFFER (OUT-IN)
                                TDMA:   0               ;TEMP DAR
                                SWRDCT: 0               ;STANDARD WORD COUNT
                                ERCOUNT:0               ;ERROR COUNT FOR MESSAGES.
                                SAVE:   0
                                HRDER:  0               ;POINTER FOR HARD ERROR
                                BLOCK:  0
                                PASSC:  0
                                UNNUM:  0               ;UNIT CURRENTLY BEING TESTED
                                UNITSV: 0               ;SET BIT=UNIT ON BUS
                                UNCMP:  0               ;FOR COMPARING FOR # OF DEVICE
                                R804DT: 0               ;FLAG FOR R804
                                NUM8:   0               ;WORK LOC FOR NUMBER INPUTS
                                CMD:    0               ;LOC FOR C82 COMMANDS
                                SWITCH: 0               ;FLAG FOR WHICH RANDOM NUMBER GEN
                                INTFLG: 0               ;FLAG FOR INTERRUPT
                                LOPCNT: 0               ;ERROR FLAG AND LOOP COUNTER FLAG
                                WRITER: 0               ;CONTAINS # OF WRITE ERRORS
                                WCERR:  0               ;CONTAINS # OF WRITE CHECK ERRORS
                                READER: 0               ;CONTAINS # OF READ ERRORS
                                COMERR: 0               ;CONTAINS # OF COMPARE ERRORS
                                MMAVA:  0               ;MEM MGMT AVAILABLE INDICATOR
                                SAVWC:  0               ;SAVE LOC FOR CONVERSATION WC ROUTINE
                                FLAG3:  0               ;LOOP IN ADDRESS + RANDOM TST FLAG
                                SAVWCB: 0               ;SAVE WC SIZE FOR -B- PORT

                                ;RH11 WORK REGISTERS
                                ;(CAN BE CHANGED IN ANY ROUTINE)
                                WORK:   0
                                WORK1:  0
                                WORK2:  0
```

```
   345  001230  005037  020116          SETSWI: CLR     $WI
   346  001234  012706  000500          BEGIN:  MOV     #500,SP         ;SET STACK TO  ••• 500 •••
   347  001240  012737  016254  000024          MOV     #.POWER,#24     ;SET UP PF VECTOR
   348  001246  012737  000340  000026          MOV     #340,#26        ;LOCK OUT THE WORLD
   349  001254  012737  015716  000030          MOV     #.HLT,#30       ;SET EMT VECTOR
   350  001262  012737  000340  000032          MOV     #340,#32        ;LOCK UP

   351  001270  012737  016622  000034          MOV     #.TRAP,#34      ;SET TRAP VECTOR
   352  001276  012737  000340  000036          MOV     #340,#36        ;LOCK UP
   353  001304  005037  001000                  CLR     ICNT            ;INIT ICNT
   354  001310  005037  001010                  CLR     LAD             ;INIT LAD
   355  001314  032737  000001  020116          BIT     #BIT0,$WI
   356  001322  001001                          BNE     2$
   357  001324  104444                          SUSWR                   ;SIZE FOR SWITCHLESS
   358  001326  042737  177677  001126  2$:     BIC     #177677,FLAG    ;CLEAR FLAG
   359  001334  042737  177776  001122          BIC     #177776,FLAG2   ;CLEAR ALL EXECPT RESTART
   360  001342  005037  001216                  CLR     FLAG3           ;CLEAR LOOP IN ADDRESS + RANDOM TST FLAG
   361  001346  032737  000001  001122          BIT     #BIT0,FLAG2     ;IS THIS THE FIRST TIME?
   362  001354  001002                          BNE     1$              ;NO
   363  001356  004737  020120                  JSR     PC,LDR          ;SAVE LOADER
   364  001362  000005                  1$:     RESET                   ;CLEAR THE WORLD
   365  001364  012737  000340  177776          MOV     #340,PS         ;LOCK UP INTERRUPT LEVELS
   366  001372  004537  012432                  JSR     R5,ERRCL        ;CLEAR ERROR COUNTER + PASS CNT
   367  001376  005037  001212                  CLR     MMAVA           ;CLEAR MEM MGMT FLAG
   368  001402  005037  001114                  CLR     AOB1            ;TEST A PORT FIRST
   369  001406  032777  000004  177412          BIT     #BIT2,@SWR      ;WANT MEM MGMT?
   370  001414  001021                          BNE     3$              ;NO
   371  001416  012737  001444  000004          MOV     #5$,4           ;SET TIMEOUT TRAP
   372  001424  012737  000340  000006          MOV     #340,6          ;SET PS
   373  001432  005037  177572                  CLR     @#SR0           ;IS MEM MGMT AVAILABLE?
   374  001436  005137  001212                  COM     MMAVA           ;YES
   375  001442  000401                          BR      4$              ;CONT
   376  001444  022626                  5$:     CMP     (6)+,(6)+       ;CLEAR STACK
   377  001446  012737  000006  000004  4$:     MOV     #6,4            ;RESET
   378  001454  005037  000006                  CLR     6               ;TRAP
   379  001460  032737  000001  001122  3$:     BIT     #BIT0,FLAG2     ;IS THIS THE FIRST TIME
   380  001466  001002                          BNE     CALM            ;NO
   381  001470  000137  020210                  JMP     $IZZAP          ;SIZE BUFFERS
   382  001474  004737  011472          CALM:   JSR     PC,@#EXTMEM     ;SET UP DATA BUFFERS
   383  001500  004737  015202          CALM1:  JSR     PC,.MAMK        ;TURN ON PARITY MEM
   384  001504  032737  000001  001122          BIT     #BIT0,FLAG2     ;1ST TIME ?
   385  001512  001006                          BNE     3$              ;NO
   386  001514  013737  001144  001214          MOV     $WRDCT,$AVWC    ;SAVE WC FOR CONVERSATION MODE COMPARE
   387  001522  013737  001112  001220          MOV     WDCTB,$AVWCB    ;SAVE WC FOR -B- PORT
   388  001530  052737  000001  001122  3$:     BIS     #BIT0,FLAG2     ;SET 1ST TIME FLAG
   389  001536  005037  001134                  CLR     DMA             ;CLEAR DAR REGISTERS
   390  001542  005037  001136                  CLR     PATNU           ;CLEAR PATTER COUNT
   391  001546  013737  001144  001130          MOV     $WRDCT,WRDCT
   392  001554  032777  000002  177244          BIT     #BIT1,@SWR      ;DATA TEST ONLY?
   393  001562  001403                          BEQ     2$              ;NO
   394  001564  052737  002000  001126          BIS     #BIT10,FLAG     ;YES
   395  001572  032777  002000  177226  2$:     BIT     #BIT10,@SWR     ;ENTER CONVERSATION MODE?
   396  001600  001007                          BNE     1$              ;YES GO TO CONVERSATION MODE
   397  001602  052737  074000  001126          BIS     #74000,FLAG
   398  001610  004537  010214                  JSR     R5,RESTOR       ;RESTORE ORIGINAL WD CNT
   399  001614  000137  003224                  JMP     ADTST
   400  001620  000137  002220          1$:     JMP     @#CONN          ;ENTER CONVERSATION MODE
```

```
                                        ;FIND OUT HOW MANY DRIVES
                                        ;FIRST TEST RSAS
                                        
001624  012701  000010        DRVENO: MOV     #8.,R1          ;PUT 8 INTO R1 FOR COUNT
001630  042737  000002  001122        BIC     #BIT1,FLAG2     ;CLEAR FOUND DRIVE FLG
001636  012777  000000  177170        MOV     #0,@RSCS2       ;SET DEVICE TO ZERO
001644  012777  000007  177174  TRY:  MOV     #7,@RSER        ;CAUSE AN ERROR +SETS BIT IN AS REG
001652  005301                        DEC     R1              ;DO A MAXIMUN OF 16 TIMES
001654  001403                        BEQ     DVNUM           ;TESTED FOR ALL DRIVES GET OUT
001656  005277  177152                INC     @RSCS2          ;INCREMENT DRIVE UNIT
001662  000770                        BR      TRY             ;REPEAT FOR NEXT DRIVE
001664  017737  177160  001162  DVNUM: MOV     @RSAS,UNITSV    ;SAVE
001672  043737  001124  001162        BIC     DROP,UNITSV     ;DROP BAD DRIVES
001700  012737  000401  001164        MOV     #401,UNCMP      ;SETUP TO CMP WITH UNITSV
001706  012737  000000  001160        MOV     #0,UNNUM        ;PUT 0 INTO UNIT NO.
001714  032777  000040  177104        BIT     #BIT5,@#SR      ;INHIBIT TYPE OUT?
001722  001005                        BNE     STTEST          ;YES
001724  104402  000730                TYPE    ,TSTNG
001730  042737  000004  001122        BIC     #BIT2,FLAG2     ;CLEAR ERROR FLAG
001736  033737  001164  001162  STTEST: BIT     UNCMP,UNITSV    ;IS THIS DRIVE ON THE SYSTEM
001744  001473                        BEQ     TRYNX           ;NO
001746  013777  001160  177060  UNTYP:  MOV     UNNUM,@RSCS2    ;YES PUT UNIT # INTO CS2
001754  005037  001166                CLR     RS04DT          ;CLEAR DRIVE TYPE FLAG
001760  022777  000004  177072        CMP     #4,@RSDT        ;RS03LA?
001766  001004                        BNE     8$              ;NO
001770  012737  000004  001166        MOV     #4,RS04DT       ;SET DRIVE TYPE FLAG
001776  000422                        BR      1$              ;CONT
002000  005777  177054          8$:   TST     @RSDT           ;IS THIS A RS03?
002004  001417                        BEQ     1$              ;YES
002006  022777  000001  177044  2$:   CMP     #1,@RSDT        ;IS THIS A RS03 4US?
002014  001413                        BEQ     1$              ;YES
002016  022777  000002  177034  3$:   CMP     #2,@RSDT        ;IS THIS A RS04?
002024  001404                        BEQ     6$              ;YES
002026  022777  000003  177024        CMP     #3,@RSDT        ;RS04?
002034  001037                        BNE     TRYNX           ;GET A NEW NUMBER
002036  052737  177777  001166  6$:   BIS     #-1,RS04DT      ;YES RS04
002044  032737  040000  001122  1$:   BIT     #BIT14,FLAG2    ;IN POWER FAIL OR CONVERSATION?
002052  001401                        BEQ     7$              ;NO
002054  000207                        RTS     PC              ;YES
002056  032777  000200  176760  7$:   BIT     #BIT7,@RSDS     ;IS THIS DRIVE READY ?
002064  001423                        BEQ     TRYNX           ;NO GET ANOTHER DRIVE
002066  032777  000040  176732        BIT     #BIT5,@#SR      ;TYPEOUT?
002074  001016                        BNE     4$              ;NO
002076  032737  000004  001122        BIT     #BIT2,FLAG2     ;WAS THERE AN ERRER?
002104  001402                        BEQ     5$              ;NO
002106  104402  000636                TYPE    ,CRLFLF
002112                          5$:
002112  013746  001160                MOV     UNNUM,-(6)      ;PUT UNNUM ON STACK
002116  104406                        TYPOS                   ;TYPE STACK IN OCTAL - SUPRESS
002120  104402  000040                TYPE    ,40             ;TYPE SPACE
002124  042737  000004  001122        BIC     #BIT2,FLAG2     ;CLEAR ERROR FLAG
002132  000426                        BR      NOWGO           ;NOW TEST
```

```
                                TRYNX:  ASL     UNCMP           ;CHECK NEXT BIT FOR DRIVE
                                        BCS     CHCKDV          ;DID WE TEST ANY REG?
                                        INC     UNNUM           ;INC UNIT #
                                        BR      $TTEST          ;CHECK FOR NEXT DRIVE

                                ;THIS PROGRAM WILL DEFAULT TO TESTING UNIT 0 IF IT CAN NOT FIND ANY DRIVES
                                CHCKDV: BIT     #BIT1,FLAG2     ;FOUND DRIVE?
                                        BNE     DONEE           ;YES WE DID TEST A DRIVE
                                        MOV     #100000,UNCMP   ;NO DRIVES TESTED, COULD NOT SET
                                        CLR     UNNUM           ;ANY AS BITS, THUS DEFAULTS TO 0
                                        MOV     UNNUM,-(6)      ;PUT UNNUM ON STACK
                                        TYPES                   ;TYPE STACK IN OCTAL - SUPRESS
                                        HALT                    ;COULD NOT SET ANY ATA BITS
                                                                ;BY SETTING ERROR BITS
                                                                ;GO BACK AND USE OTHER DIAG.
                                        BR      NOWGO           ;DEFAULT TO DRIVE 0
                                DONEE:  JMP     OUT             ;GET OUT
                                NOWGO:  BIS     #BIT1,FLAG2     ;FOUND DRIVE
                                        RTS     PC

                                ;ENTER OPERATOR CONVERSATION MODE

                                CONM:
                                        TYPE    .,+2            ;,ASCII <15><12>"-A- PORT"
                                        JSR     PC,CMPY         ;COMPARE FOR YES
                                        BEQ     2$              ;YES
                                        MOV     #-1,A0B1        ;B PORT
                                        JMP     1$              ;TEST -B- PORT
                                2$:     TYPE    ,STABUF
                                        RDOCT
                                        MOV     (6)+,STAMEM     ;START BUFFER AT 4K
                                3$:     TYPE    ,BUFSIZ
                                        RDOCT                   ;GET NUMBER
                                        MOV     (6)+,SIZEAP     ;SAVE IT
                                        CMP     #6,SIZEAP       ;CAN ONLY XFER 24K
                                        BLT     3$              ;GREATER THEN 24K
                                        JMP     NOPORT          ;GET OUT
```

```
002320  104402  000653          1$:     TYPE    ,STABUF
002324  104420                          RDOCT                   ;GET ANS
002326  012637  001074                  MOV     (6)+,STBCOM     ;AND SAVE IT
002332  104402  000674          4$:     TYPE    ,BUFSIZ
002336  104420                          RDOCT                   ;GET ANS
002340  012637  001110                  MOV     (6)+,SIZEBP     ;SAVE IT
002344  022737  000006  001110          CMP     #6,SIZEBP       ;GREATER THEN 24K?
002352  002767                          BLT     4$              ;YES ASK AGAIN
002354  052737  000100  001126          BIS     #BIT6,FLAG      ;SET B PORT FLAG
002362  004737  011472          NOPORT: JSR     PC,EXTMEM       ;CAL BUFFERS AND WC
002366  013737  001144  001130          MOV     SWRDCT,WRDCT    ;GET STANDARD WC
002374  052737  002000  001126  A1:     BIS     #BIT10,FLAG     ;SET BIT FOR DATA TEST ONLY
002402  004537  012432                  JSR     R5,ERRCL        ;CLEAR ERROR CNT + PASS CNT
002406  042737  174040  001126          BIC     #174040,FLAG    ;CLEAR MULTI FLAG MODE +PATTERN SELECT
002414  104402  002420                  TYPE    ,.+2            ;.ASCIZ <15><12>"MULTI DRIVE"
002436  004737  003300                  JSR     PC,CMPY         ;COMPARE FOR YES
002442  001004                          BNE     DATTES          ;ANS IS NO
002444  052737  004000  001126          BIS     #BIT11,FLAG     ;SET BIT FOR MULTI DRIVE
002452  000434                  1$:     BR      ASKWC
002454  104402  000510          DATTES: TYPE    ,LOADSW
002460  104420                          RDOCT
002462  012637  001170                  MOV     (6)+,NUMS       ;GET NUMBER
002466  022737  000010  001170          CMP     #10,NUMS        ;CORRECT # ?
002474  103767                          BLO     DATTES          ;NO
002476  013737  001170  001160          MOV     NUMS,UNNUM      ;SET UNIT #
002504  004737  006540                  JSR     PC,FNDTYP               ;TEST FOR RS04 OR 03
002510  005002                  1$:     CLR     R2              ;CLEAR WORK AREA
002512  000261                          SEC                     ;SET CARRY
002514  006102                  2$:     ROL     R2              ;SET BIT IN WORK
002516  005737  001170                  TST     NUMS            ;IS THIS THE RIGHT BIT FOR THE RIGHT DISK
002522  001403                          BEQ     3$              ;YES
002524  005337  001170                  DEC     NUMS            ;NO TRY AGAIN
002530  000771                          BR      2$              ;TEST AGAIN
002532  010237  001162          3$:     MOV     R2,UNITSV       ;SET DRIVE BIT IN UNITSV
002536  052737  000002  001122          BIS     #BIT1,FLAG2     ;SET FOUND DRIVE FLAG

002544                          ASKWC:
002544  104402  002550                  TYPE    ,.+2            ;.ASCIZ <15><12>"OPT WD CT"
002564  004737  003300                  JSR     PC,CMPY         ;COMPARE FOR YES
002570  001403                          BEQ     WCCON           ;YES
002572  004537  010214                  JSR     R5,RESTOR       ;RESTORE ORIGINAL WD CNT
002576  000444                          BR      OPDAR           ;CONT
```

```
532  002600                          WCCON:
533  002600  104402  002604                  TYPE    ,.+2            ;.ASCIZ <15><12>"WD CT "
534  002616  104420                          RDOCT
535  002620  012637  001170                  MOV     (6)+,NUM8       ;GET NUMBER
536  002624  005737  001170                  TST     NUM8            ;IS IT 0?
537  002630  001763                          BEQ     WCCON           ;YES ASK AGAIN FOR LENGTH

538  002632  013702  001214                  MOV     $AVWC,R2        ;GET STANDARD WC FOR -A- PORT
539  002636  005737  001114                  TST     A@B1            ;-A- PORT?
540  002642  001402                          BEQ     1$              ;YES
541  002644  013702  001220                  MOV     $AVWCB,R2       ;GET WC FOR -B- PORT
542  002650  005202                  1$:     INC     R2
543  002652  020237  001170                  CMP     R2,NUM8         ;IS NUM8 LESS THAN 8WRDCT
544  002656  101750                          BLOS    WCCON           ;YES ASK FOR COUNT AGAIN
545  002660  013737  001170  001144          MOV     NUM8,8WRDCT     ;OPERATING WORD COUNT
546  002666  013737  001144  001130          MOV     8WRDCT,WRDCT
547  002674  005737  001114                  TST     A@B1            ;B PORT?
548  002700  001403                          BEQ     OPDAR           ;NO
549  002702  013737  001130  001112          MOV     WRDCT,WDCTB     ;YES GET WC
550
551  002710  104402  000535          OPDAR:  TYPE    ,,OPDR
552  002714  004737  003300                  JSR     PC,CMPY         ;COMPARE FOR YES
553  002720  001037                          BNE     OPPAT           ;ANS IS NO
554  002722  052737  000040  001126          BIS     $BITS,FLAG      ;SET OPTIONAL DMA FLAG
555  002730  104402  002734                  TYPE    ,.+2            ;.ASCIZ "="
556  002736  104420                          RDOCT
557  002740  012637  001170                  MOV     (6)+,NUM8       ;GET NUMBER
558  002744  022737  000004  001166          CMP     #4,RS04DT       ;RS03LA?
559  002752  001004                          BNE     3$              ;NO
560  002754  022737  003777  001170          CMP     #3777,NUM8      ;IS ADDR. CORRECT?
561  002762  000412                          BR      2$
562  002764  005737  001166          3$:     TST     RS04DT          ;IS THIS A RS04?
563  002770  001404                          BEQ     1$              ;NO
564  002772  022737  017777  001170          CMP     #17777,NUM8     ;IS ADD. CORRECT?
565  003000  000403                          BR      2$              ;GET OUT
566  003002  022737  007777  001170  1$:     CMP     #7777,NUM8      ;IS ADD. CORRECT?
567  003010  101737                  2$:     BLOS    OPDAR           ;NO
568  003012  013737  001170  001142          MOV     NUM8,TDMA       ;TEMP SECTOR REGISTER
```

```
                                        OPPAT:
003020                                          TYPE    ,.+2            ;.ASCIZ <15><12>"PATN # "
003020  104402  003024                          RDOCT
003036  104420                                  MOV     (5)+,NUM8       ;GET NUMBER
003040  012637  001170                          CMP     #23,NUM8        ;TEST FOR CORRECT NO
003044  022737  000023  001170                  BLOS    OPPAT           ;ASK AGAIN
003052  101762                                  CLR     PATNU           ;CLEAR PATTERN #
003054  005037  001136                          CMP     #22,NUM8
003060  022737  000022  001170                  BEQ     OPWRT           ;DATA PATTERN UNDER PROGRAM CONTROL
003066  001411                                  BIS     #BIT15,FLAG     ;SET PROGRAM FLAG
003070  052737  100000  001126                  MOV     NUM8,PATNU      ;OPERATOR WANTS TO SELECT DATA
003076  013737  001170  001136                  CLC
003104  000241                                  ROL     PATNU
003106  006137  001136
                                        OPWRT:
003112                                          TYPE    ,.+2            ;.ASCIZ <15><12>"WRITE"
003112  104402  003116                          JSR     PC,CMPY         ;COMPARE FOR YES
003126  004737  003300                          BNE     OPRD            ;ASK ABOUT WRITE CHECK
003132  001003                                  BIS     #BIT14,FLAG     ;YES SET FLAG BIT
003134  052737  040000  001126
                                        OPRD:
003142                                          TYPE    ,.+2            ;.ASCIZ <15><12>"READ"
003142  104402  003146                          JSR     PC,CMPY         ;COMPARE FOR YES
003156  004737  003300                          BNE     OPWCK
003162  001003                                  BIS     #BIT12,FLAG     ;SET FLAG TO READ
003164  052737  010000  001126
                                        OPWCK:
003172                                          TYPE    ,.+2            ;.ASCIZ <15><12>"WRT CK"
003172  104402  003176                          JSR     PC,CMPY         ;COMPARE FOR YES
003210  004737  003300                          BNE     ADTST
003214  001003                                  BIS     #BIT13,FLAG
003216  052737  020000  001126

003224  032737  004000  001126  ADTST:  BIT     #BIT11,FLAG     ;ARE WE IN MULTI DRIVE MODE
003232  001402                          BEQ     EXMFLG          ;BRANCH IF NO.
003234  004737  001624                  JSR     PC,DRVENO       ;GET DRIVES TO BE TESTED
003240  042737  000004  001126  EXMFLG: BIC     #BIT2,FLAG      ;CLEAR XFER MODE FLAG
003246  032777  000400  175552          BIT     #BIT8,@SWR      ;XFER MODE?
003254  001403                          BEQ     1$              ;NO
003256  052737  000004  001126          BIS     #BIT2,FLAG      ;SET XFER MODE FLAG
003264  032737  002000  001126  1$:     BIT     #BIT10,FLAG     ;TEST FOR DATA TEST ONLY
003272  001414                          BEQ     ADT1            ;DO COMPLETE TEST
003274  000137  003710                  JMP     DATAT           ;DO DATA TEST ONLY

003300  104402  000641          CMPY:   TYPE    ,YORN
003304  104422                          RDLIN
003306  122737  000131  017032          CMPB    #'Y,INPUT       ;TEST FOR YES
003314  000207                          RTS     PC
003316  052737  100000  001216  ADTL:   BIS     #BIT15,FLAG3    ;SET LOOP IN ADDRESS TEST FLAG GOT HERE
                                                                ;BECAUSE PROGRAM WAS STARTED AT 234
```

```
                                ;RH11 ADDRESS TEST #1 (TRACK AND SECTOR SELECTION TEST)
                                ;WRITE 100(OCTAL) RS03, 200(OCTAL) RS04, WORDS IN EACH SECTOR
                                ;THE WORD CONTAINS THE ADDRESS OF EACH SECTOR
                                ;WHEN THE COMPLETE DISK IS WRITTEN READ
                                ;BACK EACH SECTOR AND COMPARE FOR THE CORRECT
                                ;DATA IN THE SECTOR
                                ;PS IS AT LEVEL 7 SO NO INTERRUPTS

