RP04/05/06

READ/WRITE TEST

MD-11-DZRJA-A

EP-DZRJA-A-DI-A

NOV 1978

FICHE 1 OF 1

MADE IN USA

1

>

>

2

```
                        .TITLE  MD-11-DZRJA-AA RPDH/5/6 MECHANICAL AND READ/WRITE TEST
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY C. HESS
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQRC-C2), SEPT 14, 1976.
                        ;*

                        .SBTTL  CONTROL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH  STATE           USE
                        ;*      ------  -----   ----------------------------------
                        ;*        15      0     WRITE PACK BEFORE TESTING (TEST 21)
                        ;*                1     INHIBIT WRITING PACK BEFORE TESTING (TEST 21)
                        ;*        14      0     NO STALL BETWEEN DRIVE FUNCTIONS
                        ;*                1     STALL AFTER EVERY DRIVE FUNCTION
                        ;*        13      0     USE SPECIFIC STALL TIME
                        ;*                1     USE RANDOM STALL
                        ;*        12      0     NO INCREMENTING STALL IN TEST 4
                        ;*                1     DO INCREMENTING STALL IN TEST 4
                        ;*         8      0     DO IMPLIED SEEKS WITH DATA TRANSFERS
                        ;*                1     DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
                        ;*         7      0     DO "READ HEADER AND DATA" IN TESTS 0-11
                        ;*                1     DO EXPLICIT SEEKS IN TESTS 0-11
                        ;*         6      0     60 HZ
                        ;*                1     50 HZ
                        ;*         5      0     RUN WATCHDOG TIMER
                        ;*                1     INHIBIT WATCHDOG TIMER
                        ;*         0      0     TEST DRIVE(S) IN 22 SECTOR (16 BIT) MODE
                        ;*                1     TEST DRIVE(S) IN 20 SECTOR (18 BIT) MODE

                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          ----------------- ---
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            PRINT ERROR MESSAGE ON LINE PRINTER
                        ;*         7            READ CONTROL SWITCH SETTINGS FROM TTY
                        ;*         6            INHIBIT TIME REPORTS (TESTS 12-15)
                        ;*         5            REPORT ONE ERROR PER SECTOR (TESTS 16 & 17)
                        ;*         4            INHIBIT WRITES (TEST 15)
                        ;*         3            INHIBIT WRITE CHECKS (TEST 20)
                        ;*         2            INHIBIT READ AND SOFTWARE COMPARES (TEST 20)
                        ;*         1            INHIBIT SOFTWARE COMPARES (TEST 20)
                        ;*         0            PERFORM READ AFTER WRITE CHECK ERROR (TEST 20)

                        .SBTTL  TRAP CATCHER

000000                          .=0
```

```
                                TRAP CATCHER

                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
            000174              .=174
000174  000000          DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000          SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER

                        .SBTTL  ACT11 HOOKS

                        ;;*****************************************************************
                        ;HOOKS REQUIRED BY ACT11
            000200              $SVPC=.                 ;SAVE PC
            000046              .=46
000046  017604                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
            000052              .=52
000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
            000200              .=$SVPC                 ;; RESTORE PC

                        .SBTTL  STARTING ADDRESSES
            000200      .=200
                        ;*200 = NORMAL START
000200  000137  004636          JMP     @#START1
                        ;*204 = SELECT OPERATING PARAMETERS
000204  000137  004660          JMP     @#START2
                        ;*210 = SELECT RH11/RP04/5/6 ADDRESSES
000210  000137  004626          JMP     @#START3
                        ;*214 = COMBINATION OF 204 AND 210
000214  000137  004650          JMP     @#START4

                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
            001100      STACK=  1100
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
            000011      HT=     11              ;;CODE FOR HORIZONTAL TAB
            000012      LF=     12              ;;CODE FOR LINE FEED
            000015      CR=     15              ;;CODE FOR CARRIAGE RETURN
            000200      CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
            177776      PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
            177774      STKLMT= 177774          ;;STACK LIMIT REGISTER
            177772      PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
            177570      DSWR=   177570          ;;HARDWARE SWITCH REGISTER
            177570      DCISP=  177570          ;;HARDWARE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
            000000      R0=     %0              ;;GENERAL REGISTER
            000001      R1=     %1              ;;GENERAL REGISTER
            000002      R2=     %2              ;;GENERAL REGISTER
            000003      R3=     %3              ;;GENERAL REGISTER
            000004      R4=     %4              ;;GENERAL REGISTER
            000005      R5=     %5              ;;GENERAL REGISTER
            000006      R6=     %6              ;;GENERAL REGISTER
```

BASIC DEFINITIONS

```
                    R7=     %7              ;;GENERAL REGISTER
                    SP=     %6              ;;STACK POINTER
                    PC=     %7              ;;PROGRAM COUNTER

                    ;*PRIORITY LEVEL DEFINITIONS
                    PR0=    0               ;;PRIORITY LEVEL 0
                    PR1=    40              ;;PRIORITY LEVEL 1
                    PR2=    100             ;;PRIORITY LEVEL 2
                    PR3=    140             ;;PRIORITY LEVEL 3
                    PR4=    200             ;;PRIORITY LEVEL 4
                    PR5=    240             ;;PRIORITY LEVEL 5
                    PR6=    300             ;;PRIORITY LEVEL 6
                    PR7=    340             ;;PRIORITY LEVEL 7

                    ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                    SW15=   100000
                    SW14=   40000
                    SW13=   20000
                    SW12=   10000
                    SW11=   4000
                    SW10=   2000
                    SW09=   1000
                    SW08=   400
                    SW07=   200
                    SW06=   100
                    SW05=   40
                    SW04=   20
                    SW03=   10
                    SW02=   4
                    SW01=   2
                    SW00=   1
                    .EQUIV  SW09,SW9
                    .EQUIV  SW08,SW8
                    .EQUIV  SW07,SW7
                    .EQUIV  SW06,SW6
                    .EQUIV  SW05,SW5
                    .EQUIV  SW04,SW4
                    .EQUIV  SW03,SW3
                    .EQUIV  SW02,SW2
                    .EQUIV  SW01,SW1
                    .EQUIV  SW00,SW0

                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                    BIT15=  100000
                    BIT14=  40000
                    BIT13=  20000
                    BIT12=  10000
                    BIT11=  4000
                    BIT10=  2000
                    BIT09=  1000
                    BIT08=  400
                    BIT07=  200
                    BIT06=  100
                    BIT05=  40
                    BIT04=  20
                    BIT03=  10
```

```
                        BIT02=  4
                        BIT01=  2
                        BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        ;;*************************************************************

                        .SBTTL  RH11 REGISTERS

                        ;;*************************************************************

                        ;CONTROL AND STATUS REGISTER 1 (RPCS1)

000100                  IE=     100             ;INTERRUPT ENABLE (BIT #6)
000200                  RDY=    200             ;READY (BIT #7)
000400                  A16=    400             ;HIGH ORDER BUS ADDRESS BIT (BIT #8)
001000                  A17=    1000            ;HIGH ORDER BUS ADDRESS BIT (BIT #9)
002000                  PSEL=   2000            ;PORT SELECT (BIT #10)
020000                  MXCE=   20000           ;MASSBUSS PARITY ERROR (BIT #13)
040000                  TRE=    40000           ;TRANSFER ERROR (BIT #14)
                        ;SC=    100000          ;SPECIAL CONDITION (BIT #15)

                        ;WORD COUNT REGISTER (RPWC)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        ;BUS ADDRESS REGISTER (RPBA)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        ;CONTROL AND STATUS REGISTER 2 (RPCS2)

000001                  US1=    1               ;UNIT SELECT (BIT #0)
000002                  US2=    2               ;UNIT SELECT (BIT #1)
000004                  US4=    4               ;UNIT SELECT (BIT #2)
000010                  BAI=    10              ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
```

```
                                    ;PAT=   20              ;MASSBUS PARITY TEST (BIT #4)
000040                              CLR=    40              ;CLEAR (BIT #5)
000100                              IR=     100             ;INPUT READY (BIT #6)
000200                              OR=     200             ;OUTPUT READY (BIT #7)
000400                              MPE=    400             ;MASS BUS PARITY ERROR (BIT #8)
001000                              MXF=    1000            ;MISSED TRANSFER ERROR (BIT #9)
002000                              PGE=    2000            ;PROGRAM ERROR (BIT #10)
004000                              NEM=    4000            ;NON EXISTENT MEMORY (BIT #11)
010000                              NED=    10000           ;NON EXISTENT DRIVE (BIT #12)
020000                              UPE=    20000           ;UNIBUS PARITY ERROR (BIT #13)
040000                              WCE=    40000           ;WRITE CHECK ERROR (BIT #14)
100000                              DLT=    100000          ;DATA LATE (BIT #15)

                                    ;DATA BUFFER REGISTER (RPDB)
                                    ;(EACH BIT IS CALLED BY BIT NUMBER)


                                    ;;*****************************************************************

                                    .SBTTL  RPO4/5/6 REGISTERS

                                    ;;*****************************************************************

                                    ;CONTROL AND STATUS 1 REGISTER. (#00)

000001                              GO=     1               ;GO BIT (BIT #0)
000002                              F1=     2               ;FUNCTION CODE BIT #1
000004                              F2=     4               ;FUNCTION CODE BIT #2
000010                              F3=     10              ;FUNCTION CODE BIT #3
000020                              F4=     20              ;FUNCTION CODE BIT #4
000040                              F5=     40              ;FUNCTION CODE BIT #5
004000                              DVA=    4000            ;DEVICE AVAILABLE (BIT #11)

                                    ;DRIVE STATUS REGISTER (RPDS1) (#01)

                                    ;DF5=   1               DRIVE FORWARD 5"/SEC.  (BIT #0)
000002                              DF20=   2               ;DRIVE FORWARD 20"/SEC.  (BIT #1)
000004                              DIGB=   4               ;DRIVE TO INNER GUARD BAND (BIT #2)
000010                              GRV=    10              ;GO REVERSE (BIT #3)
000020                              DL64=   20              ;DIFFERENCE LESS THAN 64 (BIT #4)
000040                              DE1=    40              ;DIFFERENCE EQUALS 1 (BIT #5)
000100                              VV=     100             ;VOLUME VALID (BIT #6)
000200                              DRY=    200             ;DRIVE READY (BIT #7)
000400                              DPR=    400             ;DRIVE PRESENT (BIT #8)
001000                              PGM=    1000            ;PROGRAMABLE (BIT #9)
002000                              LST=    2000            ;LAST SECTOR TRANSFERRED (BIT #10)
004000                              WRL=    4000            ;WRITE LOCK (BIT #11)
010000                              MOL=    10000           ;MEDIUM ON-LINE (BIT #12)
020000                              PIP=    20000           ;POSITIONING OPERATION IN PROGRESS (BIT #13)
040000                              ERR=    40000           ;COMPOSITE ERROR (BIT #14)
100000                              ATA=    100000          ;ATTENTION ACTIVE (BIT #15)

                                    ;ERROR REGISTER #01 (RPER1) (#02)

000001                              ILF=    1               ;ILLEGAL FUNCTION (BIT #0)
000002                              ILR=    2               ;ILLEGAL REGISTER (BIT #1)
```

```
000004          RMR=    4               ;REGISTER MODIFICATION REFUSED (BIT #2)
000010          PAR=    10              ;PARITY ERROR (BIT #3)
000020          FER=    20              ;FORMAT ERROR (BIT #4)
000040          WCF=    40              ;WRITE CLOCK FAIL (BIT #5)
000100          ECH=    100             ;ECC HARD ERROR (BIT #6)
000200          HCE=    200             ;HEADER COMPARE ERROR (BIT #7)
000400          HCRC=   400             ;HEADER CRC ERROR (BIT #8)
001000          AOE=    1000            ;ADDRESS OVERFLOW ERROR (BIT #9)
002000          IAE=    2000            ;INVALID ADDRESS ERROR (BIT #10)
004000          WLE=    4000            ;WRITE LOCK ERROR (BIT #11)
010000          DTE=    10000           ;DRIVE TIMING ERROR (BIT #12)
020000          OPI=    20000           ;OPERATION INCOMPLETE (BIT #13)
040000          UNS=    40000           ;DRIVE UNSAFE (BIT #14)
100000          DCK=    100000          ;DATA CHECK ERROR (BIT 15)

                ;MAINTAINABILITY REGISTER (RPMR)(#03)

000001          DMD=    1               ;DIAGNOSTIC MODE (BIT #0)
000002          MCLK=   2               ;MAINTAINABILITY CLOCK (BIT #1)
000004          MINX=   4               ;MAINTAINABILITY INDEX (BIT #2)
000010          MSTCK=  10              ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
000020          MRD=    20              ;MAINTAINABILITY READ (BIT #4)
000040          MWR=    40              ;MAINTAINABILITY WRITE (BIT #5)
000200          DTSY=   200             ;MAINTAINABILITY SYNC DETECTED (BIT #7)

                ;ATTENTION SUMMARY PSEUDO-REGISTER (RPAS) (#04)

000001          AT0=    1               ;DEVICE 0 (BIT #0)
000002          AT1=    2               ;DEVICE 1 (BIT #1)
000004          AT2=    4               ;DEVICE 2 (BIT #2)
000010          AT3=    10              ;DEVICE 3 (BIT #3)
000020          AT4=    20              ;DEVICE 4 (BIT #4)
000040          AT5=    40              ;DEVICE 5 (BIT #5)
000100          AT6=    100             ;DEVICE 6 (BIT #6)
000200          AT7=    200             ;DEVICE 7 (BIT #7)

                ;DESIRED SECTOR/TRACK ADDRESS REGISTER (RPDA) (#05)
                ;(EACH BIT IS CALLED BY BIT NUMBER)

                ;DRIVE TYPE REGISTER (RPDT) (#06)

000001          DT00=   1               ;DRIVE TYPE NUMBER BIT 1
000002          DT01=   2               ;DRIVE TYPE NUMBER BIT 2
000004          DT02=   4               ;DRIVE TYPE NUMBER BIT 3
000010          DT03=   10              ;DRIVE TYPE NUMBER BIT 4
000020          DT04=   20              ;DRIVE TYPE NUMBER BIT 5
000040          DT05=   40              ;DRIVE TYPE NUMBER BIT 6
000100          DT06=   100             ;DRIVE TYPE NUMBER BIT 7
000200          DT07=   200             ;DRIVE TYPE NUMBER BIT 8
000400          DT08=   400             ;DRIVE TYPE NUMBER BIT 9
004000          DRQ=    4000            ;DRIVE REQUEST REQUIRED (BIT #11)
020000          MOH=    20000           ;MOVING HEAD (BIT #13)
040000          TAP=    40000           ;TAPE DRIVE (BIT #14)
100000          NBA=    100000          ;NOT BLOCK ADDRESSED (BIT #15)

                ;LOOK-AHEAD REGISTER (RPLA) (#07)
```

```
                                ;RP04 ERROR REGISTER #2 (RPER2) (#10)

000001                  WCU=    1               ;WRITE CURRENT UNSAFE (BIT #0)
000002                  CSF=    2               ;CURRENT SINK FAILURE (BIT #1)
000004                  WSU=    4               ;WRITE SELECT UNSAFE (BIT #2)
000010                  CSU=    10              ;CURRENT SWITCH UNSAFE (BIT #3)
000020                  MSE=    20              ;MOTOR SEQUENCE ERROR (BIT #4)
000040                  TDF=    40              ;TRANSITIONS DETECTOR FAILURE (BIT #5)
000100                  TUF=    100             ;TRANSITIONS UNSAFE (BIT #6)
000200                  FEN=    200             ;FAILSAFE ENABLED (BIT #7)
000400                  WRU=    400             ;WRITE READY UNSAFE (BIT #8)
001000                  MHS=    1000            ;MULTIPLE HEAD SELECT (BIT #9)
002000                  NHS=    2000            ;NO HEAD SELECTION (BIT #10)
004000                  IXE=    4000            ;INDEX ERROR (BIT #11)
010000                  VU30=   10000           ;30VOLT UNSAFE (BIT #12)
020000                  PLU=    20000           ;PLO UNSAFE (BIT #13)
100000                  ACU=    100000          ;AC UNSAFE (BIT #15)

                                ;RP05/6 ERROR REGISTER #02 (RPER2) (#10)

000001                  WCU=    1               ;WRITE CURRENT UNSAFE (BIT #0)
000002                  CSF=    2               ;CURRENT SINK FAILURE (BIT #1)
000004                  WSU=    4               ;WRITE SELECT UNSAFE (BIT #2)
000010                  CSU=    10              ;CURRENT SWITCH UNSAFE (BIT #3)
000020                  RAW=    20              ;READ AND WRITE (BIT #4)
000040                  TDF=    40              ;TRANSITIONS DETECTOR FAILURE (BIT #5)
000100                  TUF=    100             ;TRANSITIONS UNSAFE (BIT #6)
000200                  ABS=    200             ;ABNORMAL STOP (BIT #7)
000400                  WRU=    400             ;WRITE READY UNSAFE (BIT #8)
001000                  MHS=    1000            ;MUTLTIPLE HEAD SELECT (BIT #9)
002000                  NHS=    2000            ;NO HEAD SELECTION (BIT #10)
004000                  IXE=    4000            ;INDEX ERROR (BIT #11)
020000                  PLU=    20000           ;PLO UNSAFE (BIT #12)

                                ;OFFSET REGISTER (RPOF) (#11)

000001                  OF25=   1               ;OFFSET 25 MICRO INCHES (BIT #0)
000002                  OF50=   2               ;OFFSET 50 MICRO INCHES (BIT #1)
```

```
000004          OF100=  4               ;OFFSET 100 MICRO INCHES (BIT #2)
000010          OF200=  10              ;OFFSET 200 MICRO INCHES (BIT #3)
000020          OF400=  20              ;OFFSET 400 MICRO INCHES (BIT #4)
000040          OF800=  40              ;OFFSET 800 MICRO INCHES (BIT #5)
000200          OFREV=  200             ;OFFSET NEGATIVE (REVERSE) (BIT #5)
002000          HCI=    2000            ;HEADER COMPARE INHIBIT (BIT #10)
004000          ECI=    4000            ;ERROR CORRECTION CODE INHIBIT (BIT #11)
010000          FMT22=  10000           ;FORMAT BIT (BIT #12

                ;DESIRED CYLINDER ADDRESS (RPCA) (#12)
                ;(EACH BIT IS CALLED BY BIT NUMBER)

                ;CURRENT CYLINDER ADDRESS (RPCC) (#13)
                ;(EACH BIT IS CALLED BY BIT NUMBER)

                ;SERIAL NUMBER REGISTER (RPSN) (#14)
                ;(EACH IS CALLED BY BIT NUMBER)

                ;RP04 ERROR REGISTER #03 (RPER3) (#15)

000001          PSU=    1               ;PACK SPEED UNSAFE (BIT #0)
000002          VUF=    2               ;VELOCITY UNSAFE (BIT #1)
000010          UWR=    10              ;ANY UNSAFE EXCEPT READ/WRITE (BIT #3)
000040          ACL=    40              ;AC LOW (BIT #5)
000100          DCL=    100             ;DC LOW (BIT #6)
040000          SKI=    40000           ;SEEK INCOMPLETE (BIT #14)
100000          OCYL=   100000          ;OFF CYLINDER (BIT #15)

                ;RP05/6 ERROR REGISTER #03 (RPER3) (#15)

000001          DCU=    1               ;DC UNSAFE (BIT #0)
000002          WAC=    2               ;WRITE AND OFFSET (BIT #1)
000040          ACL=    40              ;AC LOW (BIT #5)
000100          DCL=    100             ;DC LOW (BIT #6)
020000          OPE=    20000           ;OPERATOR PLUG ERROR (BIT #13)
040000          SKI=    40000           ;SEEK INCOMPLETE (BIT #14)
100000          OCYL=   100000          ;OFF CYLINDER ERROR (BIT #15)

                ;ECC POSITION REGISTER (RPEC1) (#16)
                ;(EACH BIT IS CALLED BY BIT NUMBER)

                ;ECC PATTERN REGISTER (RPEC2) (#17)
                ;(EACH BIT IS CALLED BY BIT NUMBER)

                ;;*****************************************************************

                ;OP CODE DEFINITIONS
000101          NOOP=101
000103          UNLOAD=103
000105          SEEK=105
000107          RECAL=107
000111          DRVCLR=111
000113          RELEASE=113
000115          OFFSET=115
000117          RTC=117
000121          READIN=121
```

```
                000123          PACK=123
                000131          SEARCH=131
                000151          WRCKD=151
                000153          WRCKHD=153
                000161          WRITE=161
                000163          WRTHD=163
                000171          READ=171
                000173          READHD=173
                000141          GETREG=141
                000143          SETFORM=143
                000145          SELDRV=145

                                ;OTHER EQUATES

                177400          SCTRWC = -256.          ;WORD COUNT FOR SECTOR
                010000          FMT22=10000             ;FORMAT 22 BIT

```

```
                                .SBTTL  COMMON TAGS
                                ;;*****************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102     000          $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103     000          $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114     000          $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115     001          $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134     000          $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135     000          $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000          $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002          $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012          $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157     000          $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REGO) WAS OBTAINED
001162  000000          $REGO:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000          $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  000000          $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001170  000000          $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001172  000000          $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001174  000000          $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001176  000000          $TMP0:  .WORD   0               ;;USER DEFINED
001200  000000          $TMP1:  .WORD   0               ;;USER DEFINED
001202  000000          $TMP2:  .WORD   0               ;;USER DEFINED
001204  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001206  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001210  177607  000377   $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001214     077          $QUES:  .ASCII  /?/             ;;QUESTION MARK
001215     015          $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001216  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
                        000015  CR      =       15
```