003324                          ADT1:                           ;ADDRESS TEST
                                ;*****************************************************************
                                ;TEST 1         ADDRESS TEST
                                ;*****************************************************************
003324  104400                  TST1:   SCOPE
003326  032737  000004  001126  ADT1A:  BIT     #BIT2,FLAG      ;XFER MODE?
003334  001402                          BEQ     3$              ;NO
003336  000137  006710                  JMP     DATAT           ;YES
003342  012737  000340  177776  3$:     MOV     #340,PS         ;LOCK UP PS
003350  012737  020000  017416          MOV     #20000,OUTBUF   ;START BUF AT 20000
003356  052737  000400  001126          BIS     #BIT8,FLAG      ;SET TEST FLAG
003364  013737  001144  001150          MOV     $WRDCT,SAVE     ;SAVE STD WD COUNT
003372  005037  001134                  CLR     DMA             ;CLEAR DISK ADD
003376  104426                          CLRDV                   ;INIT DRIVE
003400  004737  006562                  JSR     PC,WNTHU        ;GET WORD COUNT
003404  013737  001130  001144  5$:     MOV     WRDCT,$WRDCT
003412  013737  017416  001140  2$:     MOV     OUTBUF,BUF      ;SET UP CURRENT ADDRESS
003420  104414                  SEABUF: ERCLR                   ;CLEAR RS REGISTERS IF ERROR
003422  013700  017416                  MOV     OUTBUF,R0       ;SET UP ADDRESS BUFFER
003426  013701  001130                  MOV     WRDCT,R1        ;
003432  013720  001134          XSEABUF:MOV     DMA,(0)+        ;LOAD OUTBUF WITH DATA TO BE WRITTEN
003436  005301                          DEC     R1              ;FILL OUTBUF
003440  001374                          BNE     XSEABUF         ;WITH DATA
003442  012737  000061  001172          MOV     #61,CMD         ;WRITE NO I/E
003450  104416                          DKCMD                   ;GO WRITE
003452  105777  175354                  TSTB    @RSCS1          ;CHECK FOR READY
003456  100375                          BPL     .-4
003460  005777  175346                  TST     @RSCS1          ;TEST FOR ERROR
003464  100010                          BPL     WRNEXB          ;BRANCH IF NO ERROR
003466  012737  003420  001010          MOV     #SEABUF,LAD     ;SET UP LOOP ADDRESS
003474  052737  001000  001126          BIS     #BIT9,FLAG      ;SET ERROR BIT IN FLAG
003502  104430                          LOGW                    ;LOG WRITE ERROR
003504  104034                          HLT     !WC!DA!BA
003506  104400                  WRNEXB: SCOPE
003510  004737  007050                  JSR     PC,DISBUF       ;SET UP NEXT DISK ADDR.
003514  000741                          BR      SEABUF          ;WRITE NEXT SECTOR
003516  104400                  RRDSEC: SCOPE
```

```
                                RDSECT: ERCLR                   ;CLEAR ERRORS
003520  104414
003522  012737  000071  001172          MOV     #71,CMD         ;READ NO I/E
003530  104416                          DKCMD                   ;DO A READ
003532  105777  175274                  TSTB    @RSCS1          ;CHECK FOR READY
003536  100375                          BPL     .-4             ;NOT READY BRANCH BACK
003540  005777  175266                  TST     @RSCS1          ;TEST FOR ERROR
003544  100006                          BPL     ADHGT           ;BRANCH IF NO ERROR
003546  052737  001000  001126          BIS     #BIT9,FLAG      ;SET ERROR FLAG
003554  104432                          LOGR                    ;LOG READ ERROR
003556  104014                          HLT     ;WC!DA
003560  104400                          SCOPE
003562  013702  017416          ADHGT:  MOV     OUTBUF,R2
003566  013746  001130                  MOV     WRDCT,-(SP)     ;SAVE OLD WC
003572  004737  006562                  JSR     PC,WHTHU
003576  013703  001130                  MOV     WRDCT,R3
003602  012637  001130                  MOV     (SP)+,WRDCT     ;RESTORE OLD WC
003606  023712  001134          SANHT:  CMP     DMA,(2)         ;CMP FOR CORRECT ADDR.
003612  001004                          BNE     ADERR           ;BRANCH IF DATA DID NOT COMPARE
003614  005722                          TST     (2)+            ;GET NEXT ADDRESS OF INBUF
003616  005303                          DEC     R3              ;DEC SECTOR COUNT
003620  001372                          BNE     SANHT           ;TEST NEXT WORD
003622  000412                          BR      CHKADT
003624  013701  001134          ADERR:  MOV     DMA,GOOD        ;CORRECT ADDRESS
003630  011200                          MOV     (2),BAD         ;DATA IN ERROR
003632  104000                          HLT                     ;DISK ADD DID NOT MATCH WRITTEN ADDRESS
003634  104436                          LOGC                    ;LOG COMPARE ERROR
003636  004737  014204                  JSR     PC,PRNT         ;INHIBIT TYPEOUT?
003642  001002                          BNE     CHKADT          ;YES
003644  104402  000636                  TYPE    ,CRLFLF

                        ;*****REPORT ONLY ONE ERROR PER SECTOR*****

003650  104400          CHKADT: SCOPE
003652  004737  007050                  JSR     PC,DISBUF       ;SET UP NEXT DISK BUFFER
003656  000717                          BR      RRDSEC          ;CHECK NEXT SECTOR
003660  013737  001150  001144          MOV     SAVE,SWRDCT     ;GET STD WD COUNT
003666  042737  000400  001126          BIC     #BIT8,FLAG      ;CLEAR TEST FLAG
003674  032737  100000  001216          BIT     #BIT15,FLAG3    ;DOES OPERATOR WANT TO LOOP ON TEST
003702  001402                          BEQ     .+6             ;NO
003704  000137  003326                  JMP     ADT1A           ;YES
```

```
 703  003710                          DATAT:                          ;DATA TEST
 704                                  ;*****************************************************************
 705                                  ;TEST 2          DATA TEST
 706                                  ;*****************************************************************
 707  003710  104400                  TST2:   SCOPE
 708  003712  005037  001134                  CLR     DMA             ;CLEAR DISK ADDRESS

 709  003716  104426                          CLRDV                   ;CLEAR RS REGISTERS
 710  003720  013737  001144  001130          MOV     $WRDCT,WRDCT
 711  003726  032737  000100  001126          BIT     #BIT6,FLAG      ;MULTI PORT?
 712  003734  001403                          BEQ     3$              ;NO
 713  003736  005737  001114                  TST     A0B1            ;A OR B PORT?
 714  003742  001003                          BNE     1$              ;B PORT
 715  003744  004737  011350          3$:     JSR     PC,APORT        ;A PORT
 716  003750  000402                          BR      2$
 717  003752  004737  011416          1$:     JSR     PC,BPORT        ;B PORT
 718  003756  004737  006774          2$:     JSR     PC,VECTRR       ;SETUP INT VECTOR
 719  003762  012777  000340  175074          MOV     #340,@STATUS    ;SET DISK STATUS REG LOC (206)
 720  003770  012737  004022  001152          MOV     #LDAT,HRDER     ;SETUP FOR HARD ERROR RETURN
 721  003776  013737  017416  001140          MOV     OUTBUF,BUF      ;SETUP OUTPUT BUFFER
 722  004004  052737  000003  001126          BIS     #3,FLAG         ;SET COUNTER TO 3
 723  004012  004537  007716                  JSR     R5,PASEL        ;SET UP DATA BUFFERS
 724  004016  005037  001200          LDAT1:  CLR     LOPCNT          ;CLEAR ERROR FLAG
 725  004022  104414                  LDAT:   ERCLR                   ;CLEAR RS REG, IF ERROR
 726  004024  004537  006500                  JSR     R5,OPDSEL       ;SET UP DISK ADDRESS
 727  004030  032737  040000  001126          BIT     #BIT14,FLAG     ;TEST FOR WRITE
 728  004036  001456                          BEQ     $LH             ;TEST FOR READ
 729  004040  012737  000161  001172          MOV     #161,CMD        ;WRITE WITH I/E
 730  004046  104416                          DKCMD                   ;DO A WRITE
 731  004050  004737  011750                  JSR     PC,WATT         ;WAIT FOR INTERRUPT
 732  004054  012737  004022  001010          MOV     #LDAT,LAD       ;SETUP SCOPE LOOP
 733  004062  032737  001000  001126          BIT     #BIT9,FLAG      ;WAS THERE AN ERROR?
 734  004070  001423                          BEQ     WRXBL           ;CONT
 735  004072  104430                          LOGW                    ;LOG WRITE ERROR
 736  004074  005237  001200                  INC     LOPCNT          ;SET ERROR FLAG
 737  004100  004737  014204          2$:     JSR     PC,PRNT         ;TYPE ?
 738  004104  001004                          BNE     1$              ;NO
 739  004106  104402  000554                  TYPE    ,DATA
 740  004112  104402  000564                  TYPE    ,WRTERR
 741  004116  104044                  1$:     HLT     ;D$;DA          ;WRITE ERROR
 742  004120  005337  001126                  DEC     FLAG            ;DEC COUNTER
 743  004124  032737  000003  001126          BIT     #3,FLAG         ;DONE YET WITH 3RD TRY?
 744  004132  001333                          BNE     LDAT            ;NOT 3 TRIES YET? TRY AGAIN
 745  004134  004737  011454                  JSR     PC,WTNO         ;TYPE CAN NOT WRITE
```

```
  746  004.40  005737  001200          WRXBL:  TST     LOPCNT          ;WAS THERE AN ERROR?
  747  004144  001402                          BEQ     WRX1            ;NO
  748  004146  004737  011772                  JSR     PC,TYPREC       ;TYPE RECOVERED
  749  004152  005037  001200          WRX1:   CLR     LOPCNT          ;CLEAR ERROR FLAG
  750  004156  104400                          SCOPE
  751  004160  052737  000003  001126          BIS     #3,FLAG         ;CLEAR RETRY COUNT
  752  004166  004737  007050                  JSR     PC,DISBUF       ;SET BUFFER FOR WRITE CHECK
  753  004172  000711                          BR      LDAT1
  754  004174  104400                  SLH:    SCOPE
  755  004176  104414                  SLH2:   ERCLR                   ;CLEAR RS REG IF ERRORS
  756  004200  004537  006500                  JSR     R5,OPDSEL       ;IS THE OPERATOR SELECTING THE TRACK
  757  004204  032737  020000  001126          BIT     #BIT13,FLAG     ;TEST FOR WRITE CHECK
  758  004212  001002                          BNE     1$              ;YES
  759  004214  000137  004520                  JMP     ESH1            ;NO
  760  004220  013737  017416  001140  1$:     MOV     OUTBUF,BUF      ;SET UP CURRENT ADDRESS
  761  004226  012737  000151  001172          MOV     #151,CMD        ;WRITE CHECKWITH I/E
  762  004234  104416                          DKCMD                   ;GO WRITE CHECK
  763  004236  004737  011750                  JSR     PC,WATT         ;WAIT FOR INTERRUPT
  764  004242  032737  001000  001126  XESH:   BIT     #BIT9,FLAG      ;IS THERE AN ERROR?
  765  004250  001505                          BEQ     1$              ;NO ERROR
  766  004252  005737  001200                  TST     LOPCNT          ;1ST ERROR?
  767  004256  001001                          BNE     2$              ;NO
  768  004260  104434                          LOGWC                   ;YES LOG ERROR
  769  004262  032777  000100  174536  2$:     BIT     #BIT6,@SWR      ;TYPE ALL ERRORS?
  770  004270  001007                          BNE     3$              ;YES
  771  004272  032777  001000  174526          BIT     #BIT9,@SWR      ;LOOP ON ERROR?
  772  004300  001003                          BNE     3$              ;YES
  773  004302  005737  001200                  TST     LOPCNT          ;FIRST ERROR?
  774  004306  001056                          BNE     10$             ;NO
  775  004310  004737  014204          3$:     JSR     PC,PRNT         ;TYPE OUT?
  776  004314  001052                          BNE     4$              ;NO
  777  004316  104402  000554                  TYPE    ,DATA
  778  004322  104402  000574                  TYPE    ,WCKERR
  779  004326  017702  174506                  MOV     @R5BA,R2        ;GET CORRECT BA
  780  004332  023702  017416                  CMP     OUTBUF,R2       ;DID A WD GET XFERED?
  781  004336  001406                          BEQ     9$              ;NO
  782  004340  032777  000400  174460          BIT     #BIT8,@SWR      ;XFER MODE?
  783  004346  001002                          BNE     9$              ;YES
  784  004350  162702  000002                  SUB     #2,R2
  785  004354  004737  014204          9$:     JSR     PC,PRNT         ;TYPEOUT ERRORS?
  786  004360  001030                          BNE     4$              ;NO
  787  004362  005737  001212                  TST     MMAVA           ;IS MEM MGMT AVAILABLE?
  788  004366  001402                          BEQ     7$              ;NO
```

```
  789  004370  004737  006626                    JSR     PC,PHYCOV       ;YES GET VITURAL ADDR
  790  004374  010237  001222          78:       MOV     R2,WORK         ;GET BA
  791  004400                            88:
  792  004400  104402  004404                    TYPE    ,.+2            ;.ASCII <15><12>"(BA)="
  793  004414                            68:
  794  004414  017746  174602                    MOV     @WORK,-(6)      ;PUT @WORK ON STACK

  795  004420  104404                            TYPEO                   ;TYPE STACK IN OCTAL
  796  004422  104402  004426                    TYPE    ,.+2            ;.ASCII " MC="
  797  004434  017746  174376                    MOV     @RSMC,-(6)      ;PUT @RSMC ON STACK
  798  004440  104404                            TYPEO                   ;TYPE STACK IN OCTAL
  799  004442  104026                    48:     HLT     !DA!DB!BA       ;NOTE: BA REG. = +2 OF ACTUAL MEMORY
  800                                             ;LOC AFTER WORDS HAVE BEEN XFERED
  801  004444  005237  001200          108:      INC     LOPCNT          ;INC ERROR COUNT
  802  004450  022737  000010  001200            CMP     #10,LOPCNT      ;10 TRYS YET?
  803  004456  001247                            BNE     $LM2            ;NO
  804  004460  004737  006520                    JSR     PC,NOREC        ;TYPE UNRECOVERABLE
  805  004464  005737  001200          18:       TST     LOPCNT          ;ANY ERRORS?
  806  004470  001402                            BEQ     58              ;NO
  807  004472  004737  011772                    JSR     PC,TYPREC       ;TYPE RECOVERED
  808  004476  005037  001200          58:       CLR     LOPCNT          ;CLEAR ERROR COUNTER
  809  004502  104400                            SCOPE
  810  004504  012737  004176  001010            MOV     #$LM2,LAD       ;SETUP LOOP ADDRESS
  811  004512  004737  007050                    JSR     PC,DISBUF       ;SET UP THE DISK BUFFER
  812  004516  000422                            BR      $LM2A
  813  004520  004537  011302          ESM1:     JSR     R5,CLEAR        ;CLEAR BUFFER
  814  004524  004537  006500          ESM:      JSR     R5,OPDSEL       ;OPERATOR SELECTED DISK ADDRESS?
  815  004530  032737  010000  001126            BIT     #BIT12,FLAG     ;TEST FOR READ
  816  004536  001002                            BNE     18              ;YES
  817  004540  000137  004754                    JMP     @STR            ;NO READ
  818  004544  104400                    18:     SCOPE
  819  004546  042737  000003  001126            BIC     #3,FLAG         ;CLEAR RE-READ COUNT
  820  004554  005037  001200                    CLR     LOPCNT          ;CLEAR FLAG
  821  004560  000137  004570                    JMP     DSKRD           ;CONT
  822  004564  000137  004176          $LM2A:    JMP     $LM2
  823  004570  104414                    DSKRD:  ERCLR                   ;CLEAR RS REG IF ERRORS
  824  004572  012737  000171  001172            MOV     #171,CMD        ;READ WITH I/E
  825  004600  104416                            DKCMD                   ;READ
  826  004602  032777  010000  174216            BIT     #10000,@SWR     ;COMPARE?
  827  004610  001007                            BNE     TAG             ;NO
  828  004612  032737  000004  001126            BIT     #BIT2,FLAG      ;COMPARE?
  829  004620  001003                            BNE     TAG             ;NO
  830  004622  004537  010446                    JSR     R5,COMPARE      ;COMPARE
  831  004626  000402                            BR      ELM
```

```
  832  004630  004737  011750          TAG:   JSR     PC,WATT          ;WAIT FOR INTERRUPT
  833  004634  032737  001000  001126  ELH:   BIT     #BIT9,FLAG       ;IS THERE AN ERROR?
  834  004642  001433                         BEQ     ADRD             ;NO
  835  004644  032777  000100  174154         BIT     #BIT6,@SWR       ;SOFT ERROR TYPEOUT?
  836  004652  001003                         BNE     3$               ;YES
  837  004654  005737  001200                 TST     LOPCNT           ;FIRST ERROR?
  838  004660  001011                         BNE     2$               ;NO
  839  004662  004737  014204          3$:    JSR     PC,PRNT          ;TYPEOUT?
  840  004666  001004                         BNE     1$               ;NO
  841  004670  104402  000554                 TYPE    ,DATA
  842  004674  104402  000607                 TYPE    ,RDERR
  843  004700  104432                  1$:    LOGR                     ;LOG READ ERROR
  844  004702  104044                         HLT     !D$!DA
  845  004704  104400                  2$:    SCOPE
  846  004706  005237  001200                 INC     LOPCNT           ;COUNT ERRORS
  847  004712  022737  000010  001200         CMP     #10,LOPCNT       ;LAST RETRY?
  848  004720  001323                         BNE     DSKRD            ;NO
  849  004722  004737  006520                 JSR     PC,NOREC         ;TYPE UNRECOVERABLE
  850  004726  000137  004746          4$:    JMP     ADR1             ;CONT
  851  004732  104400                  ADRD:  SCOPE
  852  004734  005737  001200                 TST     LOPCNT           ;ANY ERRORS?
  853  004740  001402                         BEQ     ADR1             ;NO
  854  004742  004737  011772                 JSR     PC,TYPREC        ;TYPE RECOVERED
  855  004746  004737  007050          ADR1:  JSR     PC,DISBUF        ;GO SET UP DISK BUFFER.
  856  004752  000434                         BR      READCT           ;CONT. READING
  857  004754  005737  001126          MSTR:  TST     FLAG
  858  004760  100423                         BMI     3$               ;OPERATOR SELECTED PATTERN
  859  004762  062737  000002  001136         ADD     #2,PATNU         ;INC PATTERN INDEX
  860  004770  022737  000044  001136         CMP     #44,PATNU
  861  004776  001402                         BEQ     .+6
  862  005000  000137  003710                 JMP     DATAT            ;NOT LAST PATTERN EXIT
  863  005004  005037  001136                 CLR     PATNU            ;LAST PATTERN EXIT
  864  005010  032777  000002  174010         BIT     #BIT1,@SWR       ;DATA TEST ONLY?
  865  005016  001006                         BNE     2$               ;YES
  866  005020  032737  002000  001126         BIT     #BIT10,FLAG      ;DATA TEST ONLY?
  867  005026  001404                         BEQ     1$               ;NO
  868  005030  005137  001114          3$:    COM     ASB1             ;ALTERNATE PORTS
  869  005034  000137  006066          2$:    JMP     EXTPPR           ;LOOP
  870  005040  000137  005056          1$:    JMP     RANEX            ;DO NEXT TEST
  871
  872  005044  000137  004524          READCT: JMP    E8H              ;CONT. READING
  873  005050  052737  100000  001216  RANEL: BIS     #BIT15,FLAG3     ;SET LOOP IN RANDOM TEST GOT
  874                                                                   ;HERE BY STARTING AT LOC 240
```

```
                                RANEX:                          ;RANDOM ADDRESS DATA TEST
                                ;THIS PROGRAM WRITES, WRITECHECKS AND READS 1 SECTOR OF RANDOM DATA FROM RANDOM DISK
                                ;ADDRESSES. THIS TEST WILL MAKE 1000(10) PASSES BEFORE IT IS COMPLETED
                                ;
                                ;*****************************************************************
                                ;TEST 3          RANDOM ADDRESS RANDOM DATA TEST
                                ;*****************************************************************
005056  104400                  TST3:   SCOPE
005060  032737  000004  001126          BIT     #BIT2,FLAG      ;FAST XFER MODE?
005066  001402                          BEQ     2$              ;NO
005070  000137  006066                  JMP     EXTPPR          ;GET OUT
005074  052737  000400  001126  2$:     BIS     #BIT8,FLAG      ;SET TEST FLAG
005102  012737  020000  017416          MOV     #20000,OUTBUF   ;GET STARTING ADDR OF BUF
005110  013737  017416  001116          MOV     OUTBUF,VADDR    ;SAVE BUFFER ADDR
005116  005737  001212                  TST     MMAVA           ;MEM MGMT AVAILABLE?
005122  001402                          BEQ     1$              ;NO
005124  005037  177572                  CLR     @#SR0           ;TURN IT OFF
005130  012737  000042  001136  1$:     MOV     #42,PATNU       ;DO RANDOM COMPARE
005136  104426                          CLRDV                   ;INIT DRIVE
005140  012737  176030  001156          MOV     #-1000.,PASSC   ;SET UP PASS COUNT
005146  012737  005706  001152          MOV     #RWRED,HRDER    ;SET UP FOR HARD ERROR
005154  004737  006774                  JSR     PC,VECTRR       ;SETUP INTERRUPT VECTOR
005160  012777  000340  173676          MOV     #340,@STATUS
005166  012737  005270  001010  WRLG1:  MOV     #WRERR,LAD      ;SETUP LOOP ADDRESS
005174  012737  000001  001222          MOV     #1,WORK         ;SET UP RANDOM GENERATOR WORD
005202  013701  017416                  MOV     OUTBUF,R1
005206  004537  010040                  JSR     R5,RANDOM       ;GENERATE RANDOM DATA
005212  017737  012200  001134          MOV     @OUTBUF,DMA     ;SET UP DISK ADDRESS
005220  042737  170000  001134          BIC     #170000,DMA
005226  052737  000003  001126          BIS     #3,FLAG         ;SET COUNTER
005234  004737  006562                  JSR     PC,WNTHU        ;GET WORD COUNT
005240  013737  001130  001222  2$:     MOV     WRDCT,WORK      ;GENERATE RANDOM BUFFER
005246  013701  017416                  MOV     OUTBUF,R1
005252  004537  010040                  JSR     R5,RANDOM
005256  013737  017416  001140          MOV     OUTBUF,BUF      ;SET UP OUTPUT BUFFER
005264  005037  001200                  CLR     LOPCNT          ;CLR ERROR FLAG
```

```
                                WRERR:  ERCLR
                                        MOV     #161,CMD        ;WRITE WITH I/E
                                        DRCMD                   ;WRITE
                                        JSR     PC,WATT         ;WAIT FOR INTERRUPT
                                        BIT     #BIT9,FLAG      ;WAS THERE AN ERROR?
                                        BEQ     WRRCK1          ;NO
                                        BIT     #BIT6,@#WR      ;TYPE RETRY ?
                                        BNE     5$              ;YES
                                        TST     LOPCNT          ;FIRST TIME?
                                        BNE     6$              ;NO
                                5$:     LOGW                    ;LOG WRITE ERROR
                                        INC     LOPCNT          ;SET ERROR FLAG
                                        JSR     PC,PRNT         ;TYPEOUT?
                                        BNE     3$              ;YES
                                        TYPE    ,RANDM
                                        TYPE    ,WRTERR
                                3$:     HLT     ;DS;DA
                                6$:     SCOPE
                                        DEC     FLAG
                                        BIT     #3,FLAG
                                        BNE     WRERR           ;RETRY
                                        JSR     PC,WTNO         ;TYPE CAN NOT WRITE
                                        JMP     EXRAX           ;GET NEW NUMBER
```

```
   934  005410  005737  001200          WRRCK1: TST     LOPCNT          ;ANY ERRORS?
   935  005414  001402                          BEQ     1$              ;NO
   936  005416  004737  011772                  JSR     PC,TYPREC       ;TYPE RECOVERED
   937  005422  104400                  1$:     SCOPE
   938  005424  005037  001200                  CLR     LOPCNT          ;CLEAR LOOP COUNT
   939  005430  104414                  WRRCK:  ERCLR                   ;CLEAR RS REG IF ERRORS

   940  005432  012737  000151  001172          MOV     #151,CMD        ;WRITE CHECK WITH I/E
   941  005440  104416                          DKCMD                   ;WRITE CHECK
   942  005442  004737  011750                  JSR     PC,WATT         ;WAIT FOR INTERRUPT
   943  005446  032737  001000  001126  4$:     BIT     #BIT9,FLAG      ;ERROR?
   944  005454  001453                          BEQ     1$              ;NO
   945  005456  032777  000100  173342          BIT     #BIT6,@#MP      ;TYPE ALL RETRYS?
   946  005464  001003                          BNE     2$              ;YES
   947  005466  005737  001200                  TST     LOPCNT          ;FIRST ERROR?
   948  005472  001030                          BNE     5$              ;NO
   949  005474  104434                  2$:     LOGWC                   ;LOG WRITE CK
   950  005476  004737  014204                  JSR     PC,PRNT         ;TYPEOUT?
   951  005502  001052                          BNE     6$              ;NO
   952  005504  104402  000716                  TYPE    ,RANDM
   953  005510  104402  000574                  TYPE    ,WCKERR
   954  005514  017737  173320  001222          MOV     @RSBA,WORK      ;GET CORRECT BA
   955  005522  162737  000002  001222          SUB     #2,WORK
   956  005530  104402  005534                  TYPE    ,.+2            ;,ASCIZ <15><12>"(BA)="
   957  005544  017746  173452                  MOV     @WORK,-(6)      ;PUT @WORK ON STACK
   958  005550  104404                          TYPEO                   ;TYPE STACK IN OCTAL
   959  005552  104026                          HLT     !DB!DA!BA       ;BA=MEMORY LOC +2 OF ACTUAL WORD
   960  005554  005237  001200          5$:     INC     LOPCNT          ;INC RETRY COUNT
   961  005560  022737  000010  001200          CMP     #10,LOPCNT      ;LAST ONE YET?
   962  005566  001320                          BNE     WRRCK           ;NO
   963  005570  104402  000522                  TYPE    ,UNRECO
   964  005574  005037  001200                  CLR     LOPCNT          ;CLEAR LOPCNT
   965  005600  000137  006010                  JMP     EXRAX           ;GET NEW NUMBER
   966  005604  005737  001200          10$:    TST     LOPCNT          ;ANY ERRORS?
   967  005610  001407                          BEQ     6$              ;NO
   968  005612  104402  000616                  TYPE    ,RECOV
   969  005616  013746  001200                  MOV     LOPCNT,-(6)     ;GET NUMBER
   970  005623  104406                          TYPE8                   ;TYPE IT
   971  005624  104402  000636                  TYPE    ,CRLFLF
   972  005630  104400                  6$:     SCOPE
   973  005632  052737  000003  001126          BIS     #3,FLAG         ;SET COUNTER
```

```
  974  005640  104400                        $SCOPE
  975  005642  005037  001200                CLR     LOPCNT          ;CLEAR COUNTER
  976  005646  004537  011302                JSR     R5,CLEAR        ;CLEAR BUFFER
  977  005652  1044..                RREAD:  ERCLR                   ;CLEAR R5 REG IF ERRORS
  978  005654  012737  000171  001172        MOV     #171,CMD        ;READ WITH I/E
  979  005662  104416                        DKCMD                   ;READ
  980  005664  032777  010000  173134        BIT     #BIT12,@SWR     ;COMPARE ?
  981  005672  001003                        BNE     TAG1            ;NO
  982  005674  004537  010446                JSR     R5,COMPARE      ;YES
  983  005700  000402                        BR      RWRED           ;CONT
  984  005702  004737  011750        TAG1:   JSR     PC,WAIT         ;WAIT FOR INTERRUPT
  985  005706  032737  001000  001126 RWRED: BIT     #BIT9,FLAG      ;IS THERE AN ERROR?
  986  005714  001435                        BEQ     EXRAX           ;NO
  987  005716  104432                        LOGR                    ;LOG READ ERR
  988  005720  032777  000100  173100  18:    BIT     #BIT6,@SWR      ;TYPE ALL ERRORS?
  989  005726  001016                        BNE     2$              ;YES
  990  005730  032777  001000  173070        BIT     #BIT9,@SWR      ;LOOP ON ERROR?
  991  005736  001012                        BNE     2$              ;YES
  992  005740  005737  001200                TST     LOPCNT          ;FIRST ERROR?
  993  005744  001010                        BNE     3$              ;NO
  994  005746  004737  014204                JSR     PC,PRNT         ;TYPEOUT?
  995  005752  001004                        BNE     2$              ;NO
  996  005754  104402  000716                TYPE    ,RANDM
  997  005760  104402  000607                TYPE    ,RDERR
  998  005764  104006                2$:     MLT     1DB1DA
  999  005766  104400                3$:     $SCOPE
 1000  005770  005237  001200                INC     LOPCNT          ;UPDATE COUNTER
 1001  005774  022737  000010  001200        CMP     #10,LOPCNT      ;LAST TRY YET?
 1002  006002  001323                        BNE     RREAD           ;RETRY
 1003  006004  004737  006520                JSR     PC,NOREC        ;TYPE UNRECOVERABLE
 1004  006010  005737  001200        EXRAX:  TST     LOPCNT          ;ANY ERRORS?
 1005  006014  001402                        BEQ     EXRXX           ;NO
 1006  006016  004737  011772                JSR     PC,TYPREC       ;TYPE RICOVERED
 1007  006022  104400                EXRXX:  $SCOPE
 1008  006024  005237  001156                INC     PASSC           ;+1 PASS COUNT
 1009  006030  001402                        BEQ     1$              ;IS TEST DONE?
 1010  006032  000137  005166                JMP     @RLG1           ;NO
 1011  006036  005037  001136        1$:     CLR     PATNU           ;END OF TEST
 1012  006042  042737  000400  001126        BIC     #BIT8,FLAG      ;CLEAR TEST FLAG
 1013  006050  032737  100000  001216        BIT     #BIT15,FLAG3    ;LOOP ON THIS TEST?
 1014  006056  001402                        BEQ     .+6             ;NO
 1015  006060  000137  005056                JMP     RANEX           ;YES
```

```
1016                                    ;CHECK FOR MULTI DISK MODE
1017                                    ;IF IN MULTI DISK MODE REPORT "END"
1018                                    ;IF LAST DISK ON SYSTEM HAS BEEN EXERCISED.
1019
1020                                    ;*****************************************************************
1021                                    ;TEST 4          TEST FOR MULTI DISK MODE
1022                                    ;*****************************************************************
1023  006064  104400                    TST4:   SCOPE
1024  006066  005037  001134            EXTPPR: CLR     DMA
1025  006072  104426                            CLRDV                   ;INIT DRIVE
1026  006074  032737  004000  001126            BIT     #BIT11,FLAG     ;ARE WE IN MULTI DISK MODE
1027  006102  001404                            BEQ     EXTPP           ;NO REPORT "END"
1028  006104  004737  002134                    JSR     PC,TRYNX        ;YES TEST FOR ALL DRIVES
1029  006110  000137  003240                    JMP     EXMFLG          ;RESTART TESTING OF DRIVES
1030  006114  004737  002204            EXTPP:  JSR     PC,DONEE        ;GET PASS COUNT
1031
1032                                    ;THIS ROUTINE CLEARS THE DRIVE
1033                                    ;REGISTERS IF THERE WAS AN ERROR
1034
1035  006120  032737  001000  001126    .ERCLR: BIT     #BIT9,FLAG      ;ANY ERRORS?
1036  006126  001404                            BEQ     1$              ;NO
1037  006130  104426                            CLRDV                   ;CLEAR ALL ERRORS
1038  006132  042737  001000  001126            BIC     #BIT9,FLAG      ;CLEAR ERROR FLAG
1039  006140  000002                    1$:     RTI                     ;EXIT
1040
1041                                    ;ENTER DISK HANDLER BY THE TRAP INSTRUCTION
1042                                    ;ARGUMENT TO TRAP INSTRUCTION IS TWO ORDER
1043                                    ;BYTE OF THE CONTROL REGISTER.
1044
1045  006142  013777  001134  172672    .DKCMD: MOV     DMA,@RSDA       ;LOAD DISK ADD
1046  006150  005037  001176                    CLR     INTFLG          ;CLEAR INTERRUPT FLAG
1047  006154  032737  020000  001122            BIT     #BIT13,FLAG2    ;IN MAPPING ROUTINE?
1048  006162  001021                            BNE     4$              ;YES
1049  006164  032737  000004  001126            BIT     #BIT2,FLAG      ;MAX DATA TEST?
1050  006172  001415                            BEQ     4$              ;NO
1051  006174  022737  000042  001136            CMP     #42,PATNU       ;RANDOM DATA?
1052  006202  001411                            BEQ     4$              ;YES
1053  006204  052777  000010  172622            BIS     #10,@RSCS2      ;SET BAI
1054  006212  005077  172620                    CLR     @RSWC           ;64K XFER
1055  006216  012777  017416  172614            MOV     #OUTBUF,@RSBA   ;SETUP BA
1056  006224  000435                            BR      2$              ;CONT
1057  006226  013777  001140  172604    4$:     MOV     BUF,@RSBA       ;LOAD (CMA) BUSS ADDRESS
1058  006234  013702  001130                    MOV     WRDCT,R2        ;GET NEGATIVE
1059  006240  005402                            NEG     R2              ;WORD COUNT
1060  006242  010277  172570                    MOV     R2,@RSWC        ;LOAD WC
1061  006246  032737  000400  001126            BIT     #BIT8,FLAG      ;RANDOM TEST?
1062  006254  001033                            BNE     1$              ;YES A PORT ONLY WITH NO MEM MGMT
1063  006256  005737  001212                    TST     MMAVA           ;MEM MGMT AVAILIABLE?
1064  006262  001416                            BEQ     2$              ;NO
1065  006264  032737  000040  001122            BIT     #BIT5,FLAG2     ;SET A17 IN RSCS1
1066  006272  001403                            BEQ     3$              ;NO
1067  006274  052737  001000  001172            BIS     #BIT9,CMD       ;YES
```

```
1068  006302  032737  000020  001122  38:     BIT     #BIT4,FLAG2     ;SET A16?
1069  006310  001403                          BEQ     2$              ;NO
1070  006312  052737  000400  001172          BIS     #BIT8,CMD       ;YES
1071  006320  032737  000100  001126  28:     BIT     #BIT6,FLAG      ;MULTI PORT?
1072  006326  001406                          BEQ     1$              ;NO
1073  006330  005737  001114                  TST     ASB1            ;TEST A OR B PORT?
1074  006334  001403                          BEQ     1$              ;A PORT
1075  006336  052737  002000  001172          BIS     #BIT10,CMD      ;B PORT
1076  006344  013777  001172  172460  1$:     MOV     CMD,@RSC$1      ;LOAD FUNCTION REG.
1077  006352  000002                          RTI                     ;RETURN FROM TRAP
1078
1079                                  ;RH11 DISK INTERRUPT HANDLER
1080                                  ;ROUTINE CONTINUES ON ERRORS
1081
1082  006354  042737  001000  001126  DKINT:  BIC     #BIT9,FLAG      ;CLEAR ERROR BIT
1083  006362  005777  172444                  TST     @RSC$1          ;TEST FOR ERROR
1084  006366  100401                          BMI     2$
1085  006370  000425                          BR      INTEXT          ;JUMP IF NO ERRORS
1086  006372  017702  172434          2$:     MOV     @RSC$1,R2       ;GET CONTENTS OF C$1
1087  006376  042702  037777                  BIC     #37777,R2       ;CLEAR ALL BUT SC AND TRE
1088  006402  022702  140000                  CMP     #140000,R2      ;IS SC AND TRE BOTH SET?
1089  006406  001413                          BEQ     TRUERR          ;YES THERE IS SOME KIND OF XFER ERROR
1090  006410  032777  100000  172426          BIT     #100000,@RSDS   ;IS THE ATA BIT SET?
1091  006416  001007                          BNE     TRUERR          ;YES
1092  006420  104140                          HLT     1A$1D$          ;WRONG UNIT INTERRUPTED
1093                                                                  ;IF YOU HAVE JUST POWERED UP A DRIVE OR
1094                                                                  ;ARE RUNNING THE POWER FAIL TEST,
1095                                                                  ;INTERRUPTS WILL OCCUR FROM DRIVES OTHER
1096                                                                  ;THAN THE UNIT UNDER TEST. IF THIS TYPEOUT
1097                                                                  ;SHOWS NO ERRORS IN THE REGISTERS OF THE DRIVE
1098                                                                  ;UNDER TEST, THAT DRIVE IS OK
1099  006422  012777  177777  172420  1$:     MOV     #-1,@RSAS       ;CLEAR ALL ATA BITS
1100  006430  013716  001152                  MOV     NRDER,(SP)      ;GET RETURN ADD.
1101  006434  000002                          RTI                     ;RETRY
1102  006436  052737  001000  001126  TRUERR: BIS     #BIT9,FLAG      ;SET ERROR BIT
1103  006444  032777  004000  172354  INTEXT: BIT     #BIT11,@SWR     ;HALT ON COMPLETION FLAG
1104  006452  001401                          BEQ     .+4
1105  006454  000000                          HALT                    ;YES BIT 11 SET IN SWR HALT
1106  006456  032737  002000  001122          BIT     #BIT10,FLAG2    ;WAIT IN BACKGROUND TEST?
1107  006464  001402                          BEQ     1$              ;NO
1108  006466  012716  012416          2$:     MOV     #NPRRET,(SP)    ;MODIFY RETURN ADD.
1109  006472  010637  001176          1$:     MOV     SP,INTFLG       ;SET INT FLG
1110  006476  000002                          RTI                     ;EXIT
1111
1112                                  ;ROUTINE TO SET UP TRACK # FROM OPTION
1113                                  ;ENTER FROM JSR R5, OPDSEL
1114
1115  006500  032737  000040  001126  OPDSEL: BIT     #BIT5,FLAG      ;OPTIONAL DMA?
1116  006506  001403                          BEQ     1$              ;NO
1117  006510  013737  001142  001134          MOV     TDMA,DMA        ;GET OPT. DMA
1118  006516  000205                  1$:     RTS     R5              ;EXIT
```

```
1119
1120
1121                                        .EVEN
1122
1123  006520  004737  014204            NOREC:  JSR     PC,PRNT         ;TYPEOUT?
1124  006524  001002                            BNE     1$              ;NO
1125  006526  104402  000522                    TYPE    ,UNRECO
1126  006532  005037  001200            1$:     CLR     LOPCNT          ;CLEAR LOOP COUNTER
1127  006536  000207                            RTS     PC
1128
1129  006540  052737  040000  001122    FNDTYP: BIS     #BIT14,FLAG2    ;SET CHECK DRIVE TYPE FLAG
1130  006546  004737  001746                    JSR     PC,UNTYP        ;CHECK DRIVE TYPE FLAG
1131  006552  042737  040000  001122            BIC     #BIT14,FLAG2    ;CLEAR DRIVE TYPE FLAG
1132  006560  000207                            RTS     PC
1133  006562  022737  000004  001166    WHTHU:  CMP     #4,RS04DT       ;RS03LA?
1134  006570  001004                            BNE     1$              ;NO
1135  006572  012737  000040  001130            MOV     #40,WRDCT       ;GET WORD COUNT
1136  006600  000411                            BR      2$
1137  006602  012737  000200  001130    1$:     MOV     #200,WRDCT      ;RS04
1138  006610  005737  001166                    TST     RS04DT          ;RS04?
1139  006614  001003                            BNE     2$              ;YES
1140  006616  012737  000100  001130            MOV     #100,WRDCT      ;NO
1141  006624  000207                    2$:     RTS     PC
1142
1143
```

```
1144                                        ;ROUTINE TO CALCULATE VITURAL ADDR
1145
1146  006626  000302                PHYCOV: SWAB    R2              ;CALCULATE FROM PHYSICAL ADDR
1147  006630  004737  011736                JSR     PC,PRR2
1148  006634  006002                        ROR     R2
1149  006636  042702  177770                BIC     #177770,R2      ;GET REG #
1150  006642  032777  000400  172162        BIT     #BIT8,@RSCS1    ;IS A16 SET?
1151  006650  001402                        BEQ     1$              ;NO
1152  006652  052702  000010                BIS     #BIT3,R2        ;YES
1153  006656  032777  001000  172146  1$:   BIT     #BIT9,@RSCS1    ;IS A17 SET?
1154  006664  001402                        BEQ     2$              ;NO
1155  006666  052702  000020                BIS     #BIT4,R2        ;YES
1156  006672  013737  001070  001224  2$:   MOV     STAMEM,WORK1    ;GET BANK # FOR -A- PORT
1157  006700  005737  001114                TST     A0B1            ;ARE WE ON -A- PORT?
1158  006704  001403                        BEQ     3$              ;YES
1159  006706  013737  001074  001224        MOV     STBCOM,WORK1    ;NO -B- PORT
1160  006714  163702  001224          3$:   SUB     WORK1,R2        ;GET STARTING BANK #
1161  006720  062702  000001                ADD     #1,R2           ;GET OFFSET FOR REG #
1162  006724  000302                        SWAB    R2              ;GET BANK # INTO
1163  006726  006102                        ROL     R2              ;UPPER BITS
1164  006730  006102                        ROL     R2
1165  006732  006102                        ROL     R2
1166  006734  006102                        ROL     R2
1167  006736  006102                        ROL     R2
1168  006740  017737  172074  001224        MOV     @RSDA,WORK1     ;GET OFFSET FOR ADDR IF ANY
1169  006746  162737  000002  001224        SUB     #2,WORK1        ;CORRECT IT
1170  006754  042737  160000  001224        BIC     #160000,WORK1   ;CLEAR JUNK
1171  006762  050237  001224                BIS     R2,WORK1        ;GET REG NO
1172  006766  013702  001224                MOV     WORK1,R2
1173  006772  000207                        RTS     PC
1174
1175  006774  012777  006354  172060  VECTRR: MOV     #DKINT,@RSVEC   ;SETUP INTERRUPT VECTORS
1176  007002  013737  001066  177776        MOV     PRIORITY,PS     ;PRIORITY 4
1177  007010  000207                        RTS     PC
1178
1179                                      ;THIS ROUTINE IS USED FOR DELAYING THE START OF THIS PROGRAM
1180                                      ;IF POWER FAILED DURING TESTING, THIS WILL GIVE THE DRIVES TIME TO GET UP
1181                                      ;TO SPEED, THE DELAY WILL BE ABOUT 3-5 MINUTES DEPENDING UPON THE PROCESSOR
1182
1183  007012  012737  000677  001222  TIMUP:  MOV     #677,WORK
1184  007020  012737  177777  001224  1$:     MOV     #177777,WORK1
1185  007026  000240                2$:     NOP
1186  007030  005337  001224                DEC     WORK1
1187  007034  001374                        BNE     2$
1188  007036  005337  001222                DEC     WORK
1189  007042  001366                        BNE     1$
1190  007044  000137  003224                JMP     ADTST
```

```
1191                                    ;ROUTINE TO SETUP DISK BUFFERS
1192                                    ;ADD WORD COUNT TO STARTING DISK ADDRESSES
1193                                    ;COMPARE CALCULATED ADDRESS TO TERMINATING ADDRESS
1194
1195  007050  032737  000040  001126  DISBUF: BIT     #BIT5,FLAG      ;DID OPERATOR SELECT PATTERNS
1196  007056  001402                          BEQ     2$              ;NO

1197  007060  000137  007342                  JMP     BUFEXIT         ;YES
1198  007064  022737  000042  001136  2$:     CMP     #42,PATNU       ;RANDOM PATTERN?
1199  007072  001443                          BEQ     1$              ;YES
1200  007074  032737  000004  001126          BIT     #BIT2,FLAG      ;MAX TST?
1201  007102  001437                          BEQ     1$              ;NO
1202  007104  022737  000004  001166          CMP     #4,RS04DT       ;RS03LA?
1203  007112  001010                          BNE     4$              ;NO
1204  007114  022737  004000  001134          CMP     #4000,DMA       ;DONE YET?
1205  007122  001507                          BEQ     BUFEXIT         ;YES
1206  007124  062737  004000  001134          ADD     #4000,DMA       ;UPDATE DMA
1207  007132  000207                          RTS     PC
1208  007134  005737  001166          4$:     TST     RS04DT          ;RS04?
1209  007140  001010                          BNE     3$              ;YES
1210  007142  022737  006000  001134          CMP     #6000,DMA       ;RS03
1211  007150  001474                          BEQ     BUFEXIT         ;DONE GET OUT
1212  007152  062737  002000  001134          ADD     #2000,DMA       ;UPDATE DMA
1213  007160  000207                          RTS     PC              ;EXIT
1214  007162  022737  007000  001134  3$:     CMP     #7000,DMA       ;DONE YET?
1215  007170  001464                          BEQ     BUFEXIT         ;YES
1216  007172  062737  001000  001134          ADD     #1000,DMA       ;UPDATE ADDR
1217  007200  000207                          RTS     PC
1218  007202  004737  007556          1$:     JSR     PC,BLSZ         ;DEFINE BLOCK SIZE
1219  007206  013737  001154  001224          MOV     BLOCK,WORK;
1220  007214  005237  001134          INCSEC: INC     DMA             ;+1 SECTOR COUNT
1221  007220  022737  010000  001134          CMP     #10000,DMA      ;DONE YET?
1222  007226  001445                          BEQ     BUFEXIT         ;YES
1223  007230  005337  001154                  DEC     BLOCK           ;-1 FROM BLOCK COUNT
1224  007234  001401                          BEQ     COMDAR          ;CMP DMA TO RSDA
1225  007236  000766                          BR      INCSEC          ;RECYCLE
1226  007240  032737  001000  001126  COMDAR: BIT     #BIT9,FLAG      ;ANY ERRORS?
1227  007246  001401                          BEQ     1$              ;NO ERRORS DO COMPARE ON RSDA
1228  007250  000207                          RTS     PC              ;ERRORS DO NOT COMPARE RSDA
1229  007252  023777  001134  171562  1$:     CMP     DMA,@RSDA       ;COMPARE RSDA WITH DMA
1230  007260  001425                          BEQ     CMDAE           ;SHOULD BE EQUAL
1231  007262  104432                          LOGR                    ;AFTER TRANSFER RSDA AND DMA SHOULD BE =
1232                                          ;IF NOT, RSDA IS NOT CORRECT, DMA CONTAINS
1233                                          ;WHAT RSDA SHOULD =
1234  007264  013781  001134                  MOV     DMA,GOOD        ;GET DMA FOR CORRECT ANS IN GOOD
1235  007270  017700  171546                  MOV     @RSDA,BAD       ;GET RSDA INTO BAD
1236  007274  104000                          HLT                     ;RSDA=BAD DMA=GOOD SEE COMMENTS 7 LINES ABOVE
1237  007276  004737  014204                  JSR     PC,PRNT         ;TYPEOUT?
1238  007302  001014                          BNE     CMDAE           ;NO
1239  007304  011637  001222                  MOV     (SP),WORK       ;GET TEST PC FROM WHERE IT CAME
1240  007310  104402  007314                  TYPE    .,.+2           ;,ASCIZ " TST PC="
1241  007316  013746  001222                  MOV     WORK,-(6)       ;PUT WORK ON STACK
1242  007322  104406                          TYPOS                   ;TYPE STACK IN OCTAL - SUPRESS
1243  007324  105737  001126          CMDAE:  TSTB    FLAG            ;LAST DISK BUFFER?
1244  007330  100032                          BPL     BUFINX          ;NO
```

```
1245  007342  005037  001134          BUFEXIT:CLR     DMA             ;CLEAR ADDRESS BITS LAST DISK BUFFER
1246  007346  062716  000002                  ADD     #2,(6)          ;INC STOCK POINTER
1247  007352  042737  000200  001126  AKH:    BIC     #200,FLAG       ;CLEAR LAST DISK BUFFER FLAG
1248  007360  032737  000400  001126          BIT     #BIT8,FLAG      ;RANDOM TEST OR ADDR TEST?
1249  007366  001404                          BEQ     1$              ;NO
1250  007370  013737  001144  001130  2$:     MOV     $WRDCT,WRDCT
1251  007376  000466                          BR      EXTDR           ;EXIT
1252  007400  032737  000100  001126  1$:     BIT     #BIT6,FLAG      ;MULTI PORT?
1253  007406  001770                          BEQ     2$              ;NO
1254  007410  005737  001114                  TST     A&B1            ;A OR B PORT?
1255  007414  001765                          BEQ     2$              ;A PORT
1256  007416  013737  001112  001130          MOV     WDCTB,WRDCT     ;B PORT
1257  007424  000453                          BR      EXTDR           ;GET OUT
1258  007426  005037  001226          BUFINX: CLR     WORK2           ;CLEAR WORK2 FOR BLOCK COUNTER
1259  007432  013702  001134                  MOV     DMA,R2          ;PUT WORKING DISK ADD INTO WORK
1260  007436  005237  001226          XINCSEC:INC     WORK2           ;INCREMENT BLOCK COUNT
1261  007442  022702  007777                  CMP     #7777,R2        ;CMP FOR LAST SECTOR
1262  007446  001405                          BEQ     XINCSUR         ;+1 SURFACE LAST SECTOR BRANCH
1263  007450  005202                          INC     R2              ;INC DMA
1264  007452  005337  001224                  DEC     WORK1           ;DEC BLOCK COUNT
1265  007456  001367                          BNE     XINCSEC         ;FILLED STANDARD BUFFER YET?
1266  007460  000734                          BR      AKH             ;WILL TAKE STANDARD SIZE WORD COUNT
1267  007462  013737  001226  001130  XINCSUR:MOV     WORK2,WRDCT     ;SETTING UP BLOCK COUNT
1268  007470  000241                          CLC                     ;FOR NON STANDARD BUFFER SIZE
1269  007472  006137  001130                  ROL     WRDCT
1270  007476  006137  001130                  ROL     WRDCT
1271  007502  006137  001130                  ROL     WRDCT
1272  007506  006137  001130                  ROL     WRDCT
1273  007512  006137  001130                  ROL     WRDCT
1274  007516  022737  000004  001166          CMP     #4,RS04DT       ;RS03LA?
1275  007524  001410                          BEQ     1$              ;YES
1276  007526  000241                          CLC
1277  007530  006137  001130                  ROL     WRDCT
1278  007534  005737  001166                  TST     RS04DT          ;RS04?
1279  007540  001402                          BEQ     1$              ;NO
1280  007542  006137  001130                  ROL     WRDCT           ;YES
1281  007546  052737  000200  001126  1$:     BIS     #200,FLAG       ;SET LAST DISK BUFFER FLAG
1282  007554  000207                  EXTDR:  RTS     PC              ;EXIT
```