COMMON TAGS

```
                000012                  LF      =       12
001220  000000                  C.SWR:  .WORD   0               ;CONTROL SWITCHES
001222  000000                  SAVCSW: .WORD   0               ;PREVIOUS CONTENTS OF 'C.SWR'
001224  000000                  CNTRLC: .WORD   0               ;CONTROL "C" FLAG
001226  000000                  BUSADR: .WORD   0               ;GET ADDRESSES FROM THE TTY FLAG (0=NO, -1=YES)
001230  000000                  LPTAVL: .WORD   0               ;LPT AVAILABLE STATUS (0=NO,1=YES)
001232  000000                  DRVSEL: .WORD   0               ;DRIVES SELECTED FOR TESTING
001234  037777  000000          TSTNMS: .WORD   37777,0         ;RUN TESTS 0-15
001240  000000  000000          OPNFLG: .WORD   0,0             ;MODIFY TEST PARAMETER FLAGS
001244  000000                  CLKSTA: .WORD   0               ;CLOCK STATUS (0=NO CLOCK,+1=KW11-P,
                                                                ;AND -1=KW11-L)
001246  000020                  TICKMS: .WORD   16.             ;16 MILLISECONDS PER CLOCK TICK
001250  040432                  TICKUS: .WORD   16666.          ;16666 MIRCOSECONDS PER CLOCK TICK
001252  000000                  BYPASS: .WORD   0
001254  000000                  CHKDRV: .WORD   0               ;DRIVE UNDER TEST
001256  000000                  DRVMSK: .WORD   0               ;DRIVE MASK BIT
001260  000000                  SVSTAT: .WORD   0               ;STATUS/ERROR INDICATOR IS
                                                                ;SAVED HERE ON AN ERROR
001262  000000                  CYL.RD: .WORD   0               ;CYLINDER READ
001264  000000                  TRK.RD: .WORD   0               ;TRACK READ
001266  000000                  SEC.RD: .WORD   0               ;SECTOR READ
001270  000000                  CYL.DS: .WORD   0               ;CYLINDER DESIRED
001272  000000                  SEC.DS: .WORD   0               ;SECTOR DESIRED
001274  000000                  TRK.DS: .WORD   0               ;TRACK DESIRED
001276  000000                  TIM.UP: .WORD   0               ;MINIMUM TIME
001300  000000                          .WORD   0               ;NUMBER OF COUNTS BELOW MIN. LIMIT
001302  000000                          .WORD   0               ;MAXIMUM TIME
001304  000000                          .WORD   0               ;NUMBER OF COUNTS ABOVE MAX. LIMIT
001306  000000  000000                  .WORD   0,0             ;TOTAL TIME OF ALL SEEKS
001312  000000                          .WORD   0               ;NUMBER OF SEEKS PERFORMED
001314  000000                  TIM.DN: .WORD   0               ;MINIMUM TIME
001316  000000                          .WORD   0               ;NUMBER OF COUNTS BELOW MIN. LIMIT
001320  000000                          .WORD   0               ;MAXIMUM TIME
001322  000000                          .WORD   0               ;NUMBER OF COUNTS ABOVE MAX. LIMIT
001324  000000  000000                  .WORD   0,0             ;TOTAL TIME OF ALL SEEKS
001330  000000                          .WORD   0               ;NUMBER OF SEEKS PERFORMED
001332  000000                  TIM.PT: .WORD   0               ;POINTS TO TABLE OF TIMES
001334  000000                  WCEFLG: .WORD   0               ;FATAL WRITE CHECK ERROR FLAG (TEST 20)
001336  000000                  STALLO: .WORD   0               ;VARIABLE STALL (TEST 4)
001340  000000  000000          SVADR:  .WORD   0,0             ;SAVE DISK ADDRESS (TEST 22)
001344  000000                  SEKTMR: .WORD   0               ;SEEK TIMER  (TEST 10)
001346  000000                  SEKCNT: .WORD   0               ;SEEK COUNTER
001350  000000                  DELTA:  .WORD   0               ;TESTING RANGE FOR SERVO SETTLE DOWN TEST
001352  165000                  TRCKWC: .WORD   -(256.*22.)     ;WORD COUNT FOR A FULL TRACK IN 16 BIT MODE
001354  000012                  STALL1: .WORD   10.             ;10 MILLISECONDS STALL (TEST 0-11)
001356  000012                  STALL2: .WORD   10.             ;10 MILLISECONDS STALL (TEST 16-21)
001360  011610                  STALL3: .WORD   5000.           ;5 SEC STALL (TEST 22)
001362  000031                  MXSTAL: .WORD   25.             ;MAX. INCREMENTING STALL ALLOWED IN TEST 4
001364      144                 ERR.CT: .BYTE   100.            ;NUMBER OF ERRORS ALLOWED IN TESTS 16 - 21
                                                                ;BEFORE GOING TO THE NEXT TEST
001365      000                         .BYTE   0               ;RESERVED

                                ;ADDRESSES AND VECTORS
001366  176700                  RH.ADR: .WORD   176700          ;RH11-RH70 UNIBUS ADDRESS
001370  000254  000240          RHVEC:  .WORD   254,5*32.       ;RH11-RH70 VECTOR ADDRESS AND PRIORITY
001374  000104  000106          PKV:    .WORD   104,106         ;KW11-P VECTOR ADDRESS
```

```
                                PKCS:   .WORD   172540          ;KW11-P CONTROL AND STATUS REG.
                                PKB:    .WORD   172542          ;KW11-P COUNT SET BUFFER
                                PKC:    .WORD   172544          ;KW11-P COUNTER
                                LKV:    .WORD   100,102         ;KW11-L VECTOR ADDRESS
                                LKS:    .WORD   177546          ;KW11-L STATUS REGISTER
                                TPS:    .WORD   177564          ;TTY PRINTER STATUS
                                TPB:    .WORD   177566          ;TTY PRINTER BUFFER
                                LPS:    .WORD   177514          ;LINE PRINTER STATUS
                                LPB:    .WORD   177516          ;LINE PRINTER BUFFER

                                ;BIT TABLE
                                BITS:   .WORD   BIT00
                                        .WORD   BIT01
                                        .WORD   BIT02
                                        .WORD   BIT03
                                        .WORD   BIT04
                                        .WORD   BIT05
                                        .WORD   BIT06
                                        .WORD   BIT07
                                        .WORD   BIT08
                                        .WORD   BIT09
                                        .WORD   BIT10
                                        .WORD   BIT11
                                        .WORD   BIT12
                                        .WORD   BIT13
                                        .WORD   BIT14
                                        .WORD   BIT15
                                        .WORD   BIT00
                                        .WORD   BIT01
                                        .WORD   BIT02
                                        .WORD   BIT03
                                        .WORD   BIT04
                                        .WORD   BIT05
                                        .WORD   BIT06
                                        .WORD   BIT07

                                ;COMMON STORAGE FOR TEST PARAMETER
                                PRM:    .WORD   0
                                RPT:    .WORD   0               ;REPEAT COUNTS FOR ALL TESTS
                                FC:     .WORD   0               ;FIRST CYLINDER
                                LC:     .WORD   0               ;LAST CYLINDER
                                IC:     .WORD   0               ;INCREMENT CYLINDER
                                FT:     .WORD   0               ;FIRST TRACK
                                LT:     .WORD   0               ;LAST TRACK
                                IT:     .WORD   0               ;INCREMENT TRACK
                                FS:     .WORD   0               ;FIRST SECTOR
                                LS:     .WORD   0               ;LAST SECTOR
                                PAT:    .WORD   0               ;PATTERN CODE
                                NC1:    .WORD   0               ;NEW CYLINDER ADDRESS
                                NC2:    .WORD   0               ;NEW CYLINDER ADDRESS

                                ;TABLE OF PARAMETER POINTERS
                                PRMPT:  .WORD   PRM0
                                        .WORD   PRM1
                                        .WORD   PRM2
```

```
636  001544  002414                          .WORD   PRM3
637  001546  002436                          .WORD   PRM4
638  001550  002460                          .WORD   PRM5
639  001552  002502                          .WORD   PRM6
640  001554  002524                          .WORD   PRM7
641  001556  002544                          .WORD   PRM10
642  001560  002564                          .WORD   PRM11
643  001562  002606                          .WORD   PRM12
644  001564  002622                          .WORD   PRM13
645  001566  002636                          .WORD   PRM14
646  001570  002652                          .WORD   PRM15
647  001572  002566                          .WORD   PRM16
648  001574  002700                          .WORD   PRM17
649  001576  002712                          .WORD   PRM20
650  001600  002744                          .WORD   PRM21
651  001602  002760                          .WORD   PRM22
652  001604  000000                          .WORD   0                       ;TERMINATOR
653
654                                  ;PARAMETER UPPER LIMIT
655  001606  032767                  PRMLMT: .WORD   32767                   ;"R"
656  001610  000632                          .WORD   410.                    ;"FC"
657  001612  000632                          .WORD   410.                    ;"LC"
658  001614  001456                          .WORD   814.                    ;"FC'"
659  001616  001456                          .WORD   814.                    ;"LC'"
660  001620  001456                          .WORD   814.                    ;"IC"
661  001622  000022                          .WORD   18.                     ;"FT"
662  001624  000022                          .WORD   18.                     ;"LT"
663  001626  000022                          .WORD   18.                     ;"IT"
664  001630  000025                          .WORD   21.                     ;"FS"
665  001632  000025                          .WORD   21.                     ;"LS"
666  001634  177777                          .WORD   177777                  ;"PAT"
667
668                                  ;TABLE OF MESSAGE POINTERS
669  001636  042750                  PRMMSG: .WORD   MSG.R
670  001640  042752                          .WORD   MSG.FC
671  001642  042755                          .WORD   MSG.LC
672  001644  042760                          .WORD   MSG.FCP
673  001646  042764                          .WORD   MSG.LCP
674  001650  042770                          .WORD   MSG.IC
675  001652  042773                          .WORD   MSG.FT
676  001654  042776                          .WORD   MSG.LT
677  001656  043001                          .WORD   MSG.IT
678  001660  043004                          .WORD   MSG.FS
679  001662  043007                          .WORD   MSG.LS
680
681                                  ;STATUS/ERROR INDICATOR MESSAGES POINTER TABLE
682                                  ;DEFAULT VALUES OF TEST PARAMETERS
683  001664  001125  000310  000632  DFLT:   .WORD   1125,200.,410.,814.,0,0 ;RECAL/SEEK (T0)
684  001672  001456  000000  000000
685  001700  003377  000144  000000          .WORD   3377,100.,0,128.,0,256.,0,0,0,0,0 ;SEEK/SEEK (T1)
686  001706  000200  000000  000400
687  001714  000000  000000  000000
688  001722  000000  000000
689  001726  001177  000001  000000          .WORD   1177,1,0,410.,0,814.,1,0,0 ;INCREMENT SEEK (T2)
690  001734  000632  000000  001456
691  001742  000001  000000  000000
```

```
692  001750  001177  000010  000000          .WORD   1177,10,0,256.,0,256.,1,0,0 ;STEPPING SEEK (T3)
693  001756  000400  000000  000400
694  001764  000001  000000  000000
695  001772  001177  000001  000000          .WORD   1177,1,0,410.,0,814.,1,0,0 ;OSCILLATING SEEK (T4)
696  002000  000632  000000  001456
697  002006  000001  000000  000000
698  002014  001177  000001  000000          .WORD   1177,1,0,410.,0,814.,1,0,0 ;CONVERGING/DIVERGING SEEK (T5)
699  002022  000632  000000  001456
700  002030  000001  000000  000000
701  002036  001177  000001  000000          .WORD   1177,1,0,410.,0,814.,1,0,0 ;SERVO ADDRESSING LOGIC NOISE (T6)
702  002044  000632  000000  001456
703  002052  000001  000000  000000
704  002060  000337  011610  000000          .WORD   337,5000.,0,410.,0,814.,0,18.  ;RANDOM SEEK TEST (T7)
705  002066  000632  000000  001456
706  002074  000000  000022
707  002100  000177  000001  000000          .WORD   177,1,0,410.,0,814.,100.,0  ;SERVO SETTLE DOWN TEST (T10)
708  002106  000632  000000  001456
709  002114  000144  000000
710  002120  001177  000001  000000          .WORD   1177,1,0,410.,0,814.,1,0,0      ;ALL SEEKS TEST (T11)
711  002126  000632  000000  001456
712  002134  000001  000000  000000
713  002142  001113  000001  000000          .WORD   1113,1,0,0,0,0  ;ROTATIONAL SPEED TIMING TEST (T12)
714  002150  000000  000000  000000
715  002156  000037  000001  000000          .WORD   37,1,0,410.,0,814.      ;ONE CYLINDER SEEK TIMING TEST (T13)
716  002164  000632  000000  001456
717  002172  000037  000001  000000          .WORD   37,1,0,136.,0,255.      ;ACCESS TIME MEASURMENT TEST (T14)
718  002200  000210  000000  000377
719  002206  000037  000001  000000          .WORD   37,1,0,410.,0,814.      ;MAXIMUM SEEK TIMING TEST (T15)
720  002214  000632  000000  001456
721  002222  000113  000001  000000          .WORD   113,1,0,0,0     ;SECTOR ADDRESSING TEST (T16)
722  002230  000000  000000
723  002234  001013  000001  000000          .WORD   1013,1,0,0,0    ;TRACK ADDRESSING TEST (T17)
724  002242  000000  000000
725  002246  007777  000001  000000          .WORD   7777,1,0,410.,0,814.,64.,0,18.,1,1,0,177777 ;DATA TEST (T20)
726  002254  000632  000000  001456
727  002262  000100  000000  000022
728  002270  000001  000001  000000
729  002276  177777
730  002300  000037  047040  000000          .WORD   37,20000.,0,410.,0,814. ;EXERCISER (T21)
731  002306  000632  000000  001456
732  002314  000037  011610  000000          .WORD   37,5000.,0,136.,0,255.  ;RP04 ACCESS TIME ADJUSTMENT TEST (T22)
733  002322  000210  000000  000377
734
735                                     ;PARAMETER TABLES
736
737                                     ;RECAL/SEEK (T0)
738  002330  001125                     PRM0:   .WORD   1125
739  002332  000C.0                             .WORD   200.
740  002334  000632                             .WORD   410.
741  002336  001456                             .WORD   814.
742  002340  000000                             .WORD   0
743  002342  000000                             .WORD   0
744
745                                     ;SEEK/SEEK (T1)
746  002344  003377                     PRM1:   .WORD   3377
747  002346  000144                             .WORD   100.
```

```
   748  002350  000000                          .WORD   0
   749  002352  000200                          .WORD   128.
   750  002354  000000                          .WORD   0
   751  002356  000400                          .WORD   256.
   752  002360  000000                          .WORD   0
   753  002362  000000                          .WORD   0
   754  002364  000000                          .WORD   0
   755  002366  000000                          .WORD   0
   756  002370  000000                          .WORD   0
   757
   758                                  ;INCREMENT SEEK (T2)
   759  002372  001177                  PRM2:   .WORD   1177
   760  002374  000001                          .WORD   1
   761  002376  000000                          .WORD   0
   762  002400  000632                          .WORD   410.
   763  002402  000000                          .WORD   0
   764  002404  001456                          .WORD   814.
   765  002406  000001                          .WORD   1
   766  002410  000000                          .WORD   0
   767  002412  000000                          .WORD   0
   768
   769                                  ;STEPPING SEEK (T3)
   770  002414  001177                  PRM3:   .WORD   1177
   771  002416  000001                          .WORD   1
   772  002420  000000                          .WORD   0
   773  002422  000400                          .WORD   256.
   774  002424  000000                          .WORD   0
   775  002426  001000                          .WORD   512.
   776  002430  000001                          .WORD   1
   777  002432  000000                          .WORD   0
   778  002434  000000                          .WORD   0
   779
   780                                  ;OSCILLATING SEEK (T4)
   781  002436  001177                  PRM4:   .WORD   1177
   782  002440  000001                          .WORD   1
   783  002442  000000                          .WORD   0
   784  002444  000632                          .WORD   410.
   785  002446  000000                          .WORD   0
   786  002450  001456                          .WORD   814.
   787  002452  000001                          .WORD   1
   788  002454  000000                          .WORD   0
   789  002456  000000                          .WORD   0
   790
   791                                  ;CONVERGING/DIVERGING SEEK (T5)
   792  002460  001177                  PRM5:   .WORD   1177
   793  002462  000001                          .WORD   1
   794  002464  000000                          .WORD   0
   795  002466  000632                          .WORD   410.
   796  002470  000000                          .WORD   0
   797  002472  001456                          .WORD   814.
   798  002474  000001                          .WORD   1
   799  002476  000000                          .WORD   0
   800  002500  000000                          .WORD   0
   801
   802                                  ;SERVO ADDRESSING LOGIC NOISE GENERATOR (T6)
   803  002502  001177                  PRM6:   .WORD   1177
```

COMMON TAGS

```
                                        .WORD   1
                                        .WORD   0
                                        .WORD   410.
                                        .WORD   0
                                        .WORD   814.
                                        .WORD   1
                                        .WORD   0
                                        .WORD   0

                                ;RANDOM SEEK TEST (T7)
                                PRM7:   .WORD   337
                                        .WORD   5000.
                                        .WORD   0
                                        .WORD   410.
                                        .WORD   0
                                        .WORD   814.
                                        .WORD   0
                                        .WORD   18.

                                ;SERVO SETTLE DOWN TEST (T10)
                                PRM10:  .WORD   177
                                        .WORD   1
                                        .WORD   0
                                        .WORD   410.
                                        .WORD   0
                                        .WORD   814.
                                        .WORD   100.
                                        .WORD   0

                                ;ALL SEEKS TEST (T11)
                                PRM11:  .WORD   1177
                                        .WORD   1
                                        .WORD   0
                                        .WORD   410.
                                        .WORD   0
                                        .WORD   814.
                                        .WORD   1
                                        .WORD   0
                                        .WORD   0

                                ;ROTATIONAL SPEED TIMING TEST (T12)
                                PRM12:  .WORD   1113
                                        .WORD   1
                                        .WORD   0
                                        .WORD   0
                                        .WORD   0
                                        .WORD   0

                                ;ONE CYLINDER SEEK TIMING TEST (T13)
                                PRM13:  .WORD   37
                                        .WORD   1
                                        .WORD   0
                                        .WORD   410.
                                        .WORD   0
                                        .WORD   814.
```

```
860
861  002636  000037          ;ACCESS TIME MEASURMENT TEST (T14)
862  002640  000001  PRM14:   .WORD   37
863  002642  000000          .WORD   1
864  002644  000210          .WORD   0
865  002646  000000          .WORD   136.
866  002650  000377          .WORD   0
867                          .WORD   255.
868
869  002652  000037          ;MAXIMUM SEEK TIMING TEST (T15)
870  002654  000001  PRM15:   .WORD   37
871  002656  000000          .WORD   1
872  002660  000632          .WORD   0
873  002662  000000          .WORD   410.
874  002664  001456          .WORD   0
875                          .WORD   814.
876
877  002666  000113          ;SECTOR ADDRESSING TEST (T16)
878  002670  000001  PRM16:   .WORD   113
879  002672  000000          .WORD   1
880  002674  000000          .WORD   0
881  002676  000000          .WORD   0
882                          .WORD   0
883
884  002700  001013          ;TRACK ADDRESSING TEST (T17)
885  002702  000001  PRM17:   .WORD   1013
886  002704  000000          .WORD   1
887  002706  000000          .WORD   0
888  002710  000000          .WORD   0
889                          .WORD   0
890
891  002712  007777          ;DATA TEST (T20)
892  002714  000001  PRM20:   .WORD   7777
893  002716  000000          .WORD   1
894  002720  000632          .WORD   0
895  002722  000000          .WORD   410.
896  002724  001456          .WORD   0
897  002726  000100          .WORD   814.
898  002730  000000          .WORD   64.
899  002732  000022          .WORD   0
900  002734  000001          .WORD   18.
901  002736  000001          .WORD   1
902  002740  000000          .WORD   1
903  002742  177777          .WORD   0
904                  PTRN15: .WORD   177777
905
906  002744  000037          ;EXERCISER (T21)
907  002746  047040  PRM21:   .WORD   37
908  002750  000000          .WORD   20000.
909  002752  000632          .WORD   0
910  002754  000000          .WORD   410.
911  002756  001456          .WORD   0
912                          .WORD   814.
913
914  002760  000037          ;RP04 ACCESS TIME ADJUSTMENT TEST (T22)
915  002762  011610  PRM22:   .WORD   37
                             .WORD   5000.
```

```
916  002764  000000                          .WORD   0
917  002766  000210                          .WORD   136.
918  002770  000000                          .WORD   0
919  002772  000377                          .WORD   255.
920
921
922
923                                  ;SEEK TIMING LIMITS
924
925  002774  043370                  T7A:    .WORD   MSG7
926  002776  000000                          .WORD   0
927  003000  003142                          .WORD   1634.           ;(16.67-2%)**2
928  003002  003244                          .WORD   1700.           ;(16.67+2%)**2
929
930  003004  043370                  T7B:    .WORD   MSG7
931  003006  000000                          .WORD   0
932  003010  003131                          .WORD   1625.           ;(16.67-2.5%)**2
933  003012  003255                          .WORD   1709.           ;(16.67+2.5%)**2
934
935  003014  043422                  T10:    .WORD   MSG10A
936  003016  043471                          .WORD   MSG10B
937  003020  000000                          .WORD   0               ;NO LOWER LIMIT
938  003022  001750                          .WORD   1000.           ;(7+3)**2
939
940  003024  043506                  T11:    .WORD   MSG11A
941  003026  043554                          .WORD   MSG11B
942  003030  000000                          .WORD   0               ;NO LOWER LIMIT
943  003032  005670                          .WORD   3000.           ;(28+2)**2
944
945  003034  043571                  T12:    .WORD   MSG12A
946  003036  043633                          .WORD   MSG12B
947  003040  000000                          .WORD   0               ;NO LOWER LIMIT
948  003042  012120                          .WORD   5200.           ;(50+2)**2
949
950  003044  003104                  PAT.PT: .WORD   PAT0             ;TABLE OF POINTERS WHICH POINT TO THE
951  003046  003144                          .WORD   PAT1            ;PATTERNS USED BY THE DATA TEST
952  003050  003204                          .WORD   PAT2
953  003052  003244                          .WORD   PAT3
954  003054  003304                          .WORD   PAT4
955  003056  003344                          .WORD   PAT5
956  003060  003404                          .WORD   PAT6
957  003062  003444                          .WORD   PAT7
958  003064  003504                          .WORD   PAT8
959  003066  003544                          .WORD   PAT9
960  003070  003604                          .WORD   PAT10
961  003072  003644                          .WORD   PAT11
962  003074  003704                          .WORD   PAT12
963  003076  003744                          .WORD   PAT13
964  003100  004004                          .WORD   PAT14
965  003102  004044                          .WORD   PAT15
966
967                                  ;PATTERNS 0 THRU 15
968
969  003104  165555                  PAT0:   .WORD   165555           ;PATTERN 0
970  003106  133333                          .WORD   133333
971  003110  165555                          .WORD   165555
```

```
  972  003112  133333                        .WORD   133333
  973  003114  165555                        .WORD   165555
  974  003116  133333                        .WORD   133333
  975  003120  165555                        .WORD   165555
  976  003122  133333                        .WORD   133333
  977  003124  165555                        .WORD   165555
  978  003126  133333                        .WORD   133333
  979  003130  165555                        .WORD   165555
  980  003132  133333                        .WORD   133333
  981  003134  165555                        .WORD   165555
  982  003136  133333                        .WORD   133333
  983  003140  165555                        .WORD   165555
  984  003142  133333                        .WORD   133333
  985
  986  003144  000001              PAT1:     .WORD   000001          ;PATTERN 1
  987  003146  000003                        .WORD   000003
  988  003150  000007                        .WORD   000007
  989  003152  000017                        .WORD   000017
  990  003154  000037                        .WORD   000037
  991  003156  000077                        .WORD   000077
  992  003160  000177                        .WORD   000177
  993  003162  000377                        .WORD   000377
  994  003164  000777                        .WORD   000777
  995  003166  001777                        .WORD   001777
  996  003170  003777                        .WORD   003777
  997  003172  007777                        .WORD   007777
  998  003174  017777                        .WORD   017777
  999  003176  037777                        .WORD   037777
 1000  003200  077777                        .WORD   077777
 1001  003202  177777                        .WORD   177777
 1002
 1003  003204  177776              PAT2:     .WORD   177776          ;PATTERN 2
 1004  003206  177774                        .WORD   177774
 1005  003210  177770                        .WORD   177770
 1006  003212  177760                        .WORD   177760
 1007  003214  177740                        .WORD   177740
 1008  003216  177700                        .WORD   177700
 1009  003220  177600                        .WORD   177600
 1010  003222  177400                        .WORD   177400
 1011  003224  177000                        .WORD   177000
 1012  003226  176000                        .WORD   176000
 1013  003230  174000                        .WORD   174000
 1014  003232  170000                        .WORD   170000
 1015  003234  160000                        .WORD   160000
 1016  003236  140000                        .WORD   140000
 1017  003240  100000                        .WORD   100000
 1018  003242  000000                        .WORD   000000
 1019
 1020  003244  000000              PAT3:     .WORD   000000          ;PATTERN 3
 1021  003246  000000                        .WORD   000000
 1022  003250  000000                        .WORD   000000
 1023  003252  177777                        .WORD   177777
 1024  003254  177777                        .WORD   177777
 1025  003256  177777                        .WORD   177777
 1026  003260  000000                        .WORD   000000
 1027  003262  000000                        .WORD   000000
```

```
1028  003264  177777                          .WORD   177777
1029  003266  177777                          .WORD   177777
1030  003270  000000                          .WORD   000000
1031  003272  177777                          .WORD   177777
1032  003274  000000                          .WORD   000000
1033  003276  177777                          .WORD   177777
1034  003300  000000                          .WORD   000000
1035  003302  177777                          .WORD   177777
1036
1037  003304  000000                  PAT4:   .WORD   000000          ;PATTERN 4
1038  003306  010421                          .WORD   010421
1039  003310  021042                          .WORD   021042
1040  003312  031463                          .WORD   031463
1041  003314  042104                          .WORD   042104
1042  003316  052525                          .WORD   052525
1043  003320  063146                          .WORD   063146
1044  003322  073567                          .WORD   073567
1045  003324  104210                          .WORD   104210
1046  003326  114631                          .WORD   114631
1047  003330  125252                          .WORD   125252
1048  003332  135673                          .WORD   135673
1049  003334  146314                          .WORD   146314
1050  003336  156735                          .WORD   156735
1051  003340  167356                          .WORD   167356
1052  003342  177777                          .WORD   177777
1053
1054  003344  052525                  PAT5:   .WORD   052525          ;PATTERN 5
1055  003346  052525                          .WORD   052525
1056  003350  052525                          .WORD   052525
1057  003352  125252                          .WORD   125252
1058  003354  125252                          .WORD   125252
1059  003356  125252                          .WORD   125252
1060  003360  052525                          .WORD   052525
1061  003362  052525                          .WORD   052525
1062  003364  125252                          .WORD   125252
1063  003366  125252                          .WORD   125252
1064  003370  052525                          .WORD   052525
1065  003372  125252                          .WORD   125252
1066  003374  052525                          .WORD   052525
1067  003376  125252                          .WORD   125252
1068  003400  052525                          .WORD   052525
1069  003402  125252                          .WORD   125252
1070
1071  003404  007417                  PAT6:   .WORD   007417          ;PATTERN 6
1072  003406  007417                          .WORD   007417
1073  003410  007417                          .WORD   007417
1074  003412  170360                          .WORD   170360
1075  003414  170360                          .WORD   170360
1076  003416  170360                          .WORD   170360
1077  003420  007417                          .WORD   007417
1078  003422  007417                          .WORD   007417
1079  003424  170360                          .WORD   170360
1080  003426  170360                          .WORD   170360
1081  003430  007417                          .WORD   007417
1082  003432  170360                          .WORD   170360
1083  003434  007417                          .WORD   007417
```

```
1084  003436  170360                          .WORD   170360
1085  003440  007417                          .WORD   007417
1086  003442  170360                          .WORD   170360
1087
1088  003444  026455          PAT7:           .WORD   026455          ;PATTERN 7
1089  003446  026455                          .WORD   026455
1090  003450  026455                          .WORD   026455
1091  003452  151322                          .WORD   151322
1092  003454  151322                          .WORD   151322
1093  003456  151322                          .WORD   151322
1094  003460  026455                          .WORD   026455
1095  003462  026455                          .WORD   026455
1096  003464  151322                          .WORD   151322
1097  003466  151322                          .WORD   151322
1098  003470  026455                          .WORD   026455
1099  003472  151322                          .WORD   151322
1100  003474  026455                          .WORD   026455
1101  003476  151322                          .WORD   151322
1102  003500  026455                          .WORD   026455
1103  003502  151322                          .WORD   151322
1104
1105  003504  165555          PAT8:           .WORD   165555          ;PATTERN 8
1106  003506  133333                          .WORD   133333