```
1283                                    ;THIS ROUTINE CONVERTS A WORD COUNT TO A BLOCK COUNT
1284  007556  022737  000004  001166  BLSI:   CMP     #4,RS04DT       ;RS03LA
1285  007564  001004                          BNE     3$              ;NO
1286  007566  012737  000037  001154          MOV     #37,BLOCK       ;YES
1287  007574  000411                          BR      2$              ;CONTINUE
1288  007576  012737  000177  001154  3$:     MOV     #177,BLOCK      ;SETUP FOR RS04
1289  007604  005737  001166                  TST     RS04DT          ;RS04?
1290  007610  001003                          BNE     2$              ;YES
1291  007612  012737  000077  001154  1$:     MOV     #77,BLOCK       ;PUT SECTOR SIZE INTO BLOCK
1292  007620  013702  001130  2$:     MOV     WRDCT,R2        ;FETCH WORD COUNT
1293  007624  033702  001154          BIT     BLOCK,R2        ;ARE THEY EQUAL?
1294  007630  001406                          BEQ     RORBLK          ;YES
1295  007632  043702  001154                  BIC     BLOCK,R2        ;SET UP BLOCK OVERFLOW
1296  007636  005237  001154                  INC     BLOCK
1297  007642  063702  001154                  ADD     BLOCK,R2
1298  007646  000241                  RORBLK: CLC
1299  007650  006002                          ROR     R2
1300  007652  022737  000004  001166          CMP     #4,RS04DT       ;RS03LA
1301  007660  001003                          BNE     2$              ;NO
1302  007662  004737  011736                  JSR     PC,RRR2         ;YES
1303  007666  000410                          BR      1$              ;CONTINUE
1304  007670  000241                  2$:     CLC
1305  007672  006002                          ROR     R2
1306  007674  004737  011736                  JSR     PC,RRR2
1307  007700  005737  001166                  TST     RS04DT          ;RS#4?
1308  007704  001401                          BEQ     1$              ;NO
1309  007706  006002                          ROR     R2              ;YES
1310  007710  010237  001154          1$:     MOV     R2,BLOCK        ;BLOCK COUNT
1311  007714  000207                          RTS     PC              ;EXIT
1312
1313                                    ;ROUTINE TO SELECT DATA PATTERNS FOR TEST
1314                                    ;ENTER FROM JSR R5,PASEL
1315  007716  012737  010352  000004  PASEL:  MOV     #MEM,@#4        ;SETUP TRAP
1316  007724  012737  000340  000006          MOV     #340,@#6        ;VECTOR
1317  007732  013700  001136                  MOV     PATNU,R0        ;SET UP PATTERN NUMBER
1318  007736  010003                          MOV     R0,R3           ;GET PATTERN #
1319  007740  000241                          CLC                     ;MAKE IT #
1320  007742  006003                          ROR     R3              ;TO PATTERN # IN LISTING
1321  007744  010377  171060                  MOV     R3,@DISPLAY     ;DISPLAY PATTERN #
1322  007750  013737  001130  001222          MOV     WRDCT,WORK      ;SET UP WORK
1323  007756  013701  001116                  MOV     VADDR,R1        ;LOC. OF OUTBUFFER
1324  007762  022700  000042          1$:     CMP     #42,R0          ;TEST FOR RANDOM DATA NUMBER
1325  007766  001424                          BEQ     RANDOM          ;GO GENERATE RANDOM DATA
1326  007770  032737  000004  001126          BIT     #BIT2,FLAG      ;MAX TST?
1327  007776  001404                          BEQ     2$              ;NO
1328  010000  016037  000254  017416          MOV     PAT0(0),OUTBUF  ;GET PATTERN
1329  010006  000205                          RTS     R5
1330  010010  016000  000254          2$:     MOV     PAT0(0),R0
1331  010014  010021                  FILDAT: MOV     R0,(1)+         ;FILL BUFFER
1332  010016  005337  001222                  DEC     WORK            ;DEC. WORK COUNT
1333  010022  001374                          BNE     FILDAT          ;LOAD NEXT WORD
1334  010024  012737  000006  000004  PASEX:  MOV     #6,@#4          ;RESTORE
1335  010032  005037  000006                  CLR     @#6             ;TRAP
1336  010036  000205                          RTS     R5              ;BUFFER FULL
```

```
1337                                        ;RANDOM DATA GENERATOR SUBROUTINE
1338
1339  010040  013737  010204  010210  RANDOM: MOV     LONUM,LOSAV
1340  010046  013737  010206  010212          MOV     HINUM,HISAV
1341  010054  013700  010204          RAND1:  MOV     LONUM,R0        ;SET UP R0 WITH 5 DIGITS LOW
1342  010060  013704  010206                  MOV     HINUM,R4        ;SET UP R1 WITH 5 DIGITS HIGH
1343  010064  012703  000007                  MOV     #7,R3           ;SET UP SHIFT COUNT
1344  010070  005002                          CLR     R2              ;CLEAR R2
1345  010072  006300                  SHIFT:  ASL     R0              ;SHIFT R0 LEFT AND
1346  010074  006104                          ROL     R4              ;ROTATE CARRY INTO LSB OF R1 INTO
1347  010076  006102                          ROL     R2              ;ROTATE CARRY OUT OF R1 INTO R2
1348  010100  005303                          DEC     R3              ;DECREMENT R3
1349  010102  001373                          BNE     SHIFT           ;CONTINUE SHIFT LOOP
1350  010104  063700  010204                  ADD     LONUM,R0        ;ADDN IN NUMBER TO MAKE X 129
1351  010110  005504                          ADC     R4              ;PROPOGATE CARRY
1352  010112  063704  010206                  ADD     HINUM,R4        ;ADDN IN NUMBER TO MAKE X 129
1353  010116  005502                          ADC     R2              ;PROPOGATE CARRY
1354  010120  062700  001057                  ADD     #1057,R0        ;ADDN LOW CONSTANT
1355  010124  005504                          ADC     R4              ;PROPOGATE CARRIES
1356  010126  005502                          ADC     R2              ;PROPOGATE AGAIN
1357  010130  062704  047401                  ADD     #47401,R4       ;ADDN HIGH CONSTANT
1358  010134  005502                          ADC     R2              ;PROPOGATE CARRY
1359  010136  062702  000006                  ADD     #6,R2           ;ADDN HIGHEST CONSTANT
1360  010142  062700  000002                  ADD     #2,R0           ;REPRIME R0 WITH HIGH DIGIT
1361  010146  005504                          ADC     R4              ;PROPOGATE CARRY
1362  010150  010037  010204                  MOV     R0,LONUM        ;PUT R0 BACK IN LONUM
1363  010154  010021                          MOV     R0,(1)+         ;HOLD LONUM FOR PROGRAM
1364  010156  005337  001222                  DEC     WORK
1365  010162  001406                          BEQ     EXGEN
1366  010164  010437  010206                  MOV     R4,HINUM        ;PUT R1 BACK IN HINUM
1367  010170  010421                          MOV     R4,(1)+         ;HOLD HINUM FOR PROGRAM
1368  010172  005337  001222                  DEC     WORK
1369  010176  001326                          BNE     RAND1
1370  010200  000137  010024          EXGEN:  JMP     PASEX           ;RETURN TO PROGRAM
1371  010204  000000                  LONUM:  0
1372  010206  000000                  HINUM:  0
1373  010210  000000                  LOSAV:  0
1374  010212  000000                  HISAV:  0
1375
1376  010214  013737  001214  001144  RESTOR: MOV     SAVWC,SWRDCT    ;RESTORE ORIGINAL
1377  010222  013737  001144  001130          MOV     SWRDCT,WRDCT    ;WORD COUNT
1378  010230  013737  001220  001112          MOV     SAVWCB,WDCTB
1379  010236  000205                          RTS     R5
```

```
1380                                    ;RANDOM DATA GENERATOR SUBROUTINE
1381                                    ;WHEN SWITCH = 0 WE COME HERE
1382
1383  010240  013700  010346    RAND:   MOV     LONUM1,R0       ;SET UP R0 WITH 5 DIGITS LOW
1384  010244  013704  010350            MOV     HINUM1,R4       ;SET UP R1 WITH 5 DIGITS HIGH
1385  010250  012703  000007            MOV     #7,R3           ;SET UP SHIFT COUNT

1386  010254  005002                    CLR     R2              ;CLEAR R2
1387  010256  006300            SHIFT1: ASL     R0              ;SHIFT R0 LEFT AND
1388  010260  006104                    ROL     R4              ;ROTATE CARRY INTO LSB OF R1 INTO
1389  010262  006102                    ROL     R2              ;ROTATE CARRY OUT OF R1 INTO R2
1390  010264  005303                    DEC     R3              ;DECREMENT R3
1391  010266  001373                    BNE     SHIFT1          ;CONTINUE SHIFT LOOP
1392  010270  063700  010346            ADD     LONUM1,R0       ;ADDN IN NUMBER TO MAKE X 129
1393  010274  005504                    ADC     R4              ;PROPOGATE CARRY
1394  010276  063704  010350            ADD     HINUM1,R4       ;ADDN IN NUMBER TO MAKE X 129
1395  010302  005502                    ADC     R2              ;PROPOGATE CARRY
1396  010304  062700  001057            ADD     #1057,R0        ;ADDN LOW CONSTANT
1397  010310  005504                    ADC     R4              ;PROPOGATE CARRIES
1398  010312  005502                    ADC     R2              ;PROPOGATE AGAIN
1399  010314  062704  047401            ADD     #47401,R4       ;ADDN HIGH CONSTANT
1400  010320  005502                    ADC     R2              ;PROPOGATE CARRY
1401  010322  062702  000006            ADD     #6,R2           ;ADDN HIGHEST CONSTANT
1402  010326  062700  000002            ADD     #2,R0           ;REPRIME R0 WITH HIGH DIGIT
1403  010332  005504                    ADC     R4              ;PROPOGATE CARRY
1404  010334  010037  010346            MOV     R0,LONUM1       ;PUT R0 BACK IN LONUM
1405  010340  010437  010350            MOV     R4,HINUM1       ;PUT R1 BACK IN HINUM
1406  010344  000205            EXGEN1: RTS     R5              ;RETURN TO PROGRAM
1407  010346  000000            LONUM1: 0
1408  010350  000000            HINUM1: 0
1409
1410                                    ;TRAP OUT ROUTINE WHEN CREATING DATA BUFFER
1411
1412  010352                    MEM:
1413  010352  104402  010356            TYPE    ,+2             ;.ASCIZ <15><12>"NIM PORT "
1414  010372  005737  001114            TST     A001            ;FIND WHAT DATA BUFFER
1415  010376  001004                    BNE     38              ;BRANCH IF B
1416  010400  104402  010404            TYPE    ,+2             ;.ASCIZ "A"
1417  010406  000403                    BR      48
1418  010410                    38:
1419  010410  104402  010414            TYPE    ,+2             ;.ASCIZ "B"
1420  010416  012737  000006  000004    48:     MOV     #6,@#4          ;RESTORE
1421  010424  005037  000006            CLR     @#6             ;TRAP
1422  010430  032777  100000  170370            BIT     #BIT15,@#PSWR   ;HALT?
1423  010436  001401                    BEQ     28              ;NO
1424  010440  000000                    HALT
1425  010442  000137  001234    28:     JMP     @#BEGIN
```

```
1426                                    ;THIS ROUTINE COMPARES THE DATA READ AGAINST THE DATA EXPECTED.
1427                                    ;ALL ERRORS ARE REPORTED TO THE OPERATOR. IF BIT 4 OF THE SWITCH
1428                                    ;REGISTER IS SET, THIS ROUTINE WILL CONTINUE COMPARING AFTER AN ERROR HAS BEEN
1429                                    ;FOUND AND WILL REPORT UP TO 8 VERIFY ERRORS WITHIN THE SAME INPUT OPERATION.
1430
1431  010446  012737  177770  001146  COMPAR: MOV     #-10,ERCOUNT    ;ERROR RETRY COUNTER
1432  010454  052737  000010  001122          BIS     #BIT3,FLAG2     ;DOING COMPARE
1433  010462  013737  001130  001226          MOV     WRDCT,WORK2     ;GET THE WORD COUNT
1434  010470  013737  001116  001150          MOV     VADDR,SAVE      ;SET UP OUTBUFFER POINTER
1435  010476  005037  001174                  CLR     SWITCH          ;CLEAR RANDOM PATTERN FLAG
1436  010502  013737  010210  010346          MOV     LOSAV,LONUM1    ;GET RANDOM BASE NOS.
1437  010510  013737  010212  010350          MOV     HISAV,HINUM1
1438  010516  005737  001136                  TST     PATNU           ;TEST FOR PATTERN 0
1439  010522  001015                          BNE     1$              ;NO
1440  010524  005037  001222                  CLR     WORK            ;CLEAR COUNTER
1441  010530  062737  000001  001222  2$:     ADD     #1,WORK         ;INC COUNTER
1442  010536  001003                          BNE     3$              ;INTERRUPT YET?
1443  010540  104054                          HLT     ;DA;NC;DS       ;TIMED OUT NOT INTERRUPT
1444  010542  000137  001234                  JMP     @#BEGIN
1445  010546  005737  001176          3$:     TST     INTFLG          ;TEST FOR INT
1446  010552  001766                          BEQ     2$              ;WAIT FOR INT BEFORE COMPARING
1447  010554  000426                          BR      CMPLP1          ;CONT
1448  010556  022737  0( 742  001136  1$:     CMP     #42,PATNU       ;IS THIS RANDOM PATTERN?
1449  010564  001022                          BNE     CMPLP1          ;BRANCH IF YES
1450  010566  005737  00. 76          CMPLP:  TST     INTFLG          ;INTERRUPT YET?
1451  010572  001775                          BEQ     CMPLP           ;NO WAIT
1452  010574  005737  0011 4                  TST     SWITCH
1453  010600  001007                          BNE     2$
1454  010602  004537  010240                  JSR     R5,RAND
1455  010606  013701  010346                  MOV     LONUM1,GOOD     ;GET EVEN RANDOM WORD
1456  010612  010637  001174                  MOV     SP,SWITCH       ;SET RANDOM PATTERN FLAG
1457  010616  000411                          BR      WRDCMP
1458  010620  005037  001174          2$:     CLR     SWITCH
1459  010624  013701  010350                  MOV     HINUM1,GOOD
1460  010630  000404                          BR      WRDCMP
1461  010632  013700  001136          CMPLP1: MOV     PATNU,R0
1462  010636  016001  000254                  MOV     PAT0(R0),GOOD
1463  010642  160177  170302          WRDCMP: SUB     GOOD,@SAVE      ;COMPARE DATA
1464  010646  001017                          BNE     WDERR           ;WORD IN ERROR
1465  010650  005337  001226          WRDINC: DEC     WORK2           ;DECREMENT THE WORD COUNT
1466  010654  001410                          BEQ     ADAM            ;EXIT ROUTINE IF ZERO
1467  010656  062737  000002  001150          ADD     #2,SAVE         ;UPDATE PATTERN ADDRESS
1468  010664  022737  000042  001136          CMP     #42,PATNU       ;IS THIS RANDOM PATTERN
1469  010672  001735                          BEQ     CMPLP           ;BRANCH IF YES
1470  010674  000762                          BR      WRDCMP          ;COMPARE NEXT WORD
1471  010676  042737  000010  001122  ADAM:   BIC     #BIT3,FLAG2     ;DONE WITH COMPARE
1472  010704  000205                          RTS     R5              ;EXIT THIS ROUTINE
```

```
1473  010706  005737  001176          WDERR:  TST     INTFLG          ;DID INTERRUPT OCCUR YET?
1474  010712  001753                          BEQ     WRDCMP          ;BRANCH IF NO
1475  010714  032777  000100  170104          BIT     #BIT6,@SWR      ;TRY ALL?
1476  010722  001006                          BNE     10$             ;YES
1477  010724  005737  001200                  TST     LOPCNT          ;FIRST READ ERROR?
1478  010730  001403                          BEQ     10$             ;YES
1479  010732  005777  170074                  TST     @RSCS1          ;ANY ERRORS?
1480  010736  100757                          BMI     ADAM            ;YES DO NOT COMPARE
1481  010740  060177  170204          10$:    ADD     GOOD,@SAVE
1482  010744  017700  170200                  MOV     @SAVE,BAD       ;GET GOOD DATA
1483  010750  104436                          LOGC                    ;LOG COMPARE ERROR
1484  010752  032777  001000  170046          BIT     #BIT9,@SWR      ;LOOP ON ERROR?
1485  010760  001401                          BEQ     11$             ;NO
1486  010762  005726                          TST     (6)+            ;YES UPDATE SP
1487  010764  004737  014204          11$:    JSR     PC,PRNT         ;TYPEOUT?
1488  010770  001007                          BNE     3$              ;NO
1489  010772  104402  010776                  TYPE    ..+2            ;,ASCIZ <15><12>"CMP ERR"
1490  011010  104000                  3$:     HLT                     ;DATA COMPARE ERROR
1491  011012  004737  014204                  JSR     PC,PRNT         ;HAD TO DO IT THIS WAY SO
1492  011016  001022                          BNE     13$             ;PROGRAM COULD LOOP ON ERROR
1493  011020  104402  011024                  TYPE    ..+2            ;,ASCIZ " ADDR="
1494  011034  005737  001212                  TST     MMAVA           ;IS MEM MGMT ON?
1495  011040  001406                          BEQ     12$             ;NO
1496  011042  013746  177776                  MOV     PS,-(6)         ;GET PS
1497  011046  013746  001150                  MOV     SAVE,-(6)       ;GET VIRTUAL ADDR
1498  011052  104412                          TYPEA                   ;CONVERT TO PHY AND TYPE
1499  011054  000403                          BR      13$             ;CONT
1500  011056  013746  001150          12$:    MOV     SAVE,-(6)       ;GET ADDR
1501  011062  104406                          TYPE8                   ;TYPE IT
1502  011064  005037  001154          13$:    CLR     BLOCK           ;CLEAR THE BLOCK COUNTER
1503  011070  013702  001130                  MOV     WRDCT,R2        ;GET THE WORD COUNT
1504  011074  005202                          INC     R2              ;CORRECT FOR DA CALCULATIONS
1505  011076  163702  001226                  SUB     WORK2,R2        ;DETERMINE DISTANCE OF FAILURE INTO BUFFER
1506  011102  022737  000004  001166  2$:     CMP     #4,RS04DT       ;RS03LA?
1507  011110  001003                          BNE     14$             ;NO
1508  011112  162702  000040                  SUB     #40,R2
1509  011116  000410                          BR      9$              ;CONTINUE
1510  011120  005737  001166          14$:    TST     RS04DT          ;RS04?
1511  011124  001403                          BEQ     7$              ;NO
1512  011126  162702  000200                  SUB     #200,R2         ;RS03
1513  011132  000402                          BR      9$              ;CONT
1514  011134  162702  000100          7$:     SUB     #100,R2
1515  011140  100403                  9$:     BMI     8$
1516  011142  005237  001154                  INC     BLOCK           ;UPDATE BLOCK COUNT FOR EACH 400 WORDS
1517  011146  000755                          BR      2$
```

```
1518  011150  022737  000004  001166  001$:   CMP     #4,RS04DT       ;RS04LA
1519  011156  001003                          BNE     15$             ;NO
1520  011160  062702  000040                  ADD     #40,R2          ;RESTORE POSITIVE 5
1521  011164  000410                          BR      6$              ;CONTINUE
1522  011166  005737  001166          15$:    TST     RS04DT          ;RS04?
1523  011172  001403                          BEQ     4$              ;NO
1524  011174  062702  000200                  ADD     #200,R2         ;RS04
1525  011200  000402                          BR      6$              ;CONT
1526  011202  062702  000100          4$:     ADD     #100,R2         ;RESTORE POSITIVE NUMBER
1527  011206  013737  001134  001224  6$:     MOV     DMA,WORK1       ;GET HEAD AND SECTOR ADDRESS
1528  011214  063737  001154  001224  5$:     ADD     BLOCK,WORK1
1529  011222  004737  014204                  JSR     PC,PRNT         ;TYPEOUT?
1530  011226  001014                          BNE     1$              ;NO
1531  011230  104402  011234                  TYPE    ,.+2            ;.ASCIZ " DA="
1532  011242  013746  001224                  MOV     WORK1,-(6)      ;PUT WORK1 ON STACK
1533  011246  104406                          TYPE8                   ;TYPE STACK IN OCTAL - SUPRESS
1534  011250  104402  011254                  TYPE    ,.+2            ;.ASCIZ <15><12>
1535  011260  032777  000020  167540  1$:     BIT     #BIT4,@SWR      ;RETRY?
1536  011266  001405                          BEQ     CLEAR           ;NO
1537  011270  005237  001146                  INC     ERCOUNT         ;UPDATE ERROR COUNTER
1538  011274  001402                          BEQ     CLEAR
1539  011276  000137  010650                  JMP     WRDINC
1540  011302  032737  000004  001126  CLEAR:  BIT     #BIT2,FLAG      ;XFER TEST?
1541  011310  001404                          BEQ     3$              ;NO
1542  011312  032737  010000  001126          BIT     #BIT12,FLAG     ;READ?
1543  011320  001412                          BEQ     2$              ;NO
1544  011322  013700  001116          3$:     MOV     VADDR,R0        ;GET STARTING ADD OF BUFFER
1545  011326  013701  001130                  MOV     WRDCT,R1        ;NOW
1546  011332  005020                  1$:     CLR     (R0)+           ;CLEAR BUFFER
1547  011334  005301                          DEC     R1              ;COUNT LOCATIONS
1548  011336  001375                          BNE     1$              ;WAIT TILL DONE
1549  011340  042737  000010  001122          BIC     #BIT3,FLAG2     ;DONE WITH COMPARE
1550  011346  000205                  2$:     RTS     R5              ;NOW GET OUT
1551
1552  011350  013737  001072  017416  APORT:  MOV     SAVAST,OUTBUF   ;SET STARTING ADDR FOR OUTBUF
1553  011356  013737  001072  001116          MOV     SAVAST,VADDR    ;SAVE OUTBUF ADDR
1554  011364  005737  001212                  TST     MMAVA           ;MEM MGMT?
1555  011370  001411                          BEQ     EXIT            ;NO
1556  011372  013702  001100                  MOV     SAVMGA,R2       ;SET UP MEM MGMT
1557  011376  004737  012020          MMSET:  JSR     PC,STMM2        ;SETUP MEM MGMT
1558  011402  010237  001116                  MOV     R2,VADDR
1559  011406  013737  001120  017416          MOV     PHADDR,OUTBUF
1560  011414  000207                  EXIT:   RTS     PC
1561
1562  011416  013737  001112  001130  BPORT:  MOV     WDCTB,WRDCT     ;GET WC FOR B PORT
1563  011424  013737  001076  017416          MOV     SAVCPU,OUTBUF
1564  011432  013737  001076  001116          MOV     SAVCPU,VADDR
1565  011440  005737  001212                  TST     MMAVA           ;MEM MGMT AVAILABLE?
1566  011444  001763                          BEQ     EXIT            ;NO
1567  011446  013702  001104                  MOV     SAVMGC,R2
1568  011452  000751                          BR      MMSET
```

```
1569                                    ;TYPE CAN NOT WRITE BLOCK
1570
1571  011454  004737  014204      WTNO:   JSR     PC,PRNT         ;TYPEOUT?
1572  011460  001001                      BNE     1$              ;NO
1573  011462  000000                      HALT                    ;HALT CANT WRITE BLOCK
1574  011464  005037  001200      1$:     CLR     LOPCNT          ;CLEAR ERR COUNTER
1575  011470  000207                      RTS     PC
1576
1577                                    ;ROUTINE TO SET UP STARTING ADDRESS FOR ALL PORTS
1578                                    ;AND TO CREATE WORD COUNT MAX= 20K
1579
1580  011472  013702  001070      EXTMEM: MOV     STAMEM,R2       ;GET BANK 0
1581  011476  005702                      TST     R2              ;DID  WE TYPE 0?
1582  011500  001001                      BNE     3$              ;NO
1583  011502  005202                      INC     R2              ;YES MAKE 1
1584  011504  005737  001212      3$:     TST     MMAVA
1585  011510  001021                      BNE     1$              ;BRANCH IF MEM MGMT AVAILABLE
1586  011512  000241                      CLC
1587  011514  004737  011736              JSR     PC,RRR2
1588  011520  010237  001072              MOV     R2,SAVAST       ;SAVE A STARTING ADDR
1589  011524  032737  000100  001126      BIT     #BIT6,FLAG      ;IS THERA A B PORT?
1590  011532  001410                      BEQ     2$              ;NO
1591  011534  013702  001074              MOV     STBCOM,R2       ;YES GET STARTING
1592  011540  000241                      CLC
1593  011542  004737  011736              JSR     PC,RRR2
1594  011546  010237  001076              MOV     R2,SAVCPU       ;SAVE IT
1595  011552  000420                      BR      2$              ;GET WC
1596  011554  000302              1$:     SWAB    R2
1597  011556  006002                      ROR     R2
1598  011560  010237  001100              MOV     R2,SAVWCA       ;SAVE ADDR FOR A PORT
1599  011564  032737  000100  001126      BIT     #BIT6,FLAG      ;IS THERE B PORT?
1600  011572  001410                      BEQ     2$              ;NO
1601  011574  013702  001074              MOV     STBCOM,R2
1602  011600  000302                      SWAB    R2
1603  011602  006002                      ROR     R2
1604  011604  010237  001102              MOV     R2,SAVWCB       ;SAVE B STARTING ADDR
1605  011610  010237  001104              MOV     R2,SAVWCC       ;SAVE CPU STARTING ADDR
1606  011614  013702  001106      2$:     MOV     SIZEAP,R2       ;GET 4K BLOCK COUNT
1607  011620  022737  000007  001106      CMP     #7,SIZEAP       ;IS IT GREATER THEN 20K?
1608  011626  101411                      BLOS    4$              ;YES MAKE IT 20K
```