1107  003510  165555                          .WORD   165555
1108  003512  133333                          .WORD   133333
1109  003514  165555                          .WORD   165555
1110  003516  133333                          .WORD   133333
1111  003520  165555                          .WORD   165555
1112  003522  133333                          .WORD   133333
1113  003524  165555                          .WORD   165555
1114  003526  133333                          .WORD   133333
1115  003530  165555                          .WORD   165555
1116  003532  133333                          .WORD   133333
1117  003534  165555                          .WORD   165555
1118  003536  133333                          .WORD   133333
1119  003540  165555                          .WORD   165555
1120  003542  133333                          .WORD   133333
1121
1122  003544  000001          PAT9:           .WORD   000001          ;PATTERN 9
1123  003546  000002                          .WORD   000002
1124  003550  000004                          .WORD   000004
1125  003552  000010                          .WORD   000010
1126  003554  000020                          .WORD   000020
1127  003556  000040                          .WORD   000040
1128  003560  000100                          .WORD   000100
1129  003562  000200                          .WORD   000200
1130  003564  000400                          .WORD   000400
1131  003566  001000                          .WORD   001000
1132  003570  002000                          .WORD   002000
1133  003572  004000                          .WORD   004000
1134  003574  010000                          .WORD   010000
1135  003576  020000                          .WORD   020000
1136  003600  040000                          .WORD   040000
1137  003602  100000                          .WORD   100000
1138
1139  003604  177776          PAT10:          .WORD   177776          ;PATTERN 10
```

```
1140  003606  177775                          .WORD   177775
1141  003610  177773                          .WORD   177773
1142  003612  177767                          .WORD   177767
1143  003614  177757                          .WORD   177757
1144  003616  177737                          .WORD   177737
1145  003620  177677                          .WORD   177677
1146  003622  177577                          .WORD   177577
1147  003624  177377                          .WORD   177377
1148  003626  176777                          .WORD   176777
1149  003630  175777                          .WORD   175777
1150  003632  173777                          .WORD   173777
1151  003634  167777                          .WORD   167777
1152  003636  157777                          .WORD   157777
1153  003640  137777                          .WORD   137777
1154  003642  077777                          .WORD   077777
1155
1156  003644  172666                PAT11:    .WORD   172666          ;PATTERN 11
1157  003646  155555                          .WORD   155555
1158  003650  172666                          .WORD   172666
1159  003652  155555                          .WORD   155555
1160  003654  172666                          .WORD   172666
1161  003656  155555                          .WORD   155555
1162  003660  172666                          .WORD   172666
1163  003662  155555                          .WORD   155555
1164  003664  172666                          .WORD   172666
1165  003666  155555                          .WORD   155555
1166  003670  172666                          .WORD   172666
1167  003672  155555                          .WORD   155555
1168  003674  172666                          .WORD   172666
1169  003676  155555                          .WORD   155555
1170  003700  172666                          .WORD   172666
1171  003702  155555                          .WORD   155555
1172
1173  003704  077777                PAT12:    .WORD   077777          ;PATTERN 12
1174  003706  137777                          .WORD   137777
1175  003710  157777                          .WORD   157777
1176  003712  167777                          .WORD   167777
1177  003714  173777                          .WORD   173777
1178  003716  175777                          .WORD   175777
1179  003720  176777                          .WORD   176777
1180  003722  177377                          .WORD   177377
1181  003724  177577                          .WORD   177577
1182  003726  177677                          .WORD   177677
1183  003730  177737                          .WORD   177737
1184  003732  177757                          .WORD   177757
1185  003734  177767                          .WORD   177767
1186  003736  177773                          .WORD   177773
1187  003740  177775                          .WORD   177775
1188  003742  177776                          .WORD   177776
1189
1190  003744  153333                PAT13:    .WORD   153333          ;PATTERN 13
1191  003746  066667                          .WORD   066667
1192  003750  153333                          .WORD   153333
1193  003752  066667                          .WORD   066667
1194  003754  153333                          .WORD   153333
1195  003756  066667                          .WORD   066667
```

```
1196  003760  153333                          .WORD   153333
1197  003762  066667                          .WORD   066667
1198  003764  153333                          .WORD   153333
1199  003766  066667                          .WORD   066667
1200  003770  153333                          .WORD   153333
1201  003772  066667                          .WORD   066667
1202  003774  153333                          .WORD   153333
1203  003776  066667                          .WORD   066667
1204  004000  153333                          .WORD   153333
1205  004002  066667                          .WORD   066667
1206
1207  004004  000000                  PAT14:  .WORD   000000          ;PATTERN 14
1208  004006  177777                          .WORD   177777
1209  004010  177777                          .WORD   177777
1210  004012  177777                          .WORD   177777
1211  004014  177777                          .WORD   177777
1212  004016  177777                          .WORD   177777
1213  004020  177777                          .WORD   177777
1214  004022  177777                          .WORD   177777
1215  004024  177777                          .WORD   177777
1216  004026  177777                          .WORD   177777
1217  004030  177777                          .WORD   177777
1218  004032  177777                          .WORD   177777
1219  004034  177777                          .WORD   177777
1220  004036  177777                          .WORD   177777
1221  004040  177777                          .WORD   177777
1222  004042  177777                          .WORD   177777
1223
1224  004044  177777                  PAT15:  .WORD   177777          ;PATTERN 15
1225  004046  000000                          .WORD   000000
1226  004050  000000                          .WORD   000000
1227  004052  000000                          .WORD   000000
1228  004054  000000                          .WORD   000000
1229  004056  000000                          .WORD   000000
1230  004060  000000                          .WORD   000000
1231  004062  000000                          .WORD   000000
1232  004064  000000                          .WORD   000000
1233  004066  000000                          .WORD   000000
1234  004070  000000                          .WORD   000000
1235  004072  000000                          .WORD   000000
1236  004074  000000                          .WORD   000000
1237  004076  000000                          .WORD   000000
1238  004100  000000                          .WORD   000000
1239  004102  000000                          .WORD   000000
1240
1241                          ;DPB (DATA PARAMETER BLOCK)
1242
1243  004104      000         DPB.A:  .BYTE   0               ;(0) DRIVE NUMBER
1244  004105      000                 .BYTE   0               ;(1) OFFSET VALUE OR FMT22, ECI, AND HCI
1245  004106      000                 .BYTE   0               ;(2) COMMAND
1246  004107      000                 .BYTE   0               ;(3) PSEL AND A17 AND A16
1247  004110  000000                  .WORD   0               ;(4) WORD COUNT (MUST BE NEG.)
1248  004112  047662                  .WORD   BUFFER          ;(6) BUFFER ADDRESS OR
1249                                                          ;REGISTER TABLE POINTER
1250  004114      000                 .BYTE   0               ;(10) SECTOR ADDRESS OR
1251                                                          ;FIRST REG. INDEX
```

```
1252  004115  000                       .BYTE   0               ;(11) TRACK ADDRESS OR
1253                                                            ;LAST REG. INDEX
1254  004116  000000                    .WORD   0               ;(12) CYLINDER ADDRESS
1255  004120  004204                    .WORD   RP.REG          ;(14) ERROR TABLE POINTER
1256                                                            ;POINTS TO THE FIRST OF TWENTY
1257                                                            ;LOCATIONS OF WHERE THE DRIVER
1258                                                            ;IS TO STORE THE RH11/RP04
1259                                                            ;REGISTERS ON AN ERROR. IF LEFT
1260                                                            ;ZERO REGISTERS ARE NOT SAVED.
1261  004122  000000                    .WORD   0               ;(16) STATUS/ERROR INDICATOR
1262                                                            ;BIT15=1=>ERROR OCCURRED
1263                                                            ;BIT07=1=>DONE
1264                                                            ;BIT14-BIT09 AND BIT06-BIT03
1265                                                            ;INDICATE TYPE OF ERROR
1266
1267  004124  000                DPB.B:  .BYTE   0               ;(0) DRIVE NUMBER
1268  004125  000                        .BYTE   0               ;(1) OFFSET VALUE OR FMT22, ECI, AND HCI
1269  004126  000                        .BYTE   0               ;(2) COMMAND
1270  004127  000                        .BYTE   0               ;(3) PSEL AND A17 AND A16
1271  004130  177776                     .WORD   -2              ;(4) WORD COUNT (MUST BE NEG.)
1272  004132  047662                     .WORD   BUFFER          ;(6) BUFFER ADDRESS OR
1273                                                            ;REGISTER TABLE POINTER
1274  004134  000                        .BYTE   0               ;(10) SECTOR ADDRESS OR
1275                                                            ;FIRST REG. INDEX
1276  004135  000                        .BYTE   0               ;(11) TRACK ADDRESS OR
1277                                                            ;LAST REG. INDEX
1278  004136  000000                     .WORD   0               ;(12) CYLINDER ADDRESS
1279  004140  004204                     .WORD   RP.REG          ;(14) ERROR TABLE POINTER
1280                                                            ;POINTS TO THE FIRST OF TWENTY
1281                                                            ;LOCATIONS OF WHERE THE DRIVER
1282                                                            ;IS TO STORE THE RH11/RP04
1283                                                            ;REGISTERS ON AN ERROR. IF LEFT
1284                                                            ;ZERO REGISTERS ARE NOT SAVED.
1285  004142  000000                    .WORD   0               ;(16) STATUS/ERROR INDICATOR
1286                                                            ;BIT15=1=>ERROR OCCURRED
1287                                                            ;BIT07=1=>DONE
1288                                                            ;BIT14-BIT09 AND BIT06-BIT03
1289                                                            ;INDICATE TYPE OF ERROR
1290
1291  004144  000                DPB.C:  .BYTE   0               ;(0) DRIVE NUMBER
1292  004145  000                        .BYTE   0               ;(1) OFFSET VALUE OR FMT22, ECI, AND HCI
1293  004146  000                        .BYTE   0               ;(2) COMMAND
1294  004147  000                        .BYTE   0               ;(3) PSEL AND A17 AND A16
1295  004150  177776                     .WORD   -2              ;(4) WORD COUNT (MUST BE NEG.)
1296  004152  047662                     .WORD   BUFFER          ;(6) BUFFER ADDRESS OR
1297                                                            ;REGISTER TABLE POINTER
1298  004154  000                        .BYTE   0               ;(10) SECTOR ADDRESS OR
1299                                                            ;FIRST REG. INDEX
1300  004155  000                        .BYTE   0               ;(11) TRACK ADDRESS OR
1301                                                            ;LAST REG. INDEX
1302  004156  000000                     .WORD   0               ;(12) CYLINDER ADDRESS
1303  004160  004204                     .WORD   RP.REG          ;(14) ERROR TABLE POINTER
1304                                                            ;POINTS TO THE FIRST OF TWENTY
1305                                                            ;LOCATIONS OF WHERE THE DRIVER
1306                                                            ;IS TO STORE THE RH11/RP04
1307                                                            ;REGISTERS ON AN ERROR. IF LEFT
```

```
1308
1309  004162  000000                        .WORD   0               ;ZERO REGISTERS ARE NOT SAVED.
1310                                                                ;(16) STATUS/ERROR INDICATOR
1311                                                                ;BIT15=1=>ERROR OCCURRED
1312                                                                ;BIT07=1=>DONE
1313                                                                ;BIT14-BIT09 AND BIT06-BIT03
1314                                                                ;INDICATE TYPE OF ERROR
1315  004164     000                DTADPB: .BYTE   0               ;(0) DRIVE NUMBER
1316  004165     000                        .BYTE   0               ;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
1317  004166     000                        .BYTE   0               ;(2) COMMAND
1318  004167     000                        .BYTE   0               ;(3) PSEL AND A17 AND A16
1319  004170  000000                        .WORD   0               ;(4) WORD COUNT (MUST BE NEG.)
1320  004172  047662                        .WORD   BUFFER          ;(6) BUFFER ADDRESS OR
1321                                                                ;REGISTER TABLE POINTER
1322  004174     000                        .BYTE   0               ;(10) SECTOR ADDRESS OR
1323                                                                ;FIRST REG. INDEX
1324  004175     000                        .BYTE   0               ;(11) TRACK ADDRESS OR
1325                                                                ;LAST REG. INDEX
1326  004176  000000                        .WORD   0               ;(12) CYLINDER ADDRESS
1327  004200  004204                        .WORD   RP.REG          ;(14) ERROR TABLE POINTER
1328                                                                ;POINTS TO THE FIRST OF TWENTY
1329                                                                ;LOCATIONS OF WHERE THE DRIVER
1330                                                                ;IS TO STORE THE RH11/RP04