```
1609  011630  000241                    88:     CLC
1610  011632  006002                            ROR     R2                  ;NO CONVERT TO WC
1611  011634  004737  011736                    JSR     PC,RRR2
1612  011640  042702  000077                    BIC     #77,R2              ;CLEAR BLOCK COUNT
1613  011644  010237  001144                    MOV     R2,SWRDCT           ;SAVE -A- PORT WC
1614  011650  000403                            BR      58                  ;CONT
1615  011652  012737  060000  001144  48:     MOV     #60000,SWRDCT       ;MAKE 20K
1616  011660  032737  000100  001126  58:     BIT     #BIT6,FLAG          ;IS THERE B PORT?
1617  011666  001422                            BEQ     78                  ;NO GET OUT
1618  011670  013702  001110                    MOV     SIZEBP,R2           ;GET B 4K COUNT
1619  011674  022737  000007  001110            CMP     #7,SIZEBP           ;IS IT GREATER THEN 20K?
1620  011702  101411                            BLOS    68                  ;YES MAKE 20K
1621  011704  000241                    98:     CLC
1622  011706  006002                            ROR     R2                  ;NO CONVERT TO WC
1623  011710  004737  011736                    JSR     PC,RRR2
1624  011714  042702  000077                    BIC     #77,R2              ;CLEAR SECTOR COUNT
1625  01172   010237  001112                    MOV     R2,WDCTB            ;SAVE WC FOR -B- PORT
1626  01172   000403                            BR      78                  ;GET OUT
1627  011726  012737  060000  001112  68:     MOV     #60000,WDCTB        ;MADE 20K WD
1628  011734  000207                    78:     RTS     PC
1629
1630  011736  006002                    RRR2:   ROR     R2
1631  011740  006002                            ROR     R2
1632  011742  006002                            ROR     R2
1633  011744  006002                            ROR     R2
1634  011746  000207                            RTS     PC
1635
1636  011750  032777  000200  167050  WAIT:   BIT     #BIT7,@SWR          ;WAIT IN BACKGROUND?
1637  011756  001003                            BNE     18                  ;NO
1638  011760  004737  012312                    JSR     PC,XWAIT            ;YES
1639  011764  000401                            BR      28                  ;CONT
1640  011766  000001                    18:     WAIT
1641  011770  000207                    28:     RTS     PC
1642
1643  011772  004737  014204          TYPREC: JSR     PC,PRNT             ;TYPEOUT?
1644  011776  001007                            BNE     18                  ;NO
1645  012000  104402  000616                    TYPE    ,RECOV
1646  012004  013746  001200                    MOV     LOPCNT,-(6)         ;GET COUNT
1647  012010  104406                            TYPES                       ;TYPE IT
1648  012012  104402  000636                    TYPE    ,CRLFLF
1649  012016  000207                    18:     RTS     PC
```

```
1650
1651  012020  005737  001212          STMM2:  TST     MMAVA           ;MEM MGMT?
1652  012024  001002                          BNE     3$              ;YES
1653  012026  000137  012304                  JMP     MDON            ;GET OUT
1654  012032  005037  172340          3$:     CLR     @#KIPAR0
1655  012036  010237  001150                  MOV     R2,SAVE         ;SAVE R2

1656  012042  010237  172342                  MOV     R2,@#KIPAR1
1657  012046  006302                          ASL     R2              ;CALCULATE PHYSICAL ADDR
1658  012050  006302                          ASL     R2
1659  012052  006302                          ASL     R2
1660  012054  006302                          ASL     R2
1661  012056  006302                          ASL     R2              ;THIS BIT IS A17
1662  012060  042737  000040  001122          BIC     #BIT5,FLAG2     ;CLEAR A17?
1663  012066  103003                          BCC     1$              ;SET A17
1664  012070  052737  000040  001122          BIS     #BIT5,FLAG2     ;SET BIT 5 FOR A17
1665  012076  042737  000020  001122  1$:     BIC     #BIT4,FLAG2     ;CLEAR A16 FLAG
1666  012104  006302                          ASL     R2              ;GET A16 BIT
1667  012106  103003                          BCC     2$              ;CLEAR A16
1668  012110  052737  000020  001122          BIS     #BIT4,FLAG2     ;SET FLAG FOR A16
1669  012116  010237  001120          2$:     MOV     R2,PHADDR       ;GET PHYSICAL ADDR
1670  012122  013702  001150                  MOV     SAVE,R2         ;SET UP MEM MGMT
1671  012126  062702  000200                  ADD     #200,R2
1672  012132  010237  172344                  MOV     R2,@#KIPAR2
1673  012136  062702  000200                  ADD     #200,R2
1674  012142  010237  172346                  MOV     R2,@#KIPAR3
1675  012146  062702  000200                  ADD     #200,R2
1676  012152  010237  172350                  MOV     R2,@#KIPAR4
1677  012156  062702  000200                  ADD     #200,R2
1678  012162  010237  172352                  MOV     R2,@#KIPAR5
1679  012166  062702  000200                  ADD     #200,R2
1680  012172  010237  172354                  MOV     R2,@#KIPAR6
1681  012176  012737  077406  172300          MOV     #200*256.-400*UP*RW,@#KIPDR0    ;SET KIPDR0=RW UP 200 BLOCKS
1682  012204  012737  077406  172302          MOV     #200*256.-400*UP*RW,@#KIPDR1    ;SET KIPDR1=RW UP 200 BLOCKS
1683  012212  012737  077406  172304          MOV     #200*256.-400*UP*RW,@#KIPDR2    ;SET KIPDR2=RW UP 200 BLOCKS
1684  012220  012737  077406  172306          MOV     #200*256.-400*UP*RW,@#KIPDR3    ;SET KIPDR3=RW UP 200 BLOCKS
1685  012226  012737  077406  172310          MOV     #200*256.-400*UP*RW,@#KIPDR4    ;SET KIPDR4=RW UP 200 BLOCKS
1686  012234  012737  077406  172312          MOV     #200*256.-400*UP*RW,@#KIPDR5    ;SET KIPDR5=RW UP 200 BLOCKS
1687  012242  012737  077406  172314          MOV     #200*256.-400*UP*RW,@#KIPDR6    ;SET KIPDR6=RW UP 200 BLOCKS
1688  012250  012737  077406  172316          MOV     #200*256.-400*UP*RW,@#KIPDR7    ;SET KIPDR7=RW UP 200 BLOCKS
1689  012256  012737  007600  172356          MOV     #7600,@#KIPAR7
1690  012264  012702  020000                  MOV     #20000,R2
1691  012270  012737  012306  000250          MOV     #MMABT0,@#MMVEC
1692  012276  012737  000001  177572          MOV     #1,@#SR0        ;TURN ON MEM MGMT
1693  012304  000207                  MDON:   RTS     PC
1694
1695                                  ;MEMORY MANAGEMENT ABORT ROUTINE FOR WRITE UP
1696  012306  000000                  MMABT0: HALT                    ;MEMORY MANAGEMENT TRAP
1697  012310  000002                          RTI                     ;CAUSED THE ABORT
```

```
1698                                    ;BACKGROUND TEST FOR INTERRUPTS
1699
1700  012312  052737  002000  001122  XWAIT:  BIS     #BIT10,FLAG2    ;WAITING IN BACKGROUND TEST
1701  012320  012737  170000  012426          MOV     #170000,NPRCNT  ;SETUP TIMEOUT COUNTER
1702  012326  012701  012431                  MOV     #NPR1+1,R1      ;SETUP WAIT LOOP
1703  012332  112711  000200                  MOVB    #200,(R1)
1704  012336                          28:
1705  012336  105421                          NEGB    (R1)+
1706  012340  105441                          NEGB    -(R1)
1707  012342  105421                          NEGB    (R1)+
1708  012344  105441                          NEGB    -(R1)
1709  012346  105421                          NEGB    (R1)+
1710  012350  105441                          NEGB    -(R1)
1711  012352  105421                          NEGB    (R1)+
1712  012354  105441                          NEGB    -(R1)
1713  012356  105421                          NEGB    (R1)+
1714  012360  105441                          NEGB    -(R1)
1715  012362  105421                          NEGB    (R1)+
1716  012364  105441                          NEGB    -(R1)
1717  012366  105421                          NEGB    (R1)+
1718  012370  105441                          NEGB    -(R1)
1719  012372  105421                          NEGB    (R1)+
1720  012374  105441                          NEGB    -(R1)
1721  012376  102401                          BVS     18
1722  012400  000000                          HALT                    ;ARITHMETIC OPERATION FAILED RUN DIAG
1723  012402  005337  012426          18:     DEC     NPRCNT
1724  012406  001353                          BNE     28
1725  012410  104004                          MLT     ;DAIMC;DS       ;TIMED OUT NO INTERRUPT
1726  012412  000137  001234                  JMP     @#BEGIN
1727  012416  042737  002000  001122  NPRRET: BIC     #BIT10,FLAG2    ;CLEAR BKGROUND FLG
1728  012424  000207                          RTS     PC
1729  012426  000000                  NPRCNT: 0
1730  012430  000000                  NPR1:   0
1731                                    ;CLEAR ERROR TABLE
1732
1733  012432  012704  000020          ERRCL:  MOV     #20,R4          ;CLEAR
1734  012436  012703  017052                  MOV     #ERTAB,R3       ;ERROR
1735  012442  005023                  18:     CLR     (R3)+           ;TABLE
1736  012444  005304                          DEC     R4              ;DONE YET?
1737  012446  001375                          BNE     18              ;NO
1738  012450  005037  001004                  CLR     PCNT            ;CLEAR
1739  012454  005037  001006                  CLR     PCNT+2          ;PASS COUNT
1740  012460  005037  001124                  CLR     DROP            ;CLEAR ALL DROPPED DRIVES
1741  012464  000205                          RTS     R5              ;RETURN
```

```
                                        ;RH11 POWER FAIL TEST #1
                                        ;THE STARTING ADDRESS FOR THE WRITE POWER FAIL TEST IS 244. THE PROGRAM
                                        ;WRITE THE COMPLETE DISK WITH A 125252 PATTERN. THE PROGRAM WILL THEN
                                        ;TELL OPERATOR TO POWER DOWN, UNTIL THE POWER FAIL, THE PROGRAM WILL
                                        ;CONTINUE WRITING THE SAME PATTERN ON THE DISK.
                                        ;WHEN POWER FAIL OCCURS THE TRANSFER IS ABORTED
                                        ;AND THE PROGRAM HALTS. THE OPERATOR SHOULD
                                        ;NOW TURN POWER BACK ON. THE PROGRAM RESTARTS AND CHECKS FOR WRITE ERRORS.
                                        ;ONLY ONE ERROR IS ACCEPTABLE, THAT ERROR MAY BE AN OPI (BIT13 RSER)OR A DCK
                                        ;(BIT 15 RSER). IF THESE ARE THE ONLY ERRORS THAT OCCUR, THE DRIVE IS OK.
                                        ;IF NO ERRORS OCCUR, THE PROGRAM WILL TYPE OUT "OK".
                                        ;THE PROGRAM WILL THEN TELL YOU WHEN TO POWER DOWN AGAIN

                                        ;***ONLY ONE ERROR IS CONSIDERED ACCEPTABLE***
                                        ;NOTE: ALL DRIVES ON THE SYSTEM SHOULD BE POWERED OFF EXCEPT
                                        ;THE DRIVE UNDER TEST. **********

012466  012706  000500          PFT1:   MOV     #500,SP         ;SET UP STACK
012472  104402  000510                  TYPE    ,LOADSW
012476  104420                          RDOCT
012500  012637  001160                  MOV     (SP)+,UNNUM
012504  004737  006774          28:     JSR     PC,VECTRR       ;SETUP INT VECTOR
012510  004737  006640                  JSR     PC,FMDTYP       ;TST FOR RS03 OR 04
012514  104426                  PFWAIT: CLRDV                   ;CLEAR ALL REG
012516  004737  013272                  JSR     PC,POWFAL       ;WRITE 125252 ON DISK
012522  005037  001134          PFWAT:  CLR     DMA
012526  012737  012750  000024          MOV     #DOWN,24        ;SET UP POWER FAIL VEC.
012534  012737  000340  000026          MOV     #340,26
012542  012737  000161  001172  MYBYWR: MOV     #161,CMD        ;WRITE WITH I/E
012550  104416                          DKCMD                   ;DO IT
012552  004737  011750                  JSR     PC,WAIT         ;WAIT FOR INTERRUPT
012556  032737  001000  001126  38:     BIT     #BIT9,FLAG      ;ANY ERRORS?
012564  001406                          BEQ     18              ;NO
012566  104006                          HLT     !DA!DB
012570  012777  177777  166252          MOV     #-1,@RSAS       ;CLEAR ALL
012576  005077  166244                  CLR     @RSER           ;ERRORS
012602  004737  007050          18:     JSR     PC,DISBUF       ;SET UP NEW DISK BUFFER
012606  000755                          BR      MYBYWR
012610  000744                          BR      PFWAT
```

```
1781  012612  012737  012620  001152  UPCHK:  MOV     #16,HRDER       ;RETURN HERE IF WRONG DRIVE INTERRUPTS
1782  012620  005037  001134          16:     CLR     DMA
1783  012624  104426                  CLRDV                           ;INIT DRIVE
1784  012626  013737  001066  177776  CHKDAT: MOV     PRIORITY,PS
1785  012634  012737  000151  001172          MOV     #151,CMD        ;WRITECHECK WITH I/E
1786  012642  104416                  DKCMD                           ;DO IT
1787  012644  013737  001066  177776          MOV     PRIORITY,PS
1788  012652  004737  011750                  JSR     PC,WATT         ;WAIT FOR INTERRUPT
1789  012656  032737  001000  001126  36:     BIT     #BIT9,FLAG      ;ANY ERRORS?
1790  012664  001411                  BEQ     18              ;NO
1791  012666  104006                  HLT     ;DBIDA
1792  012670  052737  100000  001122          BIS     #BIT15,FLAG2    ;SET ERROR FLAG
1793  012676  005077  166144                  CLR     @RSER           ;CLEAR ALL
1794  012702  012777  177777  166140          MOV     #-1,@RSAS       ;ERRORS
1795  012710  004737  007050          18:     JSR     PC,DISBUF       ;SET UP NEW DISK BUFFER
1796  012714  000744                  BR      CHKDAT
1797  012716  005737  001122                  TST     FLAG2           ;ANY ERRORS?
1798  012722  100405                  BMI     20              ;YES
1799  012724  104402  012730                  TYPE    ,.+2            ;.ASCIZ <15><12>"OK"
1800  012736  042737  100000  001122  20:     BIC     #BIT15,FLAG2    ;CLEAR ERROR FLAG
1801  012744  000137  012514                  JMP     @#WATT          ;GO WAIT FOR ANOTHER
1802                                                                  ;POWER FAIL
1803
1804                                  ;POWER DOWN ROUTINE - ABORT DISK AND HALT
1805
1806  012750  012737  012760  000024  DOWN:   MOV     #UPP,24         ;SET POWER FAIL VECTOR
1807  012756  000000                  HALT
1808
1809  012760  012737  012750  000024  UPP:    MOV     #DOWN,24
1810  012766  012706  000500                  MOV     #500,SP
1811  012772  013777  001160  166034          MOV     UNNUM,@RSCS2    ;GET UNIT #
1812  013000  032777  000200  166036  18:     BIT     #BIT7,@RSDS     ;WAIT FOR DRIVE READY
1813  013006  001774                  BEQ     18
1814  013010  000137  012612                  JMP     UPCHK           ;GO CHECK DISK
```

```
                                        ;POWER FAIL TEST #2
                                        ;THIS TEST WILL TEST THE SAME DRIVE THAT WAS TESTED IN THE 1ST POWER FAIL TEST
                                        ;THE PROGRAM WILL WRITE THE COMPLETE DISK WITH A 125252 PATTERN AND WILL
                                        ;THEN TELL THE OPERATOR TO POWER DOWN THE PROCESSOR.
                                        ;THE PROGRAM WILL THEN WRITE CHECK THE DISK WHILE WAITING FOR A POWER FAIL.
                                        ;WHEN THE POWER FAIL OCCURS, THE WRITE CHECKING IS ABORTED AND

                                        ;THE PROCESSOR WILL HALT.
                                        ;THE OPERATOR SHOULD THEN TURN POWER BACK ON, THE PROGRAM WILL
                                        ;START WRITE CHECKING THE DISK AGAIN
                                        ;***NO ERRORS SHOULD OCCUR.***
                                        ;THE PROGRAM WILL TYPE OUT "OK" IF NO ERRORS OCCUR.
                                        ;THE PROGRAM WILL THEN TELL YOU TO POWER DOWN.
                                        ;DO NOT POWER OFF THE PROCESSOR AGAIN UNTIL THE PROGRAM TELLS YOU SO.
                                        ;NOTE: ALL DRIVES ON THE SYSTEM SHOULD BE POWERED OFF
                                        ;EXCEPT THE ONE UNDER TEST ********************

013014  012706  000500          PFT2:   MOV     #500,SP         ;SET UP STACK
013020  042737  001000  001122          BIC     #BIT9,FLAG2     ;CLEAR POWER FAIL
013026  012737  013050  001152          MOV     #PWRF1,NRDER    ;RETURN HERE IF WRONG DRIVE INT.
013034  104426                          CLRDV                   ;INIT DRIVE
013036  004737  006774                  JSR     PC,VECTRR       ;SETUP INT VECTOR
013042  004737  013272          PWRFL2: JSR     PC,POWFAL       ;WRITE 125252 ON DISK
013046  000401                          BR      PWRFL           ;WRITE CHECK
013050  104426                  PWRF1:  CLRDV                   ;INIT DRIVE
013052  005037  001134          PWRFL:  CLR     DMA
013056  012737  013222  000024          MOV     #PWRDN,24       ;SET UP POWER FAIL VEC.
013064  012737  000340  000026          MOV     #340,26
013072  013737  001066  177776  CHKDSK: MOV     PRIORITY,PS     ;ENABLE I/E
013100  012737  000151  001172          MOV     #151,CMD        ;WRITE CHECK WITH I/E
013106  104416                          DKCMD                   ;DO IT
013110  004737  011750                  JSR     PC,WAIT         ;WAIT FOR INTERRUPT
013114  032737  001000  001126  30:     BIT     #BIT9,FLAG      ;ANY ERRORS?
013122  001411                          BEQ     1$              ;NO
013124  104002                          HLT     ^D2             ;YES
013126  052737  100000  001122          BIS     #BIT15,FLAG2    ;SET ERROR FLAG
013134  005077  165706                  CLR     @RSER           ;CLEAR ALL
013140  012777  177777  165702          MOV     #-1,@RBAS       ;ERRORS
013146  004737  007050          18:     JSR     PC,DISBUF       ;CHECK NEXT BUFFER
013152  000747                          BR      CHKDSK
013154  032737  001000  001122          BIT     #BIT9,FLAG2     ;DID POWER FAIL?
013162  001733                          BEQ     PWRFL           ;NO
013164  005737  001122                  TST     FLAG2           ;ANY ERRORS?
013170  100405                          BMI     2$              ;YES
013172  104402  013176                  TYPE    ,.+2            ;.ASCIZ <15><12>"OK"
013204  042737  100000  001122  28:     BIC     #BIT15,FLAG2    ;CLEAR ERRORS
013212  042737  001000  001122          BIC     #BIT9,FLAG2     ;CLEAR POWER FAIL FLAG
013220  000710                  48:     BR      PWRFL2
```

```
                                        ;ROUTINE TO ABORT DISK DURING POWER FAIL
                                        ;
013222  012737  013232  000024  PWRDN:  MOV     #PWRUP,24       ;SET UP RESTART
013230  000000                          HALT
                                        ;
013232  012737  013222  000024  PWRUP:  MOV     #PWRDN,24       ;RESET POWER FAIL VECTOR
013240  012706  000500                  MOV     #500,SP
013244  013777  001160  165562          MOV     UNNUM,@RSCS2    ;GET UNIT #
013252  052737  001000  001122          BIS     #BIT9,FLAG2     ;SET POWER FAIL BIT
013260  032777  000200  165556  1$:     BIT     #BIT7,@RSDS     ;WAITING FOR
013266  001774                          BEQ     1$              ;DRIVE READY
013270  000667                          BR      PWRF1           ;GO CHECK DISK
                                        ;
                                        ;
                                        ;ROUTINE TO WRITE THE COMPLETE DISK
                                        ;WITH 125252 PATTERN
                                        ;WRITE CHECK AND REPORT ERRORS IF THEY  OCCUR
                                        ;REPORT "OK" AT COMPLETION
                                        ;
013272  012737  000020  001136  POWFAL: MOV     #20,PATNU       ;SET UP PATTERN
013300  042737  000004  001126          BIC     #BIT2,FLAG      ;CLEAR XFER MODE FLAG
013306  052737  010000  001122          BIS     #BIT12,FLAG2
013314  005037  001134                  CLR     DMA
013320  012737  020000  017416          MOV     #20000,OUTBUF   ;GET STARTING ADDR FOR BUF
013326  012737  020000  001116          MOV     #20000,VADDR
013334  012737  010000  001144          MOV     #10000,SWRDCT   ;SETUP WORD COUNT
013342  013737  001144  001130          MOV     SWRDCT,WRDCT
013350  005037  001114                  CLR     ADBI            ;A PORT ONLY
013354  013737  017416  001140          MOV     OUTBUF,BUF      ;SET UP CURRENT ADDRESS
013362  004537  007716                  JSR     R5,PASEL        ;GENERATE DATA BUFFER
013366  012737  000161  001172  WRDNW:  MOV     #161,CMD        ;WRITE WITH I/E
013374  104416                          DKCMD                   ;DO IT
013376  004737  011750                  JSR     PC,WATT         ;WAIT FOR INTERRUPT
013402  012737  000151  001172  2$:     MOV     #151,CMD        ;WRITECHECK I/E
013410  104416                          DKCMD                   ;DO IT
013412  004737  011750                  JSR     PC,WATT         ;WAIT FOR INTERRUPT
013416  032737  001000  001126  4$:     BIT     #BIT9,FLAG      ;ANY ERRORS?
013424  001402                          BEQ     1$              ;NO
013426  104006                          HLT     ;DB;DA          ;YES
013430  000000                          HALT                    ;CAN NOT WRITE WITHOUT ERROR
013432  004737  007060          1$:     JSR     PC,DISBUF       ;SET UP NEW DISK BUFFER
013436  000753                          BR      WRDNW           ;WRITE NEW BUFFER
013440  104402  000743                  TYPE    ,PDOWN
013444  000207                          RTS     PC
```

```
1906  013446  032777  000010  165352  OUT:    BIT     #BIT3,@SWR      ;TYPEOUT ERROR COUNT?
1907  013454  001526                          BEQ     1$              ;NO
1908  013456  005004                          CLR     R4              ;CLEAR UNIT #
1909  013460  005003                          CLR     R3
1910  013462  053737  001124  001162          BIS     DROP,UNITSV     ;RESTORE ALL DRIVES
1911  013470  013737  001162  001222          MOV     UNITSV,WORK     ;GET UNITS ON SYSTEM
1912  013476  012705  000401                  MOV     #401,R5         ;SETUP TEST FOR UNITS
1913  013502  030537  001222          4$:     BIT     R5,WORK         ;IS THIS UNIT ON SYS
1914  013506  001006                          BNE     2$              ;YES
1915  013510  005204                  5$:     INC     R4              ;INC UNIT #
1916  013512  010403                          MOV     R4,R3           ;SAVE UNIT #
1917  013514  000241                          CLC
1918  013516  006105                          ROL     R5              ;GET NEXT DRIVE
1919  013520  103001                          BCS     3$              ;DONE
1920  013522  000767                          BR      4$              ;FIND NEXT DRIVE
1921  013524  104402  000510          2$:     TYPE    ,LOADSW
1922  013530  010446                          MOV     R4,-(6)         ;PUT R4 ON STACK
1923  013532  104406                          TYPES                   ;TYPE STACK IN OCTAL - SUPRESS
1924  013534  004737  014012                  JSR     PC,GETERR       ;GET ERROR COUNT
1925  013540  010304                          MOV     R3,R4           ;RESTORE UNIT #
1926  013542  104402  013546                  TYPE    .,.+2           ;.ASCII <15><12>
1927  013552  104402  000564                  TYPE    ,WRTERR
1928  013556  104402  013562                  TYPE    .,.+2           ;.ASCII "S "
1929  013566  013746  001202                  MOV     WRITER,-(6)     ;PUT WRITER ON STACK
1930  013572  104406                          TYPES                   ;TYPE STACK IN OCTAL - SUPRESS
1931  013574  104402  013600                  TYPE    .,.+2           ;.ASCII <15><12>
1932  013604  104402  000607                  TYPE    ,RDERR
1933  013610  104402  013614                  TYPE    .,.+2           ;.ASCII "S "
1934  013620  013746  001206                  MOV     READER,-(6)     ;PUT READER ON STACK
1935  013624  104406                          TYPES                   ;TYPE STACK IN OCTAL - SUPRESS
1936  013626  104402  013632                  TYPE    .,.+2           ;.ASCII <15><12>
1937  013636  104402  000574                  TYPE    ,WCKERR
1938  013642  104402  013646                  TYPE    .,.+2           ;.ASCII "S "
1939  013652  013746  001204                  MOV     WCERR,-(6)      ;PUT WCERR ON STACK
1940  013656  104406                          TYPES                   ;TYPE STACK IN OCTAL - SUPRESS
1941  013660  104402  013664                  TYPE    .,.+2           ;.ASCII <15><12>"COMPARE ERRS "
1942  013704  013746  001210                  MOV     COMERR,-(6)     ;PUT COMERR ON STACK
1943  013710  104406                          TYPES                   ;TYPE STACK IN OCTAL - SUPRESS
1944  013712  104402  013716                  TYPE    .,.+2           ;.ASCII <15><12>
1945  013722  000672                          BR      5$              ;GET NEXT DRIVE
1946  013724  043737  001124  001162  3$:     BIC     DROP,UNITSV     ;REDROP DRIVES
1947  013732  062706  000002          1$:     ADD     #2,SP           ;RESTORE SP DUE TO JMP EXIT FROM JSR ROUTINE
1948  013736  005137  001114                  COM     ASB1            ;SET A OR B PORT FLAG
1949  013742  032777  000040  165056          BIT     #BIT5,@SWR      ;TYPEOUT PASS COUNT?
1950  013750  001035                          BNE     DONE            ;NO
1951  013752  104402  013756                  TYPE    .,.+2           ;.ASCII <15><12>"END PASS "
1952  013772  013746  001006                  MOV     PCNT+2,-(6)     ;PUT PCNT+2 ON STACK
1953  013776  104406                          TYPES                   ;TYPE STACK IN OCTAL - SUPRESS
1954  014000  104402  014004                  TYPE    .,.+2           ;.ASCII <15><12>
1955  014010  000415                          BR      DONE
```