1331                                                                ;REGISTERS ON AN ERROR. IF LEFT
1332                                                                ;ZERO REGISTERS ARE NOT SAVED.
1333  004202  000000                        .WORD   0               ;(16) STATUS/ERROR INDICATOR
1334                                                                ;BIT15=1=>ERROR OCCURRED
1335                                                                ;BIT07=1=>DONE
1336                                                                ;BIT14-BIT09 AND BIT06-BIT03
1337                                                                ;INDICATE TYPE OF ERROR
1338
1339
1340
1341                                    ;SAVE RH11/RP04 REGISTERS HERE ON ERROR
1342
1343  004204  000000                RP.REG: .WORD   0               ;RPCS1 (776700) CONTROL & STATUS #1
1344  004206  000000                        .WORD   0               ;RPWC (776702) WORD COUNT
1345  004210  000000                        .WORD   0               ;RPBA (776704) BUS ADDRESS
1346  004212  000000                        .WORD   0               ;RPDA (776706) DESIRED SECTOR/TRACK
1347  004214  000000                        .WORD   0               ;RPCS2 (776710) CONTROL & STATUS #2
1348  004216  000000                        .WORD   0               ;RPDS1 (776712) DISK STATUS
1349  004220  000000                        .WORD   0               ;RPER1 (776714) ERROR REG. #1
1350  004222  000000                        .WORD   0               ;RPAS (776716) ATTENTION SUMMARY
1351  004224  000000                        .WORD   0               ;RPLA (776720) LOOK AHEAD
1352  004226  000000                        .WORD   0               ;RPDB (776722) DATA BUFFER
1353  004230  000000                        .WORD   0               ;RPMR (776724) MAINTAINABILITY
1354  004232  000000                        .WORD   0               ;RPDT (776726) DRIVE TYPE
1355  004234  000000                        .WORD   0               ;RPSN (776730) SERIAL NUMBER
1356  004236  000000                        .WORD   0               ;RPOF (776732) OFFSET
1357  004240  000000                        .WORD   0               ;RPCA (776734) DESIRED CYLINDER
1358  004242  000000                        .WORD   0               ;RPCC (776736) CURRENT CYLINDER
1359  004244  000000                        .WORD   0               ;RPER2 (776740) ERROR REG #2
1360  004246  000000                        .WORD   0               ;RPER3 (776742) ERROR REG #3
1361  004250  000000                        .WORD   0               ;RPEC1 (776744) ECC POSITION
1362  004252  000000                        .WORD   0               ;RPEC2 (776746) ECC PATTERN
1363
```

```
1364                                        ;STATUS/ERROR MESSAGE POINTER TABLE
1365   004254   044672                      STATBL: .WORD   MSGB14          ;OFFLINE OR UNSAFE DRIVE REQUESTED
1366   004256   044734                              .WORD   MSGB13          ;UNLOAD DRIVE REQUESTED
1367   004260   044765                              .WORD   MSGB12          ;PERSISTENT UNSAFE
1368   004262   045007                              .WORD   MSGB11          ;PARITY ERROR OCCURRED
1369   004264   045035                              .WORD   MSGB10          ;FATAL PARITY ERROR
1370   004266   045060                              .WORD   MSGB09          ;SOFTWARE TIMEOUT ON THIS DRIVE
1371   004270   045117                              .WORD   MSGB08          ;SOFTWARE TIMEOUT ON ANOTHER DRIVE
1372   004272   045161                              .WORD   MSGB06          ;ERROR OCCURRED DURING I/O OPERATION
1373   004274   045225                              .WORD   MSGB05          ;ERROR OCCURRED DURING NON-I/O OPERATION
1374   004276   045275                              .WORD   MSGB04          ;UNSAFE OCCURRED
1375   004300   045315                              .WORD   MSGB03          ;AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
1376   004302   045365                              .WORD   MSGB02          ;DRIVE HAS NOT RESPONDED TO PORT REQUEST
1377   004304   045435                              .WORD   MSGB01          ;DRIVE HAS BECOME NONEXISTENT
```

```
                                ERROR POINTER TABLE

                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT

004306                          SERRTB:
                                ;*EM AND DH ARE ASCIZ MESSAGES, DT IS A STRING OF WORDS THAT POINT TO THE
                                ;*DATA TO BE TYPED AND DF IS A STRING OF DATA THAT TELL HOW THE DT WORDS
                                ;*ARE TO BE TYPED.  IF ANY OF THE POINTERS ARE NOT NEEDED FOR A PARTICULAR
                                ;*ERROR IT IS REPLACED WITH A ZERO.
                                ;*EACH OF THE ITEMS BELOW REFER TO THE ERROR NUMBER AND INDICATE
                                ;*THE INFORMATION THAT WILL BE TYPED WHEN THE ERROR OCCURS.
                                ;*UNLESS STATED OTHER ALL NUMBERS ARE OCTAL

                                ;*  ERROR ITEM 1
                                ;*    RH11 INTERRUPT OCCURED (RPAS = 0)
                                ;*    ERR PC  RPAS
                                ;*    SERRPC  SREG3

004306  044013                          EM1
004310  045473                          DH1
004312  047054                          DT1
004314  047506                          DF1

                                ;*  ERROR ITEM 2
                                ;*    UNEXPECTED ATTENTION OCCURRED
                                ;*    ERR PC  DRIVE   RPAS    RPOS1   RPER1     RPER2     RPER3
                                ;*    SERRPC  SREG1   SREG3   RPERRS  RPERRS+2  RPERRS+4  RPERRS+6

004316  044056                          EM2
004320  045510                          DH2
004322  047060                          DT2
004324  047512                          DF2

                                ;*  ERROR ITEM 3
                                ;*    MASSBUS PARITY ERROR (MCPE=1)
                                ;*    TEST    ERR PC  ADDRESS DATA
                                ;*    STMP0   SERRPC  RD.ADR  RD.WRD

004326  044114                          EM3
004330  045576                          DH3
004332  047076                          DT3
004334  047516                          DF3

                                ;*  ERROR ITEM 4
                                ;*    MASSBUS PARITY ERROR (PAR=1)
                                ;*    TEST    ERR PC  ADDRESS GDDATA  BDDATA
```

```
1434                                    ;*    $TMPO   $ERRPC  WRT.ADR WRT.WD  RD.WRD
1435
1436  004336  044151                            EM4
1437  004340  045633                            DH4
1438  004342  047106                            DT4
1439  004344  047522                            DF4
1440
1441                                    ;*  ERROR ITEM 5
1442                                    ;*    ADDRESS PLUG CHANGE BIT SET
1443                                    ;*    ERR PC  DRIVE   RPAS    RPDS1   RPER1     RPER2     RPER3
1444                                    ;*    $ERRPC  $REG1   $REG3   RPERRS  RPERRS+2  RPERRS+4 RPERRS+6
1445
1446  004346  044205                            EM5
1447  004350  045510                            DH2
1448  004352  047060                            DT2
1449  004354  047512                            DF2
1450
1451                                    ;*  ERROR ITEM 6 -- NOT USED
1452
1453  004356  000000                            0
1454  004360  000000                            0
1455  004362  000000                            0
1456  004364  000000                            0
1457
1458                                    ;*  ERROR ITEM 7 -- NOT USED
1459
1460  004366  000000                            0
1461  004370  000000                            0
1462  004372  000000                            0
1463  004374  000000                            0
1464
1465                                    ;*  ERROR ITEM 10
1466                                    ;*    RH11/RP04/5/6 FAILED TO RESPOND TO ADDRESSING
1467                                    ;*    RPCS1   ERR PC
1468                                    ;*    RH.ADR  $ERRPC
1469
1470  004376  044241                            EM10
1471  004400  045702                            DH10
1472  004402  047140                            DT10
1473  004404  047526                            DF10
1474
1475                                    ;*  ERROR ITEM 11
1476                                    ;*    DRIVE SELECTED IS NOT ONLINE
1477                                    ;*    DRIVE   ERR PC
1478                                    ;*    $REG2   $ERRPC
1479
1480  004406  044317                            EM11
1481  004410  045721                            DH11
1482  004412  047144                            DT11
1483  004414  047532                            DF11
1484
1485                                    ;*  ERROR ITEM 12
1486                                    ;*    IMPROPER HEADER DATA
1487                                    ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR
1488                                    ;*    $TMPO   $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS  SEC.DS
1489                                    ;*        GDCYL   GDTRK   GDSCTR  BDCYL   BDTRK   BDSCTR
```

```
1490                                    ;*          CYL.DS  TRK.DS  SEC.DS  CYL.RD  TRK.RD  SEC.RD
1491                                    ;*     CYLNDR, TRACK, AND SECTOR ARE DECIMAL
1492
1493  004416  044354                            EM12
1494  004420  045740                            DH12
1495  004422  047150                            DT12
1496  004424  047536                            DF12
1497
1498                                    ;*  ERROR ITEM 13
1499                                    ;*    DATA COMPARE FAILURE
1500                                    ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR
1501                                    ;*    STMPO   $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS  SEC.DS
1502                                    ;*       GDDAT   BDDAT   WRDCNT  GDADR   BDADR
1503                                    ;*       $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR
1504                                    ;*     CYLNDR, TRACK, SECTOR, AND WRDCNT  ARE DECIMAL
1505
1506  004426  044401                            EM13
1507  004430  045740                            DH12
1508  004432  047202                            DT13
1509  004434  047546                            DF13
1510
1511                                    ;*  ERROR ITEM 14 -- FOLLOWS #13
1512                                    ;*       $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR
1513
1514  004436  003000                            0
1515  004440  000000                            0
1516  004442  047220                            DT13A
1517  004444  047556                            DF14
1518
1519                                    ;*  ERROR ITEM 15
1520                                    ;*    DATA COMPARE FAILURE
1521                                    ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR
1522                                    ;*    STMPO   $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS  SEC.DS
1523                                    ;*       GDDAT   BDDAT   WRDCNT  GDADR   BDADR
1524                                    ;*       $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR
1525                                    ;*       CYLNDR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL
1526
1527  004446  044401                            EM13
1528  004450  045740                            DH12
1529  004452  047202                            DT13
1530  004454  047546                            DF13
1531
1532                                    ;*  ERROR ITEM 16 -- FOLLOWS #15
1533                                    ;*       $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR
1534
1535  004456  000000                            0
1536  004460  000000                            0
1537  004462  047220                            DT13A
1538  004464  047556                            DF14
1539
1540                                    ;*  ERROR ITEM 17
1541                                    ;*    DISK ERROR IN TIMING TEST
1542                                    ;*    TEST    ERR PC  DRIVE   RPCS1   RPDS1     RPER1     RPER2     RPER3
1543                                    ;*    STMPO   $ERRPC  CHKDRV  RP.REG  RP.REG+12 RP.REG+14 RP.REG+40 RP.REG+42
1544
1545  004466  044426                            EM17
```

```
                                ERROR POINTER TABLE

1546  004470  046154                    DH17
1547  004472  047232                    DT17
1548  004474  047562                    DF17
1549
1550                            ;*  ERROR ITEM 20
1551                            ;*    CLOCK (KW11-P) OVERFLOW IN TIMING TEST
1552                            ;*    TEST    ERR PC  DRIVE   RPCS1   RPDS1     RPER1     RPER2     RPER3
1553                            ;*    $TMP0   $ERRPC  CHKDRV  RP.REG  RP.REG+12 RP.REG+14 RP.REG+40 RP.REG+42
1554
1555  004476  044460                    EM20
1556  004500  046154                    DH17
1557  004502  047232                    DT17
1558  004504  047562                    DF17
1559
1560                            ;*  ERROR ITEM 21
1561                            ;*    DATA COMPARE FAILURE
1562                            ;*    TEST      ERR PC  TST PC  DRIVE   CYLNDR  TRACK
1563                            ;*    $TMP0     $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS
1564                            ;*        GDDAT   BDDAT   WRDCNT  SECTOR
1565                            ;*        $REG1   $BDDAT  $REG4   $REG1
1566                            ;*          CYLINDR, TRACK, WRDCNT, AND SECTOR ARE DECIMAL
1567
1568  004506  044401                    EM13
1569  004510  046252                    DH21
1570  004512  047252                    DT21
1571  004514  047566                    DF21
1572
1573                            ;*  ERROR ITEM 22--FOLLOWS #21
1574                            ;*        $REG1   $BDDAT   $REG4   $REG1
1575
1576  004516  000000                        0
1577  004520  000000                        0
1578  004522  047266                    DT[illegible]A
1579  004524  047576                    DF22
1580
1581                            ;*  ERROR ITEM 23
1582                            ;*      DISK ERROR DURING SEEK
1583                            ;*      TEST    ERR PC  DRIVE   CYLNDR  RPCS1   RPCS2     RPDS1
1584                            ;*      $TMP0   $ERRPC  CHKDRV  CYL.DS  RP.REG RP.REG+10 RP.REG+12
1585                            ;*          RPER1     RPER2     RPER3     RPCA      RPCC
1586                            ;*          RP.REG+14 RP.REG+40 RP.REG+42 RP.REG+34 RP.REG+36
1587
1588  004526  044527                    EM23
1589  004530  046367                    DH23
1590  004532  047276                    DT23
1591  004534  047602                    DF23
1592
1593                            ;*  ERROR ITEM 24
1594                            ;*      SEEK NOT COMPLETE WITHIN 120 MS
1595                            ;*      TEST    ERR PC  DRIVE   CYLNDR  RPCS1   RPCS2     RPDS1
1596                            ;*      $TMP0   $ERRPC  CHKDRV  CYL.DS  RP.REG RP.REG+10 RP.REG+12
1597                            ;*          RPER1     RPER2     RPER3     RPCA      RPCC
1598                            ;*          RP.REG+14 RP.REG+40 RP.REG+42 RP.REG+34 RP.REG+36
1599
1600  004536  044556                    EM24
1601  004540  046367                    DH23
```