```
1956  014012  006304                  GETERR: ASL     R4              ;GET LOC IN
1957  014014  006304                          ASL     R4              ;ERR TABLE
1958  014016  062704  017052                  ADD     #ERTAB,R4
1959  014022  112437  001202                  MOVB    (R4)+,WRITER    ;GET WRITE ERRS
1960  014026  112437  001206                  MOVB    (R4)+,READER    ;GET READ ERRS
1961  014032  112437  001204                  MOVB    (R4)+,WCERR     ;GET WRITE CK ERRS
1962  014036  112437  001210                  MOVB    (R4)+,COMERR    ;GET COMPARE ERRS
1963  014042  000207                          RTS     PC
1964
1965                                  .SBTTL          #DONE - BELL AND SCOPE ROUTINE
1966
1967  014044  104400                  DONE:   SCOPE                   ;TERMINATIONG SCOPE FOR LOOPING
1968  014046  062737  000001  001006          ADD     #1,PCNT+2       ;ADD 1 TO THE PASS COUNT
1969  014054  005537  001004                  ADC     PCNT            ;MAKE IT DOUBLE PREC.
1970  014060  013700  000042          4$:     MOV     @#42,R0         ;GET MONITOR ADDRESS
1971  014064  001405                          BEQ     #END1           ;IF NONE
1972  014066  000005                          RESET
1973  014070  004710                  #ENDAD: JSR     7,(0)           ;GO TO MONITOR
1974  014072  000240  000240  000240          240,240,240             ;SAVE ROOM FOR ACT11
1975  014100  000137  003224          #END1:  JMP     ADTST           ;RETURN
1976
1977  014104  000000                  .TBIT:  0                       ;T BIT FLAG
1978
1979  014106  012702  000001          .LOGW:  MOV     #1,R2           ;LOG WRITE ERR
1980  014112  005003                  CLIND:  CLR     R3              ;CLEAR INDEX FOR TABLE
1981  014114  000413                          BR      ADDRR
1982
1983  014116  012702  000400          .LOGR:  MOV     #400,R2         ;LOG WRITE ERR
1984  014122  000773                          BR      CLIND
1985
1986  014124  012702  000001          .LOGWC: MOV     #1,R2           ;LOG WRITE CK ERR
1987  014130  012703  000002          #ETIND: MOV     #2,R3           ;SET INDEX FOR NEXT WD
1988  014134  000403                          BR      ADDRR
1989
1990  014136  012702  000400          .LOGC:  MOV     #400,R2         ;LOG COMPARE ERR
1991  014142  000772                          BR      #ETIND
1992
1993  014144  005737  001200          ADDRR:  TST     LOPCNT          ;1ST TIME ERROR?
1994  014150  001014                          BNE     1$              ;NO DO NOT COUNT IT
1995  014152  013704  001150                  MOV     UNNUM,R4        ;GET UNIT #
1996  014156  006304                          ASL     R4              ;GET
1997  014160  006304                          ASL     R4              ;POSITION IN
1998  014162  060304                          ADD     R3,R4           ;ERR TABLE
1999  014164  060264  017052                  ADD     R2,ERTAB(R4)    ;TO ADD ERROR
2000  014170  004737  014204                  JSR     PC,PRNT         ;TYPEOUT?
2001  014174  001402                          BEQ     1$              ;YES
2002  014176  004737  014772                  JSR     PC,DRP          ;SHOULD I DROP DRIVE?
2003  014202  000002                  1$:     RTI
2004
2005  014204  032777  020000  164614  PRNT:   BIT     #BIT13,@#SWR    ;INHIBIT TYPEOUT?
2006  014212  000207                          RTS     PC
```

```
 2007  014214  052737  000004  001122  RSREG:  BIS     #BIT2,FLAG2     ;SET ERROR FLAG
 2008  014222  005737  016040                  TST     .NLTCT          ;SHOULD WE TYPE GOOD AND BAD
 2009  014226  001017                          BNE     8$              ;NO
 2010  014230  104402  014234                  TYPE    ,.+2            ;.ASCIZ "BAD<15>"
 2011  014242  010046                          MOV     R0,-(SP)        ;PUT R0 ON STACK
 2012  014244  104404                          TYPEO                   ;TYPE STACK IN OCTAL

 2013  014246  104402  014252                  TYPE    ,.+2            ;.ASCIZ " GOOD<15>"
 2014  014262  010146                          MOV     R1,-(SP)        ;PUT R1 ON STACK
 2015  014264  104404                          TYPEO                   ;TYPE STACK IN OCTAL
 2016  014266                          8$:
 2017  014266  104402  014272                  TYPE    ,.+2            ;.ASCIZ " CS1<15>"
 2018  014300  017746  164526                  MOV     @RSCS1,-(SP)    ;PUT @RSCS1 ON STACK
 2019  014304  104404                          TYPEO                   ;TYPE STACK IN OCTAL
 2020  014306                          18:
 2021  014306  104402  014312                  TYPE    ,.+2            ;.ASCIZ " ER<15>"
 2022  014320  017746  164522                  MOV     @RSER,-(SP)     ;PUT @RSER ON STACK
 2023  014324  104404                          TYPEO                   ;TYPE STACK IN OCTAL
 2024  014326                          28:
 2025  014326  104402  014332                  TYPE    ,.+2            ;.ASCIZ " CS2<15>"
 2026  014340  017746  164470                  MOV     @RSCS2,-(SP)    ;PUT @RSCS2 ON STACK
 2027  014344  104404                          TYPEO                   ;TYPE STACK IN OCTAL
 2028  014346  032737  000200  016040          BIT     #200,.NLTCT     ;PRINT SECOND SET ?
 2029  014354  001112                          BNE     SEEC            ;YES
 2030  014356  032737  000100  016040          BIT     #A8,.NLTCT      ;PRINT ER ?
 2031  014364  001410                          BEQ     3$              ;NO
 2032  014366  104402  014372                  TYPE    ,.+2            ;.ASCIZ " AS<15>"
 2033  014400  017746  164444                  MOV     @RSAS,-(SP)     ;PUT @RSAS ON STACK
 2034  014404  104404                          TYPEO                   ;TYPE STACK IN OCTAL
 2035  014406  032737  000020  016040  38:     BIT     #BA,.NLTCT      ;PRINT BUS ADDRESS
 2036  014414  001410                          BEQ     4$              ;NO
 2037  014416  104402  014422                  TYPE    ,.+2            ;.ASCIZ " BA<15>"
 2038  014430  017746  164404                  MOV     @RSBA,-(SP)     ;PUT @RSBA ON STACK
 2039  014434  104404                          TYPEO                   ;TYPE STACK IN OCTAL
 2040  014436  032737  000004  016040  48:     BIT     #DA,.NLTCT      ;PRINT DA ?
 2041  014444  001410                          BEQ     5$              ;NO
 2042  014446  104402  014452                  TYPE    ,.+2            ;.ASCIZ " DA<15>"
 2043  014460  017746  164356                  MOV     @RSDA,-(SP)     ;PUT @RSDA ON STACK
 2044  014464  104404                          TYPEO                   ;TYPE STACK IN OCTAL
 2045  014466  032737  000010  016040  58:     BIT     #MC,.NLTCT      ;PRINT MC?
 2046  014474  001410                          BEQ     6$              ;NO
 2047  014476  104402  014502                  TYPE    ,.+2            ;.ASCIZ " MC<15>"
 2048  014510  017746  164322                  MOV     @RSMC,-(SP)     ;PUT @RSMC ON STACK
 2049  014514  104404                          TYPEO                   ;TYPE STACK IN OCTAL
 2050  014516  032737  000040  016040  68:     BIT     #DS,.NLTCT      ;DRIVE STATUS
 2051  014524  001410                          BEQ     9$              ;NO
 2052  014526  104402  014532                  TYPE    ,.+2            ;.ASCIZ " DS<15>"
 2053  014540  017746  164300                  MOV     @RSDS,-(SP)     ;PUT @RSDS ON STACK
 2054  014544  104404                          TYPEO                   ;TYPE STACK IN OCTAL
 2055  014546  032737  000002  016040  98:     BIT     #DB,.NLTCT      ;PRINT DATA BUFFER
```

```
2056  014554  001461                          BEQ     PTDONE          ;NO
2057  014556  104402  014562                  TYPE    ,.+2            ;.ASCIZ " DB="
2058  014570  017746  164260                  MOV     @RSDB,-(6)      ;PUT @RSDB ON STACK
2059  014574  104404                          TYPEO                   ;TYPE STACK IN OCTAL
2060  014576  000137  014720                  JMP     PTDONE          ;GET OUT
2061  014602  042737  000200  016040  SEEC:   BIC     #200,.HLTCT     ;CLEAR COMMON BIT
2062  014610  032737  000240  016040          BIT     #DT,.HLTCT      ;PRINT DRIVE TYPE?
2063  014616  001410                          BEQ     10$             ;NO
2064  014620  104402  014624                  TYPE    ,.+2            ;.ASCIZ " DT="
2065  014632  017746  164222                  MOV     @RSDT,-(6)      ;PUT @RSDT ON STACK
2066  014636  104404                          TYPEO                   ;TYPE STACK IN OCTAL
2067  014640  032737  000220  016040  10$:   BIT     #NR,.HLTCT      ;PRINT NR?
2068  014646  001410                          BEQ     11$             ;NO
2069  014650  104402  014654                  TYPE    ,.+2            ;.ASCIZ " NR="
2070  014662  017746  164170                  MOV     @RSNR,-(6)      ;PUT @RSNR ON STACK
2071  014666  104404                          TYPEO                   ;TYPE STACK IN OCTAL
2072  014670  032737  000204  016040  11$:   BIT     #LA,.HLTCT      ;PRINT LA?
2073  014676  001410                          BEQ     PTDONE          ;NO
2074  014700  104402  014704                  TYPE    ,.+2            ;.ASCIZ " LA="
2075  014712  017746  164134                  MOV     @RSLA,-(6)      ;PUT @RSLA ON STACK
2076  014716  104404                          TYPEO                   ;TYPE STACK IN OCTAL
2077  014720  032737  010000  001122  PTDONE: BIT     #BIT12,FLAG2    ;POWER FAIL TEST?
2078  014726  001111                          BNE     RETT            ;YES
2079  014730  104402  014734                  TYPE    ,.+2            ;.ASCIZ <15><12>"PASS "
2080  014744  013746  001006                  MOV     PCNT+2,-(6)     ;PUT PCNT+2 ON STACK
2081  014750  104406                          TYPE$                   ;TYPE STACK IN OCTAL - SUPRESS
2082  014752  032777  001000  164046          BIT     #BIT9,@SWR      ;LOOPING ON ERROR?
2083  014760  001404                          BEQ     DRP             ;NO
2084  014762  104402  014766                  TYPE    ,.+2            ;.ASCIZ <15><12>
2085  014772  032777  000001  164026  DRP:    BIT     #BIT0,@SWR      ;DROP DRIVE?
2086  015000  001464                          BEQ     RETT            ;NO
2087  015002  013704  001160                  MOV     UNNUM,R4        ;GET UNIT #
2088  015006  004737  014012                  JSR     PC,GETERR       ;GET ERRORS
2089  015012  063737  001202  001206          ADD     WRITER,READER   ;ADD THE ERRORS
2090  015020  063737  001206  001204          ADD     READER,WCERR
2091  015026  063737  001204  001210          ADD     WCERR,COMERR
2092  015034  022737  000023  001210          CMP     #23,COMERR      ;DROP DRIVE?
2093  015042  103043                          BHIS    RETT            ;NO
2094  015044  053737  001164  001124          BIS     UNCMP,DROP      ;DROP DRIVE
2095  015052  104402  015056                  TYPE    ,.+2            ;.ASCIZ <15><12>"DROPPED UNIT "
2096  015076  013746  001160                  MOV     UNNUM,-(6)      ;PUT UNNUM ON STACK
2097  015102  104406                          TYPE$                   ;TYPE STACK IN OCTAL - SUPRESS
2098  015104  104402  000636                  TYPE    ,CRLFLF
2099  015110  113703  001124                  MOVB    DROP,R3         ;GET DROPPED UNITS
2100  015114  113704  001162                  MOVB    UNITSV,R4       ;GET ALL DRIVES
2101  015120  020304                          CMP     R3,R4           ;ALL DRIVES DROPPED?
2102  015122  001003                          BNE     2$              ;NO
2103  015124  000000                          HALT                    ;NO MORE DRIVES
2104  015126  000137  001234                  JMP     @#BEGIN         ;RESTART TEST
2105  015132  032737  100000  001126  2$:     BIT     #BIT15,FLAG     ;DID OPERATOR SELECT PATTERN
2106  015140  001002                          BNE     3$              ;YES
2107  015142  005037  001136                  CLR     PATNU           ;NO CLEAR IT
2108  015146  000137  006666          3$:     JMP     @#EXITPR        ;GET NEXT DRIVE
2109  015152  000207                  RETT:   RTS     PC
```

```
2110                                        ;ROUTINE TO RESTORE LOADER
2111  015154  013705  015200        RLDR:   MOV     LDR1,R5         ;GET FIRST ADDRESS OF WHERE LOADER IS
2112                                                                ;TO BE RESTORED
2113  015160  012704  017446                MOV     #17446,R4       ;ADDRESS WHERE LOADER IS STORED
2114  015164  012702  000155                MOV     #155,R2         ;WORD COUNT
2115  015170  012425                18:     MOV     (R4)+,(R5)+     ;RESTORE

2116  015172  005302                        DEC     R2
2117  015174  001375                        BNE     1$
2118  015176  000000                        HALT                    ;DONE
2119  015200  017446                LDR1:   .WORD   17446           ;FIRST ADDRESS WHERE LOADERS ARE SAVED
2120
2121          172100                PARCSR=172100
2122          000114                PARVEC=114
2123  015202  012737  015274  000114  .MAMK:  MOV     #.PARSRV,@#PARVEC
2124  015210  012737  000340  000116        MOV     #340,@#PARVEC+2 ;SET PRI LEVEL TO 7
2125  015216  013746  000004                MOV     @#4,-(SP)       ;SAVE CURRENT ERROR VECTOR
2126  015222  013746  000006                MOV     @#6,-(SP)       ;SAVE PRIOITY LEVEL
2127  015226  012737  000006  000004        MOV     #6,@#4
2128  015234  012737  000002  000006        MOV     #RTI,@#6
2129  015242  012700  172100                MOV     #PARCSR,R0      ;GET FIRST CSR ADDR
2130  015246  012702  000001                MOV     #1,R2
2131  015252  012720  000001        18:     MOV     #1,(R0)+        ;SET ACTION ENABLE IF AVAILABLE
2132  015256  006302                        ASL     R2              ;SHIFT AVAILABILITY INDICATOR
2133  015260  103374                        BCC     1$
2134  015262  012637  000006                MOV     (SP)+,@#6       ;RESTORE ERROR VECTOR PRIORITY
2135  015266  012637  000004                MOV     (SP)+,@#4       ;AND INTERRUPT VECTOR
2136  015272  000207                        RTS     PC
2137                                ;PARITY MEMORY TRAP
2138
2139  015274  104402  000760        .PARSRV:        TYPE    ,PARERR
2140  015300  052737  004000  001122          BIS     #BIT11,FLAG2    ;SET ERROR FLAG
2141  015306  032737  000010  001122          BIT     #BIT3,FLAG2     ;WERE WE COMPARING DURING ERROR?
2142  015314  001422                          BEQ     13$             ;NO
2143  015316  104402  015322                  TYPE    .+2             ;,ASCII * ADDR=*
2144  015332  005737  001212                  TST     MMAVA           ;IS MEM MGMT ON?
2145  015336  001406                          BEQ     12$             ;NO
2146  015340  013746  177776                  MOV     PS,-(6)         ;GET PS
2147  015344  013746  001150                  MOV     SAVE,-(6)       ;GET VIRTUAL ADDR
2148  015350  104412                          TYPEA                   ;CONVERT TO PHY AND TYPE
2149  015352  000403                          BR      13$             ;CONT
2150  015354  013746  001150          12$:    MOV     SAVE,-(6)       ;GET ADDR
2151  015360  104406                          TYPE8                   ;TYPE IT
2152  015362  032777  100000  163436  13$:    BIT     #BIT15,@#MR     ;HALT ON ERROR?
2153  015370  001401                          BEQ     1$              ;NO
2154  015372  000000                          HALT                    ;YES
2155  015374  012706  000500          1$:     MOV     #500,SP         ;RESET STACK
2156  015400  000137  003240                  JMP     EXMFLG          ;RESTART TEST
```

```
2157                                    .SBTTL          $TYPE - TTY TYPEOUT ROUTINE
2158
2159                                    ;THIS ROUTINE IS USE TO TYPE ASCII MESSAGES ON THE TTY. THE
2160                                    ;CALL CAN BE IN ONE OF 3 FORMS: 1) "TYPE ,ADR" - TYPES THE
2161                                    ;MESSAGE STARTING IN LOCATION "ADR;", 2) "TYPE ,CHAR" - TYPES
2162                                    ;THE ASCII "CHAR", AND 3) "PRINT <<15><12>"MESSAGE"> - TYPES
2163                                    ;THE MESSAGE WHICH IS INLINE ASCII. THE FILLER CHARACTER WHICH IS
2164                                    ;TYPED AFTER A LINE FEED IS IN FILCHR AND THE NUMBER OF FILLERS
2165                                    ;IS IN FILCHR+1.
2166
2167  015404  010446                    .TYPE:  MOV     R4,-(6)         ;SAVE R4
2168  015406  010546                            MOV     R5,-(6)         ;SAVE R5
2169  015410  017605  000004                    MOV     @4(6),R5        ;GET ADDRESS TO BE TYPED
2170  015414  032705  177400                    BIT     #177400,R5      ;IS IT A TYPE?
2171  015420  001002                            BNE     1$              ;NO
2172  015422  016605  000004                    MOV     4(6),R5         ;GET ADDRESS OF CHARACTER
2173  015426  105715                    1$:     TSTB    (R5)            ;TERMINATOR?
2174  015430  001423                            BEQ     2$              ;GET OUT IF SO
2175  015432  122715  000012                    CMPB    #12,(R5)        ;IS THE CHAR A LINE FEED
2176  015436  001012                            BNE     4$              ;NO - GET OUT
2177  015440  113704  001015                    MOVB    FILCHR+1,R4     ;GET THE FILL COUNT
2178  015444  113777  001014  163352  5$:     MOVB    FILCHR,@TPB     ;TYPE A FILLER
2179  015452  105777  163340                    TSTB    @TPS            ;DONE YET?
2180  015456  100375                            BPL     .-4             ;NO - WAIT
2181  015460  005304                            DEC     R4              ;DEC COUNT
2182  015462  001370                            BNE     5$              ;LOOP UNTIL 0
2183  015464  112577  163334          4$:     MOVB    (R5)+,@TPB      ;LOAD AND TYPE THE CHARACTER
2184  015470  105777  163322                    TSTB    @TPS            ;IS THE PRINTER READY
2185  015474  100375                            BPL     .-4             ;WAIT UNTIL IT IS
2186  015476  000753                            BR      1$              ;GET THE NEXT CHARACTER
2187  015500  017646  000004          2$:     MOV     @4(6),-(6)      ;GET ADDRESS TO BE TYPED
2188  015504  062766  000002  000006          ADD     #2,6(6)         ;ADD 2 TO THE ADDRESS
2189  015512  022666  000004                    CMP     (6)+,4(6)       ;IS IT .+2?
2190  015516  001006                            BNE     3$              ;NO
2191  015520  062705  000002                    ADD     #2,R5           ;ADD 2 TO THE ADDRESS
2192  015524  042705  000001                    BIC     #1,R5           ;BACK UP TO AN EVEN BYTE
2193  015530  010566  000004                    MOV     R5,4(6)         ;RESTORE ADDRESS
2194  015534  012605                    3$:     MOV     (6)+,R5         ;RESTORE R5
2195  015536  012604                            MOV     (6)+,R4         ;RESTORE R4
2196  015540  000002                            RTI                     ;RETURN
```

```
                                .SBTTL          $SCOPE - SCOPE LOOP HANDLER

                                ;THIS ROUTINE HANDLES THE ITERATIONS, LOOPING, ERROR
                                ;LOOPING, AND THE DISPLAYING OF THE TEST NUMBER.
                                ;"SCOPE" IS PLACED BETWEEN EACH SUBTEST IN THE TEST AND
                                ;RECORDS THE STARTING ADDRESS OF THE SUBTEST IN "LAD:"

015542  104442                  .SCOPE: KBDIN                   ;GO CHECK FOR ^G
015544  032777  040000  163254          BIT     @$SW14,@$SWR    ;LOOP ON TEST?
015552  001045                        BNE     .KIT            ;LOOP ON TEST IS SET
015554  000416                        BR      3$              ;SKIP - NOP FOR XOR TESTER
015556  013746  000004                MOV     @#4,-(6)        ;PUSH @#4 ON STACK
015562  012737  015602  000004        MOV     #48,@#4         ;SET FOR TIMEOUT
015570  005737  177060                TST     @#177060        ;ERROR ON XOR?
015574  012637  000004                MOV     (6)+,@#4        ;POP STACK INTO @#4
015600  000422                        BR      .$VLAD          ;NO ERROR - GO TO NEXT TEST
015602  022626                  48:     CMP     (6)+,(6)+       ;CLEAR STACK
015604  012637  000004                MOV     (6)+,@#4        ;POP STACK INTO @#4
015610  000426                        BR      .KIT            ;ERROR - LOOP ON TEST
015612  032777  004000  163206  38:     BIT     @$SW11,@$SWR    ;KILL ITERATIONS
015620  001012                        BNE     .$VLAD          ;YES - KILL ITERATIONS
015622  105737  001061                TSTB    ICNT+1          ;FIRST ONE?
015626  001404                        BEQ     2$              ;BRANCH IF FIRST
015630  123737  015714  001001        CMPB    TIMES,ICNT+1    ;DONE?
015636  003013                        BGT     .KIT            ;BRANCH IF NOT
015640  112737  000001  001001  2$:    MOVB    #1,ICNT+1       ;FIRST ITERATION
015646  105237  001000                .$VLAD: INCB    ICNT            ;COUNT TEST NUMBERS
015652  011637  001010                MOV     (6),LAD         ;SAVE LOOP ADDRESS
015656  013777  001000  163144        MOV     ICNT,@$DISPLAY  ;DISPLAY TEST NO. AND ITERATION COUNT
015664  000002                        RTI                     ;RETURN

015666  105237  001001          .KIT:   INCB    ICNT+1          ;INC THE ITERATION COUNT
015672  013777  001000  163130  .OVER:  MOV     ICNT,@$DISPLAY  ;SET UP DISPLAY
015700  005737  001010                TST     LAD             ;FIRST ONE?
015704  001760                        BEQ     .$VLAD          ;YES
015706  013716  001010                MOV     LAD,(6)         ;FUDGE RETURN ADDRESS
015712  000002                        RTI                     ;FIXES PS

015714  000001                TIMES:  1                       ;RUN 1 TIMES
```

```
2236                                    .SBTTL          $HLT - HLT ROUTINE (ERROR TYPEOUT)
2237
2238                                    ;THIS ROUTINE PRINTS OUT ERROR MESSAGES STARTING WITH THE
2239                                    ;ADDRESS OF THE "HLT". IT ALSO COUNTS THE NUMBER OF ERRORS
2240                                    ;AND HAS THE CAPABILITY OF LOOPING ON ERROR, BELL ON ERROR,
2241                                    ;"HALT" ON ERROR, AND INHIBIT TYPEOUTS. AN OPTIONAL ARGUMENT
2242                                    ;(HLT+2) WILL BE PLACED IN ".HLTCT:" FOR ADITIONAL TYPEOUTS.
2243
2244  016716  005237  001002          .HLT:   INC     ERRORS          ;INC THE ERROR COUNT
2245  016722  104442                          KBDIN                   ;GO CHECK FOR ^G
2246  016724  032777  020000  163074          BIT     #SW13,@#SWR     ;SKIP TYPEOUT IF SET
2247  016732  001025                          BNE     2$              ;SKIP TYPEOUTS
2248  016734  104402  016746                  TYPE    ,.+2            ;.ASCIZ <15><12>
2249  016744  011637  001012                  MOV     (6),HLTADR      ;PUT ADDRESS OF INSTRUCTION ON STACK
2250  016750  162737  000002  001012          SUB     #2,HLTADR       ;FUDGE ADDRESS
2251  016756  117737  163030  016040          MOVB    @HLTADR,.HLTCT  ;GET HLT ARGUEMENT
2252  016764  013746  001012                  MOV     HLTADR,-(6)     ;PUT HLTADR ON STACK
2253  016770  104404                          TYPEO                   ;TYPE STACK IN OCTAL
2254  016772  104402  016776                  TYPE    ,.+2            ;.ASCIZ "  "
2255  017002  004737  014214                  JSR     PC,@#SREG       ;GO TO USER ERROR ROUTINE
2256  017006  005777  163014          28:     TST     @#SWR           ;HALT ON ERROR
2257  017012  100001                          BPL     .+4             ;SKIP IF CONTINUE
2258  017014  000000                          HALT                    ;HALT ON ERROR!
2259  017016  032777  001000  163002          BIT     #SW9,@#SWR      ;CHECK FOR INHIBIT LOOP ON ERROR
2260  017024  001003                          BNE     3$              ;SKIP IF LOOP ON ERROR
2261  017026  105037  001001                  CLRB    ICNT+1          ;CLEAR ITERATION COUNT
2262  017032  000002                          RTI                     ;RETURN
2263  017034  000137  015666          3$:     JMP     .KIT            ;LOOP ON TEST UNTIL NO ERRORS
2264
2265  016040  000000                  .HLTCT: 0                       ;HLT ARGUMENT
```

```
                                .SBTTL          $OCTAL - OCTAL TYPEOUT ROUTINE

                                ;THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE
                                ;ALL 6 CHARACTERS, SUPPRESS LEADING ZEROES, OR TYPE THE
                                ;16 BITS. IT IS CALLED VIA THE TYPOCT, TYPBIT, OR TYPOCS MACRO'S.

016042  012737  170101  016230 .TYPEB: MOV     #170101,.PR     ;SET BIT FLAG AND 16. CHARACTER COUNT
016050  000411                         BR      .PTIT           ;NOW TYPE IT IN BIT FORM
016052  112737  000001  016230 .TYPEO: MOVB    #1,.PR          ;SET ZERO FILL SWITCH
016060  000402                         BR      .+6             ;SKIP
016062  005037  016230         .TYPES: CLR     .PR             ;SUPRESS LEADING ZERO'S
016066  112737  177772  016231         MOVB    #-6,.PR+1       ;SET COUNT
016074                         .PTIT:
016074  010446                         MOV     R4,-(6)         ;PUSH R4 ON STACK
016076  010546                         MOV     R5,-(6)         ;PUSH R5 ON STACK
016100  016605  000010                 MOV     10(6),R5        ;GET THE DATA
016104  012704  016232                 MOV     #,PR+2,R4       ;SET POINTER TO FIRST ASCII CHAR.
016110  105014                         CLRB    (4)             ;CLEAR FIRST BYTE
016112  000411                         BR      .PRF            ;ROTATE FIRST BIT
016114  105014         .PRL:   CLRB    (4)             ;CLEAR BYTE OF CHARACTER
016116  032737  000100  016230         BIT     #100,.PR        ;BIT TYPING MODE?
016124  001004                         BNE     .PRF            ;YES - SKIP 2 ROTATES
016126  006105                         ROL     R5              ;ROTATE BIT INTO C
016130  106114                         ROLB    (4)             ;PACK IT
016132  006105                         ROL     R5              ;ROTATE BIT INTO C
016134  106114                         ROLB    (4)             ;PACK IT
016136  006105                 .PRF:   ROL     R5              ;ROTATE BIT INTO C
016140  106114                         ROLB    (4)             ;PACK IT
016142  105714                         TSTB    (4)             ;IS IT ZERO?
016144  001402                         BEQ     .+6             ;SKIP INC
016146  105237  016230                 INCB    .PR             ;SET FILL SWITCH
016152  105737  016230                 TSTB    .PR             ;CHECK FILL SWITCH
016156  001402                         BEQ     .+6             ;SKIP BISET
016160  152724  000060                 BISB    #'0,(4)+        ;MAKE INTO ASCII CHAR
016164  105237  016231                 INCB    .PR+1           ;INC COUNT
016170  001351                         BNE     .PRL            ;REPEAT
016172  022704  016232                 CMP     #.PR+2,R4       ;EMPTY BUFFER?
016176  001002                         BNE     .+6             ;SKIP IF NOT
016200  112724  000060                 MOVB    #'0,(4)+        ;LOAD 1 ZERO
016204  105014                         CLRB    (4)             ;NULL TERMINATOR
016206  104402  016232                 TYPE    ,.PR+2          ;TYPE IT
016212  012605                         MOV     (6)+,R5         ;POP STACK INTO R5
016214  012604                         MOV     (6)+,R4         ;POP STACK INTO R4
016216  016666  000002  000004         MOV     2(6),4(6)       ;GET RID OF
016224  012616                         MOV     (6)+,(6)        ;DATA WORD
016226  000002                         RTI                     ;RETURN

016230  000012         .PR:    .BLKW   12              ;COUNT, SWITCH, AND OUTPUT BUFFER
```

```
2314                                    .SBTTL          $POWER - POWER DOWN AND UP ROUTINES
2315
2316                                    ;THIS IS THE POWER FAIL ROUTINE WHICH WILL SAVE ALL
2317                                    ;THE GENERAL REGISTERS AND USER DEFINED REGISTERS THEN
2318                                    ;WAIT FOR POWER TO GO DOWN AND BE RESTORED.
2319                                    ;IF THERE ISN'T ENOUGH TIME FOR SAVING ALL THE REGISTERS,
2320                                    ;THE PROGRAM WILL HALT AT '.ILLUP'.
2321
2322  016254  012777  016402  000126  .POWER: MOV     #.ILLUP,@#PUVEC ;SET FOR FAST UP
2323  016262  012777  000340  000122          MOV     #340,@#PUVEC+2  ;PRIO:7
2324  016270  010046                          MOV     R0,-(6)         ;PUSH R0 ON STACK
2325  016272  010146                          MOV     R1,-(6)         ;PUSH R1 ON STACK
2326  016274  010246                          MOV     R2,-(6)         ;PUSH R2 ON STACK
2327  016276  010346                          MOV     R3,-(6)         ;PUSH R3 ON STACK
2328  016300  010446                          MOV     R4,-(6)         ;PUSH R4 ON STACK
2329  016302  010546                          MOV     R5,-(6)         ;PUSH R5 ON STACK
2330  016304  010637  016406                  MOV     SP,.SAVR6       ;SAVE SP
2331  016310  012777  016320  000072          MOV     #.POWUP,@#PUVEC ;SET UP VECTOR
2332  016316  000000                          HALT                    ;WAIT FOR PF
2333
2334  016320  013706  016406          .POWUP: MOV     .SAVR6,SP       ;GET SP
2335  016324  005001                          CLR     R1              ;WAIT LOOP FOR THE TTY
2336  016326  005201                  10:     INC     R1              ;WAIT FOR THE INC
2337  016330  001376                          BNE     10              ;OF  WORD
2338  016332  012605                          MOV     (6)+,R5         ;POP STACK INTO R5
2339  016334  012604                          MOV     (6)+,R4         ;POP STACK INTO R4
2340  016336  012603                          MOV     (6)+,R3         ;POP STACK INTO R3
2341  016340  012602                          MOV     (6)+,R2         ;POP STACK INTO R2
2342  016342  012601                          MOV     (6)+,R1         ;POP STACK INTO R1
2343  016344  012600                          MOV     (6)+,R0         ;POP STACK INTO R0
2344  016346  012737  016254  000024          MOV     #.POWER,@#24    ;SET UP THE POWER DOWN VECTOR
2345  016354  012737  000340  000026          MOV     #340,@#26       ;PRIO:7
2346  016362  104402  016366                  TYPE    ,.+2            ;.ASCIZ <15><12>"POWER"
2347  016376  000137  007012                  JMP     TIMUP           ;JMP TO USER ADDRESS
2348
2349  016402  000000          .ILLUP: HALT                    ;THE POWER UP SEQUENCE WAS STARTED
2350  016404  000776                  BR      .-2             ; BEFORE THE POWER DOWN WAS COMPLETE
2351
2352  016406  000000          .SAVR6: 0                       ;PUT THE SP HERE
2353  016410  000024  000026  .PUVEC: 24,26                   ;POWER UP VECTOR
```

```
                                        .SBTTL          $TYPEA - 18 BIT ADDRESS TYPER
                                        ;THIS ROUTINE TAKES 2 ARGUMENTS OFF THE STACK (OLD
                                        ;SP AND ADDRESS) AND, USING THE MEMORY MANAGEMENT REGISTERS, TYPES
                                        ;THE ADDRESS SUPPLIED IN 18 BIT FORM. THIS ROUTINE IS LINKED
                                        ;VIA THE 'TYPADR' MACRO.

016414                                  .TYPEA:
016414  010446                                MOV     R4,-(6)         ;PUSH R4 ON STACK
016416  010546                                MOV     R5,-(6)         ;PUSH R5 ON STACK
016420  016605  000012                        MOV     12(6),R5        ;R5 = OLD PS WITH PREVIOUS MODE
016424  016604  000010                        MOV     10(6),R4        ;R4 = ADDRESS TO BE DECODED AND TYPED
016430  016666  000006  000010                MOV     6(6),10(6)      ;MOVE
016436  016666  000004  000006                MOV     4(6),6(6)       ;DOWN
016444  016666  000002  000004                MOV     2(6),4(6)       ;FOUR
016452  012616                                MOV     (6)+,(6)        ;WORDS
016454  010346                                MOV     R3,-(6)         ;PUSH R3 ON STACK
016456  000305                                SWAB    R5              ;GET THE
016460  006005                                ROR     R5              ;2 PREVIOUS
016462  006005                                ROR     R5              ;MODE BITS
016464  006005                                ROR     R5              ;INTO POSITION
016466  042705  177771                        BIC     #177771,R5      ;TO USE AS AN OFFSET
016472  016505  016612                        MOV     .SATAB(5),R5    ;R5 = SPACE ADDRESS FOR MM
016476  010403                                MOV     R4,R3           ;R3 = REGISTER OFFSET
016500  042704  160000                        BIC     #160000,R4      ;CLEAR THE MM REG SELECT BITS
016504  000303                                SWAB    R3              ;NOW MAKE
016506  006003                                ROR     R3              ;MM REG
016510  006003                                ROR     R3              ;SELECT BITS
016512  006003                                ROR     R3              ;INTO AN
016514  006003                                ROR     R3              ;OFFSET
016516  042703  177761                        BIC     #177761,R3      ;CLEAR THE JUNK BITS
016522  060305                                ADD     R3,R5           ;ADD THE OFFSET TO THE TABLE
016524  011505                                MOV     (5),R5          ;GET THE ISAR DATA
016526  006305                                ASL     R5              ;THIS IS
016530  006305                                ASL     R5              ;TO SHIFT
016532  006305                                ASL     R5              ;THE SEGMENT
016534  006305                                ASL     R5              ;ADDRESS
016536  005003                                CLR     R3              ;INTO AN
016540  006305                                ASL     R5              ;AN 18 BIT
016542  006103                                ROL     R3              ;ADDRESS
016544  006305                                ASL     R5              ;POSITION
016546  006103                                ROL     R3              ;WITH R3 CONTAINING
016550  060405                                ADD     R4,R5           ;THE UPPER 2 BITS
016552  005503                                ADC     R3              ;AND R5 CONTAINING
016554  006305                                ASL     R5              ;THE 16 BIT ADDRESS
016556  006103                                ROL     R3              ;THEN SHIFT FOR TYPING
016560  110337  016232                        MOVB    R3,.PR+2        ;GET THE FIRST NUMBER FROM R3
016564  062737  000060  016232                ADD     #'0,.PR+2       ;MAKE IT INTO A NUMBER
016572  012704  016233                        MOV     #.PR+3,R4       ;FUDGE IN THE POINTER
016576  012737  178401  016230                MOV     #178401,.PR     ;AND THE FLAGS (FILL & 5 BYTES)
016604  012603                                MOV     (6)+,R3         ;POP STACK INTO R3
016606  000137  016114                        JMP     .PRL            ;DECODE AND TYPE THE REST

016612  172340                      .SATAB:   172340                  ;KISAR0
016614  172240                                172240                  ;SISAR0
016616  172340                                172340                  ;KISAR0 - NEVER USED
```

```
  2410  016620  177640                          177640                  ;UISAR0
```

```
2411                                    .SBTTL          $TRAP - TRAP HANDLER
2412
2413                                    ;THIS ROUTINE DECODES A TRAP CALL AND JUMPS TO THE APROPRATE
2414                                    ;SUBROUTINE. THE CALL IS A "TRAP+N" WHERE N IS A MULTIPLE OF 2.
2415                                    ;THE "SET" MACRO WILL CREATE THE TABLE NEEDED. IT HAS TO
2416                                    ;FOLLOW THIS MACRO.
2417
2418  016622  011646                    .TRAP:  MOV     (6),-(6)        ;GET ADDRESS OF TRAP +2
2419  016624  162716  000002                    SUB     #2,(6)          ;MAKE IT ADDRESS OF TRAP
2420  016630  017616  000000                    MOV     @(6),(6)        ;GET TRAP INSTRUCTION
2421  016634  062716  112242                    ADD     #,TRP+2-TRAP,(6);GET DATA AND MAKE IT AN OFFSET
2422  016640  013667                    .TRP:   MOV     @(6)+,PC        ;GO TO PROPER SUBROUTINE
2423
2424  016642  015542                            .SCOPE                  ;SCOPE   =  TRAP+0      (104400)
2425  016644  015404                            .TYPE                   ;TYPE    =  TRAP+2      (104402)
2426  016646  016052                            .TYPEO                  ;TYPEO   =  TRAP+4      (104404)
2427  016650  016062                            .TYPES                  ;TYPES   =  TRAP+6      (104406)
2428  016652  021226                            .TYPED                  ;TYPED   =  TRAP+10     (104410)
2429  016654  016414                            .TYPEA                  ;TYPEA   =  TRAP+12     (104412)
2430  016656  006120                            .ERCLR                  ;ERCLR   =  TRAP+14     (104414)
2431  016660  006142                            .DKCMD                  ;DKCMD   =  TRAP+16     (104416)
2432  016662  017264                            .RDOCT                  ;RDOCT   =  TRAP+20     (104420)
2433  016664  016710                            .RDLIN                  ;RDLIN   =  TRAP+22     (104422)
2434  016666  021172                            .UPDAT                  ;UPDAT   =  TRAP+24     (104424)
2435  016670  000316                            .CLRDV                  ;CLRDV   =  TRAP+26     (104426)
2436  016672  014106                            .LOGW                   ;LOGW    =  TRAP+30     (104430)
2437  016674  014116                            .LOGR                   ;LOGR    =  TRAP+32     (104432)
2438  016676  014124                            .LOGWC                  ;LOGWC   =  TRAP+34     (104434)
2439  016700  014136                            .LOGC                   ;LOGC    =  TRAP+36     (104436)
2440  016702  017174                            .CNTLU                  ;CNTLU   =  TRAP+40     (104440)
2441  016704  017112                            .KBDIN                  ;KBDIN   =  TRAP+42     (104442)
2442  016706  020000                            .SUSWR                  ;SUSWR   =  TRAP+44     (104444)
```

```
2443                                    .SBTTL          $RDLIN - TTY INPUT ROUTINE
2444
2445                                    ;THIS ROUTINE INPUTS A LINE TERMINATED BY A RETURN INTO ADDRESS
2446                                    ;INPUT AND RETURNS A LINE FEED. THE BUFFER HAS A NULL TERMINATOR
2447                                    ;INSTEAD OF THE RETURN. RUBOUTS ARE HANDLED BY RETYPING
2448                                    ;THE LINE. BUFFER OVERFLOW ERRORS LIKE A RUBOUT.
2449
2450  016710  010546                    .RDLIN: MOV     R5,-(6)         ;SAVE R5
2451  016712  012705  017032            1$:     MOV     #INPUT,R5       ;GET ADDRESS
2452  016716  022705  017052            2$:     CMP     #INPUT+16.,R5   ;BUFFER FULL?
2453  016722  001423                            BEQ     4$              ;YES - TYPE "?"
2454  016724  105737  177560                    TSTB    @#177560        ;WAIT FOR
2455  016730  100375                            BPL     .-4             ;A CHARACTER
2456  016732  113715  177562                    MOVB    @#177562,(5)    ;GET CHARACTER
2457  016736  142715  000200                    BICB    #200,(5)        ;GET RID OF JUNK
2458  016742  122715  000025                    CMPB    #25,(5)         ;IS IT A ^U
2459  016746  001006                            BNE     5$              ;BRANCH IF NOT
2460  016750  104402  016754                    TYPE    ,.+2            ;.ASCIZ "^U"<15><12>
2461  016762  000753                            BR      1$              ;START OVER
2462  016764  122715  000177            5$:     CMPB    #177,(5)        ;IS IT A RUBOUT
2463  016770  001005                            BNE     3$              ;SKIP IF NOT
2464  016772                            4$:
2465  016772  104402  016776                    TYPE    ,.+2            ;.ASCIZ "?"<15><12>
2466  017002  000743                            BR      1$              ;ZAP THE BUFFER AND LOOP
2467  017004  111527  000000            3$:     MOVB    (5),@0          ;SET UP FOR TYPING
2468  017010  104402  017006                    TYPE    ,3$+2           ;ECHO IT
2469  017014  122725  000015                    CMPB    #15,(5)+        ;CHECK FOR RETURN
2470  017020  001336                            BNE     2$              ;LOOP IF NOT RETURN
2471  017022  104402  000012                    TYPE    ,12             ;TYPE A LINE FEED
2472  017026  012605                            MOV     (6)+,R5         ;RESTORE R5
2473  017030  000002                            RTI                     ;RETURN
2474
2475  017032  000020            INPUT:  .BLKB   16.             ;TTY INPUT AREA
2476  017052  000020            ERR/$:  .BLKW   16.
2477
2478  017112  005737  000042    .KBDIN: TST     42              ;GOT XXDP OR ACT
2479  017116  001057                    BNE     OKT             ;YES,GET OUT
2480  017120  022737  000176  001026            CMP     $SWREG,SWR      ;GOT SWITCH-LESS MACHINE?
2481  017126  001053                            BNE     OKT             ;NO GET OUT
2482  017130  105777  161664                    TSTB    @TKS            ;HAVE A CHARACTER
2483  017134  100050                            BPL     OKT             ;NO GET OUT
2484  017136  017737  161660  017260            MOV     @TKB,.MSG
2485  017144  042737  177600  017260            BIC     #177600,.MSG    ;STRIP OFF GARBAGE
2486  017152  122737  000007  017260            CMPB    #7,.MSG         ;DO WE HAVE A ^G
2487  017160  001036                            BNE     OKT             ;NO,GET OUT
2488  017162  104402  017166                    TYPE    ,.+2            ;.ASCIZ <15><12>"^G"
2489  017174                            .CNTLU:
2490  017174  104402  017200                    TYPE    ,.+2            ;.ASCIZ <15><12>"SWR= "
2491  017210  013746  000176                    MOV     $SWREG,-(6)     ;PUT $SWREG ON STACK
2492  017214  104404                            TYPEO                   ;TYPE STACK IN OCTAL
2493  017216  104402  017222                    TYPE    ,.+2            ;.ASCIZ "    NEW= "
2494  017234  104420                            RDOCT
2495  017236  012637  017260                    MOV     (SP)+,.MSG      ;GET NEW VALUE OFF STACK
2496  017242  005737  017262                    TST     CTR             ;DID WE TYPE <CR> OF 000000?
2497  017246  001403                            BEQ     OKT             ;DON'T CHANGE IF <CR>
2498  017250  013737  017260  000176            MOV     .MSG,$SWREG     ;CHANGE VALUE OF $SWREG
```

```
017256  000002          OKT:    RTI                     ;ALL DONE-EXIT

017260  000000          .MSG:   0
017262  000000          CTN:    0
                        .SBTTL          $RDOCT - OCTAL INPUT ROUTINE

                        ;THIS ROUTINE CALLS RDLIN, INPUTS A LINE FROM THE TTY AND CONVERTS
                        ;IT INTO AN OCTAL NUMBER WHICH IS THE FIRST WORD ON THE STACK.

017264  011646          .RDOCT: MOV     (6),-(6)        ;MOVE THE PC
017266  016666  000004  000002          MOV     4(6),2(6)       ;MOVE THE PS
017274  010146                  MOV     R1,-(6)         ;PUSH R1 ON STACK
017276  010246                  MOV     R2,-(6)         ;PUSH R2 ON STACK
017300  010346                  MOV     R3,-(6)         ;PUSH R3 ON STACK
017302  104422                  4$:     RDLIN                   ;READ A LINE INTO INPUT
017304  005001                  CLR     R1              ;INIT DATA WORD
017306  005037  017262                  CLR     CTN             ;CLEAR COUNT WORD
017312  012703  017032                  MOV     $INPUT,R3       ;INIT POINTER
017316  112302                  1$:     MOVB    (3)+,R2         ;GET A BYTE
017320  122702  000015                  CMPB    #15,R2          ;WAS IT A ,CR?
017324  001421                  BEQ     2$              ;GET OUT IF YES
017326  122702  000060                  CMPB    #'0,R2          ;CHECK FOR 0 OR GREATER
017332  003024                  BGT     3$              ;ERROR - LESS THAN 0
017334  122702  000067                  CMPB    #'7,R2          ;CHECK FOR 7 OR LESS
017340  002421                  BLT     3$              ;ERROR - GREATER THAN 7
017342  006002                  ROR     R2              ;GET
017344  006002                  ROR     R2              ;INTO
017346  006002                  ROR     R2              ;POSITION
017350  006101                  ROL     R1              ;FIRST BIT
017352  006102                  ROL     R2              ;GET
017354  006101                  ROL     R1              ;SECOND BIT
017356  006102                  ROL     R2              ;GET
017360  006101                  ROL     R1              ;THIRD BIT
017362  005237  017262                  INC     CTN             ;YES HE TYPED SOMETHING
017366  000753                  BR      1$              ;LOOP
017370  010166  000012          2$:     MOV     R1,12(6)        ;SAVE THE RESULT
017374  012603                  MOV     (6)+,R3         ;POP STACK INTO R3
017376  012602                  MOV     (6)+,R2         ;POP STACK INTO R2
017400  012601                  MOV     (6)+,R1         ;POP STACK INTO R1
017402  000002                  RTI                     ;RETURN

017404                  3$:
017404  104402  017410                  TYPE    ,.+2            ;.ASCIZ "?"<15><12>
017414  000732                  BR      4$              ;TRY AGAIN
017416  000000          OUTBUF: 0