ERROR POINTER TABLE

```
1602  004542  047276                  DT23
1603  004544  047602                  DF23
1604
1605
1606
1607                          ;;*****************************************************************
1608                          ;;*****************************************************************
1609                          ;*  ERROR ITEMS 23-40 NOT USED
1610                          ;*  ERROR ITEMS 41-46 WILL HAVE AN EM THAT
1611                          ;*  VARIES DEPENDING ON THE ERROR, IT WILL BE IN THE FORM:
1612                          ;*  RH11/RP04/5/6 ERROR (MESSAGE)
1613                          ;*  WHERE MESSAGE WILL BE ONE OR MORE OF THE FOLLOWING:
1614                          ;*  1)  OFFLINE OR UNSAFE DRIVE REQUESTED
1615                          ;*  2)  UNLOADED DRIVE REQUESTED
1616                          ;*  3)  PERSISTENT UNSAFE
1617                          ;*  4)  PARITY ERROR OCCURRED
1618                          ;*  5)  FATAL PARITY ERROR
1619                          ;*  6)  SOFTWARE TIMEOUT ON THIS DRIVE
1620                          ;*  7)  SOFTWARE TIMEOUT ON ANOTHER DRIVE
1621                          ;*  8)  ERROR OCCURRED DURING I/O OPERATION
1622                          ;*  9)  ERROR OCCURRED DURING NON-I/O OPERATION
1623                          ;*  10) UNSAFE OCCURRED
1624                          ;*  11) AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
1625
1626  004546                  ITEMH1:
1627
1628                          ;*  ERROR ITEM 41
1629                          ;*    RH11/RP04/5/6 ERROR (MESSAGE)
1630                          ;*    TEST    ERR PC  TST PC  DRIVE
1631                          ;*    STMP0   $ERRPC  $REG0   CHKDRV
1632
1633  004546  044616              EMH1
1634  004550  046522              DH41
1635  004552  047326              DT41
1636  004554  047612              DF41
1637
1638                          ;*  ERROR ITEM 42
1639                          ;*    RH11/RP04/5/6 ERROR (MESSAGE)
1640                          ;*    TEST    ERR PC  TST PC  DRIVE   RPCS1   RPCS2   RPDS1
1641                          ;*    STMP0   $ERRPC  $REG0   CHKDRV  RP.REG  RP.REG+10 RP.REG+12
1642
1643  004556  044616              EMH1
1644  004560  046560              DH42
1645  004562  047336              DT42
1646  004564  047616              DF42
1647
1648                          ;*  ERROR ITEM 43
1649                          ;*    RH11/RP04/5/6 ERROR (MESSAGE)
1650                          ;*    TEST    ERR PC  TST PC  DRIVE   RPCS1   RPCS2   RPDS1
1651                          ;*    STMP0   $ERRPC  $REG0   CHKDRV  RP.REG  RP.REG+10 RP.REG+12
1652                          ;*        RPER1     RPER2     RPER3
1653                          ;*        RP.REG+14 RP.REG+40 RP.REG+42
1654
1655  004566  044616              EMH1
1656  004570  046560              DH42
1657  004572  047354              DT43
```