                        ;NOTE FOR PROGRAMMER********* PROGRAM AT THIS POINT CAN NOT EXCEED A PC OF 17466*****
```

```
2546            020000                  .=      20000
2547                                    ;NOTE    ALL THIS CODE GETS DESTROYED WHEN PATTERN IS WRITTEN
2548
2549  020000  032737  000001  020116  .SUSWR: BIT     #BIT0,SWI
2550  020006  001037                          BNE     XXX
2551  020010  013746  000006                  MOV     6,-(SP)         ;SAVE 6 ON STACK
2552  020014  013746  000004                  MOV     4,-(SP)         ;SAVE 4 ON STACK
2553  020020  012737  020040  000004          MOV     #16,4           ;SET UP TRAP ADDRESS
2554  020026  022777  177777  160772          CMP     @-1,@SWR        ;TEST 177570
2555  020034  001402                          BEQ     2$              ;FAKE OUT
2556  020036  000407                          BR      3$              ;HARDWARE AVAILABLE
2557  020040  022626                  1$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
2558  020042  012737  000176  001026  2$:     MOV     #SWREG,SWR      ;SET UP SOFTWARE REGISTERS
2559  020050  012737  000174  001030          MOV     #DISPREG,DISPLAY
2560  020056  022737  000176  001026  3$:     CMP     #SWREG,SWR      ;1ST TIME THRU?
2561  020064  001004                          BNE     4$              ;NO CHANGE STILL 177570
2562  020066  005737  000042                  TST     42              ;ANY XXDP OR ACT
2563  020072  001001                          BNE     4$              ;SWR=000000
2564  020074  104440                          CNTLU                   ;GET INITIAL. SETTINGS
2565  020076  012637  000004          4$:     MOV     (SP)+,4         ;REPLACE 4 FROM STACK
2566  020102  012637  000006                  MOV     (SP)+,6         ;REPLACE 6 FROM STACK
2567  020106  052737  000001  020116  XXX:    BIS     #BIT0,SWI       ;SET THE BEENHEREBIT
2568  020114  000002                          RTI                     ;ALL DONE
2569
2570  020116  000000                  SWI:    0
2571
2572
2573
2574                                    ;ROUTINE TO SAVE ABS LOADER
2575  020120  012700  017776          LDR:    MOV     #17776,R0
2576  020124  012737  020144  000004          MOV     #2$,4           ;SET TIME OUT TRAP VECTOR
2577  020132  012737  000340  000006          MOV     #340,6
2578  020140  005720                          TST     (R0)+
2579  020142  000776                          BR      .-2
2580  020144  022626                  2$:     CMP     (SP)+,(SP)+
2581  020146  012737  000006  000004          MOV     #6,4
2582  020154  005037  000006                  CLR     6
2583  020160  162700  000334                  SUB     #334,R0         ;POINT R0 BACK TO LOADER
2584  020164  010037  015200                  MOV     R0,LDR1         ;SAVE FOR RESTORE ROUTINE
2585  020170  012702  000155                  MOV     #155,R2         ;WORD COUNT
2586  020174  012703  017446                  MOV     #17446,R3       ;WHERE LOADER IS TO BE STORED
2587  020200  012023                  1$:     MOV     (R0)+,(R3)+     ;STORE LOADER
2588  020202  005302                          DEC     R2
2589  020204  001375                          BNE     1$
2590  020206  000207                          RTS     PC              ;RETURN
2591
2592
2593                                    ; -A- PORT SIZE
2594
2595  020210  052737  020000  001122  SIZEAP: BIS     #BIT13,FLAG2    ;SET MAPPING BIT
2596  020216  004737  001624                  JSR     PC,DRVENO       ;FIND DRIVE
2597  020222  012737  000002  001224          MOV     #2,WORK1        ;START WITH ONE 4K BUFFER
2598  020230  012737  000001  001070          MOV     #1,STAMEM       ;FIRST 4K BANK
2599  020236  012737  057476  001140          MOV     #57476,BUF      ;GET STARTING ADDR. 8K
2600  020244  012737  000001  001130          MOV     #1,WRDCT        ;LOAD WC
2601  020252  005037  001134                  CLR     DMA             ;LOAD DA
```

```
 2602  020256  012777  000040  160550          MOV     #40,@RSCS2      ;CLEAR ALL RS REG
 2603  020264  013777  001160  160542          MOV     UNNUM,@RSCS2    ;GET DRIVE #
 2604  020272  012737  000071  001172          MOV     #71,CMD         ;DO A READ
 2605  020300  104416                  40$:    DKCMD                   ;NOW
 2606  020302  105777  160524          10$:    TSTB    @RSCS1          ;DONE YET?
 2607  020306  100375                          BPL     10              ;NO
 2608  020310  032777  004000  160516          BIT     #4000,@RSCS2    ;DID NEM SET?
 2609  020316  001012                          BNE     $IS1            ;YES
 2610  020320  005777  160506                  TST     @RSCS1          ;ANY ERRORS?
 2611  020324  100005                          BPL     30              ;NO
 2612  020326  012737  000006  001106          MOV     #6,SIZEAP       ;GET SIZE OF BUFFER
 2613  020334  000137  020526                  JMP     @#$SIZERR       ;FOR USER IF HE WISHES IT
 2614  020340  104424                  30$:    UPDAT                   ;GET NEXT 4K BANK
 2615  020342  000756                          BR      40              ;TEST BANK
 2616  020344  005337  001224          $IS1:   DEC     WORK1           ;DEC SIZE OF BUFFER
 2617  020350  013737  001224  001106          MOV     WORK1,SIZEAP    ;LOAD SIZE OF A BUFFER
 2618  020356  104402  020362                  TYPE    .,.+2           ;.ASCII <15><12> "PORT -A- DATA BUFFER     4K TO "
 2619  020424  004737  021132                  JSR     PC,$IZPR
```

```
                                        ; -B- PORT SIZE
                                        ;
020430  012737  000001  001224  SIZEBP: MOV     #1,WORK1        ;START WITH ONE 4K BUFFER
020436  012737  037476  001140          MOV     #37476,BUF      ;GET STARTING ADDR. 4K
020444  012737  000001  001130          MOV     #1,WRDCT        ;LOAD WC
020452  005037  001134                  CLR     DMA             ;LOAD DA
020456  012777  000040  160350          MOV     #40,@RSCS2      ;CLEAR ALL RS REG
020464  013777  001160  160342          MOV     UNNUM,@RSCS2    ;SET DRIVE #
020472  012737  002071  001172          MOV     #2071,CMD       ;DO A READ
020500  104416                  SIZ2:   DKCMD                   ;NOW
020502  105777  160324          1$:     TSTB    @RSCS1          ;DONE YET?
020506  100375                          BPL     1$              ;NO
020510  032777  004000  160316          BIT     #4000,@RSCS2    ;DID NEM SET?
020516  001072                          BNE     SIZE1           ;YES
020520  005777  160306                  TST     @RSCS1          ;ANY ERRORS?
020524  100047                          BPL     SIZ3            ;NO
020526                          SIZERR:
020526  104402  020532                  TYPE    ,.+2            ;,ASCIZ <18><12>"WILL NOT CONTINUE TO SIZE MEMORY BECAUS
020616  012737  000006  001110          MOV     #6,SIZEBP       ;GIVE PROGRAM A BUFFER
020624  104060                          MLT     ;BAIDS          ;YOU CAN ENTER CONVERSATION MODE
020626  052737  000001  001122          BIS     #BIT0,FLAG2     ;BEEN HERE BEFORE FLAG
020634  042737  020000  001122          BIC     #BIT13,FLAG2    ;CLEAR MAPPING FLAG
020642  000000                          HALT                    ;OR GO TO BEGAS
020644  032737  000400  001122  SIZE1:  BIT     #BIT8,FLAG2     ;FOUND MEMORY YET?
020652  001011                          BNE     SIZE3           ;YES
020654  052737  000400  001122          BIS     #BIT8,FLAG2     ;SET FOUND NEM FLAG
020662  013737  001224  001074          MOV     WORK1,STSCON    ;GET 1ST BANK
020670  012737  000001  001224          MOV     #1,WORK1
```

```
 2648  020676  104424                  SIZB3:  UPDAT                   ;GET NEXT 4K BANK
 2649  020700  000677                          BR      SIZ2            ;TEST BANK
 2650  020702  000404                          BR      SIZB2
 2651  020704  032737  000400  001122  SIZB1:  BIT     #BIT8,FLAG2     ;FOUND MEMORY?
 2652  020712  001771                          BEQ     SIZB3           ;NO
 2653  020714  005337  001224          SIZB2:  DEC     WORK1           ;DEC SIZE OF BUFFER

 2654  020720  013737  001224  001110          MOV     WORK1,SIZEBP    ;LOAD SIZE OF B BUFFER
 2655  020726  032737  000400  001122          BIT     #BIT8,FLAG2     ;FOUND MEMORY?
 2656  020734  001017                          BNE     66              ;YES
 2657  020736  104402  020742                  TYPE    ,.+2            ;.ASCIZ <15><12>"NO MEMORY ON -B- PORT"
 2658  020772  000447                          BR      10
 2659  020774                          66:
 2660  020774  104402  021000                  TYPE    ,.+2            ;.ASCIZ <15><12> "PORT -B- DATA BUFFER"
 2661  021030  013737  001074  001224          MOV     $TBCOM,WORK1
 2662  021036  005001                          CLR     R1
 2663  021040  005002                          CLR     R2
 2664  021042  004737  021140                  JSR     PC,SIZP
 2665  021046  104402  021052                  TYPE    ,.+2            ;.ASCIZ " TO "
 2666  021060  013737  001110  001224          MOV     SIZEBP,WORK1
 2667  021066  063737  001074  001224          ADD     $TBCOM,WORK1
 2668  021074  005337  001224                  DEC     WORK1
 2669  021100  004737  021132                  JSR     PC,SIZPR
 2670  021104  052737  000100  001126          BIS     #BIT6,FLAG      ;SET MULTI PORT FLAG
 2671  021112  042737  020000  001122  10:     BIC     #BIT13,FLAG2    ;CLEAR MAPPING FLAG
 2672  021120  052737  000002  001122          BIS     #BIT1,FLAG2     ;SET BEEN HERE FLAG
 2673  021126  000137  001474                  JMP     CALM            ;CAL BUFFER AND MC
 2674
 2675  021132  005001                  SIZPR:  CLR     R1              ;INIT SETUP
 2676  021134  012702  000004                  MOV     #4,R2
 2677  021140  062701  000001          SIZP:   ADD     #1,R1           ;SETUP FOR BANK NO
 2678  021144  062702  000004                  ADD     #4,R2           ;SETUP FOR SIZE PG MEMORY
 2679  021150  020137  001224                  CMP     R1,WORK1        ;IS THIS THE RIGHT SIZE?
 2680  021154  001371                          BNE     SIZP            ;NO
 2681  021156  010246                          MOV     R2,-(6)         ;PUT R2 ON STACK
 2682  021160  104410                          TYPED                   ;TYPE STACK IN DECIMAL
 2683  021162  104402  021166                  TYPE    ,.+2            ;.ASCIZ "K"
 2684  021170  000207                          RTS     PC              ;RETURN
 2685
 2686                                  ;ADD 4K TO TEST ADDR
 2687
 2688  021172  005237  001224          .UPDAT: INC     WORK1           ;INC BANK #
 2689  021176  062737  020000  001140          ADD     #20000,BUF      ;UPDATE BY 4K
 2690  021204  022737  177476  001140          CMP     #177476,BUF     ;EXCEEDED MEM YET?
 2691  021212  001401                          BEQ     10              ;YES
 2692  021214  000002                          RTI
 2693  021216  062716  000002          10:     ADD     #2,(6)          ;UPDATE RETURN PC BY 2
 2694  021222  000002                          RTI
 2695
 2696  021224  021226                          .TYPED                  ;TYPED   9  TRAP+46      (104446)
 2697                                  .SBTTL          $TYPED - CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 2698
 2699  021226  012737  100040  021454  .TYPED: MOV     #100040,.BSIGN  ;SET BLANK SWITCH AND SIGN
 2700  021234  010046                          MOV     R0,-(6)         ;PUSH R0 ON STACK
 2701  021236  010146                          MOV     R1,-(6)         ;PUSH R1 ON STACK
 2702  021240  010246                          MOV     R2,-(6)         ;PUSH R2 ON STACK
 2703  021242  010346                          MOV     R3,-(6)         ;PUSH R3 ON STACK
```

```
2704  021244  010546                        MOV     R5,-(6)          ;PUSH R5 ON STACK
2705  021246  012737  100040  021454        MOV     #100040,.DSIGN   ;SET BLANK SWITCH AND SIGN
2706  021254  016605  000016                MOV     16(6),R5         ;GET DATA TO BE TYPED
2707  021260  100004                        BPL     1$               ;BR IF INPUT IS POS.
2708  021262  005405                        NEG     R5               ;MAKE THE BINARY NUMBER POS.
2709  021264  112737  000055  021454        MOVB    #'-,.DSIGN       ;MAKE THE ASCII NUMBER NEG.
2710  021272  005000                1$:     CLR     R0               ;ZERO THE CONSTANTS INDEX
2711  021274  012703  021444                MOV     #,DBLK,R3        ;SETUP THE OUTPUT POINTER
2712  021300  112723  000040                MOVB    #' ,(R3)+        ;SET THE FIRST CHARACTER TO A BLANK
2713  021304  005002                2$:     CLR     R2               ;CLEAR THE BCD NUMBER
2714  021306  016001  021434                MOV     .DTBL(R0),R1     ;GET THE CONSTANT
2715  021312  160105                3$:     SUB     R1,R5            ;FORM THIS BCD DIGIT
2716  021314  002402                        BLT     4$               ;BR IF DONE
2717  021316  005202                        INC     R2               ;INCREASE THE BCD DIGIT BY 1
2718  021320  000774                        BR      3$
2719  021322  060105                4$:     ADD     R1,R5            ;ADD BACK THE CONSTANT
2720  021324  005702                        TST     R2               ;CHECK IF BCD DIGIT=0
2721  021326  001003                        BNE     5$               ;FALL THROUGH IF 0
2722  021330  105737  021455                TSTB    .DSIGN+1         ;STILL DOING LEADING 0'S?
2723  021334  100410                        BMI     7$               ;BR IF YES
2724  021336  106337  021455                ASLB    .DSIGN+1         ;MSD?
2725  021342  103003                        BCC     6$               ;BR IF NO
2726  021344  113763  021454  177777        MOVB    .DSIGN,-1(R3)    ;YES--SET THE SIGN
2727  021352  052702  000060                BIS     #'0,R2           ;MAKE THE BCD DIGIT ASCII
2728  021356  052702  000040                BIS     #' ,R2           ;MAKE IT A SPACE IF NOT ALREADY A DIGIT
2729  021362  110223                        MOVB    R2,(R3)+         ;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2730  021364  005720                        TST     (R0)+            ;JUST INCREMENTING
2731  021366  020027  000010                CMP     R0,#10           ;CHECK THE TABLE INDEX
2732  021372  002744                        BLT     2$               ;GO DO THE NEXT DIGIT
2733  021374  003002                        BGT     8$               ;GO TO EXIT
2734  021376  010502                        MOV     R5,R2            ;GET THE LSD
2735  021400  000764                        BR      6$               ;GO CHANGE TO ASCII
2736  021402  105013                8$:     CLRB    (R3)             ;SET THE TERMINATOR
2737  021404  012605                        MOV     (6)+,R5          ;POP STACK INTO R5
2738  021406  012603                        MOV     (6)+,R3          ;POP STACK INTO R3
2739  021410  012602                        MOV     (6)+,R2          ;POP STACK INTO R2
2740  021412  012601                        MOV     (6)+,R1          ;POP STACK INTO R1
2741  021414  012600                        MOV     (6)+,R0          ;POP STACK INTO R0
2742  021416  016666  000002  000004        MOV     2(6),4(6)        ;FUDGE DATA
2743  021424  012616                        MOV     (6)+,(6)         ;OFF STACK
2744  021426  104402  021444                TYPE    ,.DBLK           ;NOW TYPE THE NUMBER
2745  021432  000002                        RTI                      ;RETURN
2746
2747  021434  023420  001750  000144  .DTBL: 10000.,1000.,100.,10.
2748  021442  000012
2749  021444  000004                .DBLK:  .BLKW   4
2750  021454  000000                .DSIGN: 0
2751
2752          000001                        .END
```

```
ADAM    010676          CMDAE   007334          FLAG2   001122          LDR     020120          PAT17   000312
ADDRR   014144          CMPLP   010566          FLAG3   001216          LDR1    015200          PAT2    000260
ADERR   003624          CMPLP1  010632          FNDTYP  006540          LOADSW  000510          PAT20   000314
ADHGT   003562          CMPY    003300          GETERR  014012          LOGC  = 104436          PAT3    000262
ADRD    004732          CNTLU = 104440          GOOD  =%000001          LOGR  = 104432          PAT4    000264
ADR1    004746          CONDAR  007240          HINUM   010206          LOGW  = 104430          PAT5    000266
ADTL    003316          CONERR  001210          HINUM1  010350          LOGWC = 104434          PAT6    000270
ADTST   003224          COMPAR  010446          HISAV   010212          LONUM   010204          PAT7    000272
ADT1    003324          CONN    002220          HLT   = 104000          LONUM1  010346          PC    =%000007
ADT1A   003326          CRLFLF  000636          HLTADR  001012          LOPCNT  001200          PCNT    001004
AKH     007352          CTN     017262          HNDER   001152          LOSAV   010210          PDOWN   000742
APORT   011350          DA    = 000004          ICNT    001000          MDON    012304          PFT1    012466
AS    = 000100          DATA    000554          INCSEC  007214          MEM     010352          PFT2    013014
ASKWC   002544          DATAT   003710          INPUT   017032          MMABTS  012306          PFWAT   012522
ASR1    001114          DATTES  002454          INTEXT  006444          MMAVA   001212          PFWATT  012514
A1      002374          DB    = 000002          INTFLG  001176          MMSET   011376          PHADDR  001120
BA    = 000020          DISBUF  007050          KBDIN = 104442          MMVEC = 000250          PHYCOV  006626
BAD   =%000000          DISPLA  001030          KDPAR0= 172360          MR    = 000220          PONFAL  013272
BEGIN   001234          DISPRE  000174          KDPAR1= 172362          MSTR    004764          PRIORI  001066
BELL  = 000007          DKCMD = 104416          KDPAR2= 172364          MYBYWR  012542          PRNT    014204
BIT0  = 000001          DKINT   006354          KDPAR3= 172366          N     = 000005          PS    = 177776
BIT1  = 000002          DMA     001134          KDPAR4= 172370          NOPORT  002362          PSW   = 177776
BIT10 = 002000          DONE    014044          KDPAR5= 172372          NOREC   006520          PTDONE  014720
BIT11 = 004000          DONEE   002204          KDPAR6= 172374          NOWGO   002210          PWRDN   013222
BIT12 = 010000          DOWN    012750          KDPAR7= 172376          NPRCNT  012426          PWRFL   013052
BIT13 = 020000          DROP    001124          KDPDR0= 172320          NPRRET  012416          PWRFL2  013042
BIT14 = 040000          DRP     014772          KDPDR1= 172322          NPR1    012430          PWRF1   013050
BIT15 = 100000          DRVENB  001624          KDPDR2= 172324          NUM8    001170          PWRUP   013232
BIT2  = 000004          DS    = 000040          KDPDR3= 172326          OKT     017256          QQ    = 000001
BIT3  = 000010          DSKRD   004570          KDPDR4= 172330          OPDAR   002710          RAND    010240
BIT4  = 000020          DT    = 000240          KDPDR5= 172332          OPDSEL  006500          RANDM   000716
BIT5  = 000040          DYNUM   001664          KDPDR6= 172334          OPPAT   003020          RANDOM  01[illegible]
BIT6  = 000100          ELM     004634          KDPDR7= 172336          OPRD    003142          RAND1   010044
BIT7  = 000200          ERCLR = 104414          KIPAR0= 172340          OPWCK   003172          RANEL   000650
BIT8  = 000400          ERCOUN  001146          KIPAR1= 172342          OPWRT   003112          RANEX   000[illegible]
BIT9  = 001000          ERRCL   012432          KIPAR2= 172344          OUT     013446          RDERR   00[illegible]
BLOCK   001154          ERRORS  001002          KIPAR3= 172346          OUTF 7  017416          RDLIN = 10[illegible]
BLSZ    007556          ERTAB   017052          KIPAR4= 172350          PARCSR= 172100          RDOCT = 10[illegible]
BPORT   011416          ESN     004524          KIPAR5= 172352          PARERR  000760          RD[illegible]
BUF     001140          ESN1    004530          KIPAR6= 172354          PARVEC= 000114          RE[illegible]
BUFEXI  007342          EXGEN   010200          KIPAR7= 172356          PASEL   007716          READER  001[illegible]
BUFINX  007426          EXGEN1  010244          KIPDR0= 172300          PASEX   010024          RECOV   0[illegible]
BUFSIZ  000674          EXNPLG  003240          KIPDR1= 172302          PASSC   001156          RESTOR  01[illegible]
CALM    001474          EXRAX   006010          KIPDR2= 172304          PATHU   001136          RETT    010[illegible]
CALM1   001500          EXRXX   006022          KIPDR3= 172306          PAT0    000254          RLDR    01[illegible]
CHCKDV  002150          EXTDR   007034          KIPDR4= 172310          PAT1    000256          RORBLK  007640
CHKADT  003650          EXTHEN  011472          KIPDR5= 172312          PAT10   000274          RRDSEC  003010
CHKDAT  012626          EXTPA   006114          KIPDR6= 172314          PAT11   000276          RREAD   003002
CHKDSK  013072          EXTPPR  006066          KIPDR7= 172316          PAT12   000300          RRR2    011730
CLEAR   011302          EXTT    011414          LA    = 000204          PAT13   000302          RSAB    001050
CLIND   014112          FILCHR  001014          LAB     001010          PAT14   000304          RSDA    001040
CLRDV = 104426          FILDAT  010014          LDAT    004022          PAT15   000306          RSC01   001032
CMD     001172          FLAG    001126          LDAT1   004016          PAT16   000310          RSC02   001034
```

DZRSCE.P11 SYMBOL TABLE

```
RSDA    001042          SIZERR  020526          SW6   =  000100          WC    =  000010          .HLT    015716
RSDB    001054          SIZP    021140          SW7   =  000200          WCCON   002600          .HLTCT  016040
RSDS    001044          SIZPR   021132          SW8   =  000400          WCERR   001204          .ILLUP  016402
RSDT    001060          SIZZAP  020210          SW9   =  001000          WCKERR  000574          .KBDIN  017112
RSER    001046          SIZZBP  020430          TAG     004630          WDCTB   001112          .KIT    015666
RSLA    001052          SIZ1    020344          TAG1    000702          WDERR   010706          .LOGC   014136
RSMR    001056          SIZ2    020500          TDMA    001142          WNTHU   000562          .LOGR   014116
RSREG   014214          SIZ3    020644          TIMES   015714          WORK    001222          .LOGW   014106
RSVEC   001062          SLN     004174          TINUP   007012          WORK1   001224          .LOGWC  014124
RSWC    001036          SLN2    004176          TKB     001022          WORK2   001226          .NAMK   015202
RS04DT  001166          SLN2A   004564          TKS     001020          WRDCMP  010642          .MSG    017260
RW    =  000006          SP    =%000006          TPB     001024          WRDCT   001130          .OPDR   000035
RWRED   005706          SR0   =  177572          TPS     001016          WRDINC  010650          .OVER   015672
R0    =%000000          SR1   =  177574          TRACK   001132          WRDWN   013366          .PARER  015274
R1    =%000001          SR2   =  177576          TRUERR  004436          WRERR   005270          .POWER  016264
R2    =%000002          SR3   =  172516          TRY     001044          WRITER  001202          .POWUP  016320
R3    =%000003          STABUF  004653          TRYNX   002134          WRLG1   005166          .PR     015230
R4    =%000004          STAMEM  001070          TSTNG   000730          WRMEXB  003506          .PRF    016136
R5    =%000005          STATUS  001064          TST1    003324          WRRCK   004430          .PRL    016114
SAMHT   003606          STBCOM  001074          TST2    003710          WRRCK1  004410          .PTIT   016074
SAVAST  001072          STMM3   012020          TST3    005056          WRYERR  000564          .PUTEC  016410
SAVCPU  001076          STTEST  001736          TST4    006064          WRXBL   004140          .QQ,  =  000050
SAVE    001150          SUSWR   104444          TYPE    104402          WRX1    004152          .RDLIN  016710
SAVMGA  001100          SWI     020116          TYPEA   104412          WTNO    011454          .RDOCT  017264
SAVMGB  001102          SWITCH  001174          TYPED   104446          XEAM    004242          .SATAB  016612
SAVMGC  001104          SWR     001026          TYPEO   104404          XINCAK  007434          .SAVR4  016406
SAVWC   001214          SWRDCT  001144          TYPES   104406          XINCAU  007462          .SCOPE  015542
SAVWCB  001220          SWREG   000176          TYPREC  011772          XSEABU  003432          .SUSWR  020000
SCOPE   104400          SW0   =  000001          UNCMP   001164          XWAIT   010312          .SVLAD  015646
SEABUF  003420          SW1   =  000002          UNITSV  001162          XXX     020106          .SWOPER 021000
SEEC    014602          SW10  =  002000          UNWUM   001160          YORN    000641          .TSIT   014104
SETIND  014130          SW11  =  004000          UNRECO  000522          $ENDAD  014070          .TRAP   016622
SETSWI  001230          SW12  =  010000          UNTYP   001746          $END1   014100          .TRP    016046
SHIFT   010072          SW13  =  020000          UP    =  000000          .CLRDV  004316          .TYPE   015404
SHIFT1  010256          SW14  =  040000          UPCHK   012612          .CNTLU  017174          .TYPEA  016414
SIZB1   020704          SW15  =  100000          UPDAT   104424          .DBLK   021444          .TYPEB  016042
SIZB2   020714          SW2   =  000004          UPP     012760          .DKCMD  006142          .TYPED  021226
SIZB3   020676          SW3   =  000010          VADDR   001116          .DSIGN  021404          .TYPEO  016062
SIZEAP  001106          SW4   =  000020          VECTRR  006774          .DTBL   021434          .TYPES  016062
SIZEBP  001110          SW5   =  000040          WAIT    011750          .ERCLR  006120          .UPDAT  021172
.       =  021456
```

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*,RSCE/SOL_DZRSCE.P11
RUN-TIME: 22 22 1 SECONDS
RUN-TIME RATIO: 87/4681.8
CORE USED: 12K (23 PAGES)