ERROR POINTER TABLE

```
1658  004574  047622                          DF43
1659
1660                                  ;*  ERROR ITEM 44
1661                                  ;*    RH11/RP04/5/6 ERROR (MESSAGE)
1662                                  ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK     SECTOR
1663                                  ;*    $TMP0   $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS    SEC.DS
1664                                  ;*        RPCS1     RPCS2     RPDS1     RPCC      RPCA      RPDA
1665                                  ;*        RP.REG    RP.REG+10 RP.REG+12 RP.REG+36 RP.REG+34 RP.REG+06
1666                                  ;*      RPER1       RPER2       RPER3
1667                                  ;*      RP.REG+14   RP.REG+40   RP.REG+42
1668                                  ;*      CYLNDR,TRACK, AND SECTOR ARE DECIMAL
1669
1670  004576  044616                          EM41
1671  004600  045740                          DM12
1672  004602  047400                          DT44
1673  004604  047632                          DF44
1674
1675                                  ;*  ERROR ITEM 45
1676                                  ;*    RH11/RP04/5/6 ERROR (MESSAGE)
1677                                  ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK     SECTOR
1678                                  ;*    $TMP0   $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS    SEC.DS
1679                                  ;*        RPCS1     RPCS2     RPDS1     RPCC      RPCA      RPDA
1680                                  ;*        RP.REG    RP.REG+10 RP.REG+12 RP.REG+36 RP.REG+34 RP.REG+06
1681                                  ;*      RPER1       RPER2       RPER3       RPWC      RPBA      RPDB
1682                                  ;*      RP.REG+14   RP.REG+40   RP.REG+42   RP.REG+2   RP.REG+4 RP.REG+22
1683                                  ;*      CYLNDR,TRACK, AND SECTOR ARE DECIMAL
1684
1685  004606  044616                          EM41
1686  004610  045740                          DM12
1687  004612  047440                          DT45
1688  004614  047646                          DF45
1689
1690                                  ;*  ERROR ITEM 46
1691                                  ;*    FATAL WRITE CHECK ERROR (MESSAGE)
1692                                  ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK     SECTOR
1693                                  ;*    $TMP0   $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS    SEC.DS
1694                                  ;*        RPCS1     RPCS2     RPDS1     RPCC      RPCA      RPDA
1695                                  ;*        RP.REG    RP.REG+10 RP.REG+12 RP.REG+36 RP.REG+34 RP.REG+06
1696                                  ;*      RPER1       RPER2       RPER3       RPWC      RPBA      RPDB
1697                                  ;*      RP.REG+14   RP.REG+40   RP.REG+42   RP.REG+2   RP.REG+4 RP.REG+22
1698                                  ;*      CYLNDR,TRACK, AND SECTOR ARE DECIMAL
1699
1700  004616  044642                          EM46
1701  004620  045740                          DM12
1702  004622  047440                          DT45
1703  004624  047646                          DF45
1704
```

```
1705
1706                                    .SBTTL  START OF PROGRAM
1707
1708
1709  004626  012737  177777  001226  START3: MOV     #-1,@#BUSADR    ;GET BUSADR FLAG
1710  004634  000402                          BR      STRT1A
1711  004636  005037  001226          START1: CLR     @#BUSADR        ;CLR BUSADR FLAG
1712  004642  005037  001224          STRT1A: CLR     @#CNTRLC        ;NO CONTROL "C"
1713  004646  000411                          BR      START
1714  004650  012737  177777  001226  START4: MOV     #-1,@#BUSADR    ;SET BUSADR FLAG
1715  004656  000402                          BR      STRT2A
1716  004660  005037  001226          START2: CLR     @#BUSADR        ;CLR BUSADR FLAG
1717  004664  012737  177777  001224  STRT2A: MOV     #-1,@#CNTRLC    ;SET CONTROL "C" FLAG
1718  004672  000005                  START:  RESET
1719                                    .SBTTL  INITIALIZE THE COMMON TAGS
1720                                    ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1721  004674  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1722  004700  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1723  004702  022706  001140                  CMP     #SWR,R6 ;;DONE?
1724  004706  001374                          BNE     .-6             ;;LOOP BACK IF NO
1725  004710  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1726                                    ;;INITIALIZE A FEW VECTORS
1727  004714  012737  022564  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1728  004722  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
1729  004730  012737  017624  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1730  004736  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1731  004744  012737  023104  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1732  004752  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
1733  004760  012737  176543  023556          MOV     #176543,$HINUM  ;;PRIME THE RANDOM NUMBER GENERATOR
1734  004766  012737  123456  023560          MOV     #123456,$LONUM  ;;BOTH HIGH AND LOW WORDS
1735  004774  005037  001204                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
1736  005000  005037  001206                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1737  005004  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1738  005012  012737  005012  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1739  005020  012737  005020  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1740                                    ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1741                                    ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1742  005026  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1743  005032  012737  005066  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1744  005040  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1745  005046  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1746  005054  022777  177777  174056          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1747  005062  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1748                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
1749  005064  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
1750  005066  012716  005074          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1751  005072  000002                          RTI
1752  005074  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1753  005102  012737  000174  001142          MOV     #DISPREG,DISPLAY
1754  005110  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1755
1756  005114  012700  001160                  MOV     #$REGAD,R0      ;FIRST ADDRESS
1757  005120  005020                  1$:     CLR     (R0)+           ;CLEAR VARIABLE STORAGE
1758  005122  022700  001210                  CMP     #$BELL,R0       ;DONE?
1759  005126  001374                          BNE     1$              ;NO--BRANCH
1760  005130  013737  001414  001150          MOV     @#TPS,@#$TPS    ;SETUP THE STATUS AND BUFFER REG'S
```

```
1761  005136  013737  001416  001152          MOV     @#TPB,@#STPB    ;FOR THE TYPE ROUTINE
1762  005144  005227  177777                  INC     #-1             ;FIRST START ?
1763  005150  001026                          BNE     3$              ;BR IF NOT
1764  005152  104401  047662                  TYPE    ,TITLE          ;TYPE THE PROGRAM'S TITLE
1765  005156  005737  000042                  TST     @2              ;AUTO ACCEPT OR CHAIN MODE ?
1766  005162  001006                          BNE     2$              ;BR IF EITHER
1767  005164  122737  000011  000041          CMPB    #11,@1          ;LOADED FROM AN RP04/5/6 ?
1768  005172  001002                          BNE     2$              ;BR IF NOT
1769  005174  104401  047762                  TYPE    ,LOADRV         ;INSTRUCT THE OPERATOR TO REMOVE THE PACK
1770                                                                  ;ON DRIVE 0 IF DRIVE 0 IS TO BE TESTED
1771  005200  105737  000041          2$:     TSTB    @#41            ;LOADED FROM PAPER TAPE ?
1772  005204  001410                          BEQ     3$              ;BR IF NOT
1773  005206  004737  050256                  JSR     PC,$SIZE        ;SIZE THE MEMORY
1774  005212  023727  050352  100000          CMP     $LSTAD,#100000  ;16K OR MORE ON THE SYSTEM ?
1775  005220  103002                          BHIS    3$              ;BR IF YES
1776  005222  104401  050160                  TYPE    ,NOLOAD         ;INFORM THE OPEATOR THAT THE 'XXDP' LOADER
1777                                                                  ;WILL BE OVERWRITTEN
1778  005226  004737  021322          3$:     JSR     PC,$TKINT       ;TURN ON THE TTY KEYBOARD INTERRUPT
1779                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1780  005232  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1781  005236  001006                          BNE     67$             ;;BRANCH IF YES
1782  005240  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1783  005246  001005                          BNE     68$             ;;BRANCH IF NO
1784  005250  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1785  005252  000403                          BR      68$
1786  005254  112737  000001  001134  67$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1787  005262                          68$:
1788  005262  005227  177777                  INC     #-1             ;SEE IF FIRST START
1789  005266  001002                          BNE     SRTINT          ;BR IF NOT
1790  005270  004737  050354                  JSR     PC,GETADR       ;GET OR CHECK THE RH11 ADDRESS
1791  005274  104401  001215          SRTINT: TYPE    ,$CRLF          ;CR-LF
1792  005300  004737  024002                  JSR     PC,@#LP.AVL     ;CHECK FOR A LINE PRINTER
1793  005304  005037  177776                  CLR     @#PS            ;ENSURE THE PRIORITY = 0
1794  005310  012737  000001  001104          MOV     #1,$ICNT        ;SET ITERATION COUNT TO 1
1795  005316  004737  031374                  JSR     PC,@#GETSWR     ;GO CHECK FOR CONTROL SWITCHES
1796  005322  004737  024044                  JSR     PC,@#ST.CLK     ;INITIALIZE THE CLOCK
1797  005326  004737  034502          SETVEC: JSR     PC,RPINIT       ;CHECK THE DRIVE STATUS
1798  005332  012737  177777  034424          MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
1799  005340  062727  177777  000000          ADD     #-1,#0          ;FIRST START ?
1800  005346  103003                          BCC     11$             ;BR IF YES
1801  005350  005737  001224                  TST     CNTRLC          ;CONTROL 'C' SWITCH SET ?
1802  005354  001102                          BNE     SRTDRV          ;CONTINUE IF YES
1803  005356  012737  000340  177776  11$:    MOV     #PR7,PS         ;SET PRIORITY TO 7
1804  005364  005004                          CLR     R4              ;DRIVE TABLE POINTER
1805  005366  104401  001215                  TYPE    ,$CRLF          ;CR-LF
1806  005372  104401  043050                  TYPE    ,SYSTAT         ;TYPE STATUS HEADING
1807  005376                          1$:
1808  005376  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
1809                                                                  ;;TYPE DRIVE NUMBER
1810  005400  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1811  005402     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
1812  005403     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1813  005404  104401  044010                  TYPE    ,MSG.SP         ;SPACES
1814  005410  104401  044010                  TYPE    ,MSG.SP         ;SPACES
1815  005414  105764  034336                  TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
1816  005420  100416                          BMI     4$              ;BR IF UNSAFE
```

```
1817  005422  001020                        BNE     5$              ;BR IF ONLINE
1818  005424  105764  034346                TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
1819  005430  001404                        BEQ     2$              ;BR IF NONEXISTENT
1820  005432  100006                        BPL     3$              ;BR IF OFFLINE
1821  005434  104401  043144                TYPE    ,NOTRP          ;DRIVE NOT AN RP04/5/6
1822  005440  000440                        BR      9$              ;CHECK NEXT DRIVE
1823  005442  104401  043117        2$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
1824  005446  000435                        BR      9$              ;CHECK NEXT DRIVE
1825  005450  104401  043076        3$:     TYPE    ,UNTOFF         ;DRIVE OFFLINE
1826  005454  000405                        BR      6$              ;PRINT DRIVE TYPE
1827  005456  104401  043134        4$:     TYPE    ,NOTSAF         ;DRIVE UNSAFE
1828  005462  000402                        BR      6$              ;PRINT DRIVE TYPE
1829  005464  104401  043107        5$:     TYPE    ,UNTON          ;DRIVE ONLINE
1830  005470  104401  044010        6$:     TYPE    ,MSG.SP         ;SPACES
1831  005474  012737  043162  005540        MOV     #RP04B,B$       ;ADDRESS OF RP04 MESSAGE
1832  005502  132764  000001  034346        BITB    #BIT00,DRVTYP(R4)  ;RP0' ?
1833  005510  001012                        BNE     7$              ;BR IF YES
1834  005512  012737  043167  005540        MOV     #RP05,B$        ;ADDRESS OF RP05 MESSAGE
1835  005520  132764  000002  034346        BITB    #BIT01,DRVTYP(R4)  ;RP05 ?
1836  005526  001003                        BNE     7$              ;BR IF YES
1837  005530  012737  043174  005540        MOV     #RP06,B$        ;ADDRESS OF RP06 MESSAGE
1838  005536  104401                7$:     TYPE                    ;TYPE THE DRIVE TYPE MESSAGE
1839  005540  000000                B$:     .WORD   0               ;MESSAGE ADDRESS HERE
1840  005542  104401  001215        9$:     TYPE    ,$CRLF          ;CR-LF
1841  005546  005204                        INC     R4              ;INCREMENT DRIVE NUMBER/TABLE POINTER
1842  005550  020427  000010                CMP     R4,#8.          ;FINISHED ?
1843  005554  001310                        BNE     1$              ;BR IF NOT
1844  005556  005037  177775                CLR     PS              ;SET PRIORITY BACK TO '0'
1845  005562  005737  001224        SRTDRV: TST     @#CNTRLC        ;CONTROL "C" START/RESTART?
1846  005566  001417                        BEQ     1$              ;NO--BRANCH
1847  005570  013746  001222                MOV     SAVCSW,-(SP)    ;GET THE PREVIOUS 'C.SWR' CONTENTS
1848  005574  063716  001220                ADD     C.SWR,(SP)      ;SET UP TO SEE IF 'BIT00' IS DIFFERENT
1849  005600  032726  000001                BIT     #BIT00,(SP)+    ;IS 'BIT00' DIFFERENT ?
1850  005604  001405                        BEQ     9$              ;BR IF NOT
1851  005606  013737  001220  001222        MOV     C.SWR,SAVCSW    ;STORE PRESENT 'C.SWR' VALUE
1852  005614  004737  024322                JSR     PC,LODFLT       ;RESET PARAMETERS TO THEIR DEFAULT  ALUES
1853  005620  004737  031624        9$:     JSR     PC,@#GT.PRM     ;GET PARAMETERS
1854  005624  000420                        BR      4$
1855  005626  004737  024322        1$:     JSR     PC,LODFLT       ;SETUP DEFAULT PARAMETERS
1856  005632  005037  001232                CLR     DRVSEL          ;NO DRIVES SELECTED
1857  005636  005000                        CLR     R0              ;DETERMINE THE DRIVES THAT
1858  005640  012701  000001                MOV     #1,R1           ;ARE AVAILABLE FOR TESTING
1859  005644  105760  034336        2$:     TSTB    DRVSTA(R0)
1860  005650  003403                        BLE     3$
1861  005652  156037  034452  001232        BISB    ATABIT(R0),@#DRVSEL
1862  005660  005200                3$:     INC     R0
1863  005662  106301                        ASLB    R1
1864  005664  001367                        BNE     2$
1865  005666  005037  034426        4$:     CLR     @#SEEKFG        ;CLEAR SEEK FLAG
1866  005672  032737  000400  001220        BIT     #SWR8,@#C.SWR   ;DO SEEK BEFORE DATA TRANSFER?
1867  005700  001002                        BNE     5$              ;YES--BRANCH
1868  005702  005137  034426                COM     @#SEEKFG        ;NO
1869  005706  122737  000011  000041  5$:     CMPB    #11,41  ;LOADED FROM AN RP04/5/6 ?
1870  005714  001003                        BNE     10$             ;BR IF NOT
1871  005716  042737  000001  001232        BIC     #BIT00,DRVSEL   ;CLEAR THE DRIVE 0 SELECTION BIT
1872  005724  104401  043201        10$:    TYPE    ,DRIVES         ;'DRIVES(S) TO BE TESTED'
```

```
1873  005730  005037  017540                  CLR     @#$ENDCT        ;DETERMINE PASSES TO MAKE AND
1874  005734  005000                          CLR     R0              ;THE DRIVES TO BE TESTED
1875  005736  013701  001232                  MOV     @#DRVSEL,R1     ;ANY DRIVES SELECTED?
1876  005742  001004                          BNE     6$              ;YES--BRANCH
1877  005744  104401  043232                  TYPE    ,NONE           ;'NONE'
1878
1879  005750  000137  017356                  JMP     @#$EOP          ;GO TO END OF PROGRAM
1880  005754  006201                  6$:     ASR     R1              ;REPORT THE DRIVES TO BE TESTED
1881  005756  103011                          BCC     7$
1882  005760  005237  017540                  INC     @#$ENDCT        ;GIVE THIS DRIVE A PASS
1883  005764  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
1884  005766  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1885  005770     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
1886  005771     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1887  005772  005701                          TST     R1              ;MORE DRIVES?
1888  005774  001404                          BEQ     8$              ;NO--BRANCH
1889  005776  104401  043237                  TYPE    ,COMMA          ;','
1890  006002  005200                  7$:     INC     R0              ;FORM DRIVE NUMBER
1891  006004  000763                          BR      6$
1892  006006  013737  017540  017532  8$:     MOV     @#$ENDCT,@#$EOPCT
1893  006014  005737  001244                  TST     @#CLKSTA        ;KW11-P AVAILABLE
1894  006020  003006                          BGT     RSTRT1          ;YES--BRANCH
1895  006022  032737  036000  001234          BIT     #36000,@#TSTNMS ;NO--ANY TIMING TESTS TO BE PERFORMED?
1896  006030  001402                          BEQ     RSTRT1          ;NO--BRANCH
1897  006032  104401  043241                  TYPE    ,NOCLOK         ;'NO KW11-P CLOCK  TIMING TESTS WILL NOT BE PERFORMED'
1898  006036  005737  001232          RSTRT1: TST     DRVSEL          ;ANY DRIVES SELECTED ?
1899  006042  001002                          BNE     1$              ;BR IF YES
1900  006044  000137  004560                  JMP     START2          ;GET DRIVE SELECTION ENTRY
1901  006050  005037  001254          1$:     CLR     CHKDRV          ;INIT. THE CHECK DRIVE KEY
1902  006054  012737  000001  001256          MOV     #1,DRVMSK       ;START TO CHECK DESIRED DRIVES
1903  006062  033737  001256  001232  RSTRT2: BIT     DRVMSK,DRVSEL   ;IS THIS DRIVE SELECTED?
1904  006070  001010                          BNE     DRVOK           ;YES--GO CHECK IF DRIVE IS READY FOR TESTING
1905  006072  012706  001100          RESTART: MOV    #STACK,SP       ;SETUP THE STACK POINTER
1906  006076  005237  001254                  INC     CHKDRV          ;MOVE  TO NEXT DRIVE NUMBER
1907  006102  106337  001256                  ASLB    DRVMSK          ;POSITION THE MASK
1908  006106  103753                          BCS     RSTRT1          ;BRANCH IF THE DRIVE NUMBER NEEDS INITIALIZED
1909  006110  000764                          BR      RSTRT2
1910
1911  006112  013702  001254          DRVOK:  MOV     CHKDRV,R2       ;PICKUP THE DRIVE NUMBER
1912  006116  105762  034336                  TSTB    DRVSTA(R2)      ;IS DESIRED DRIVE ON-LINE?
1913  006122  003005                          BGT     1$              ;YES, BRANCH
1914  006124  104011                          ERROR   11              ;DRIVE SELECTED IS NOT ONLINE
1915  006126  043737  001256  001232          BIC     DRVMSK,DRVSEL   ;CLEAR DRIVE'S SELECTION BIT
1916  006134  000756                          BR      RESTART         ;RETURN
1917  006136  010237  004104          1$:     MOV     R2,@#DPB.A      ;SET THE DRIVE NUMBER INTO THE DPB'S
1918  006142  010237  004124                  MOV     R2,@#DPB.B
1919  006146  010237  004144                  MOV     R2,@#DPB.C
1920  006152  010237  004164                  MOV     R2,@#DTADPB
1921  006156  004737  024736                  JSR     PC,@#LDCMD      ;LOAD COMMAND INTO DPB.B AND DPB.C
1922  006162  012737  017356  001252          MOV     #$EOP,@#BYPASS  ;IF ERROR GO TO END OF PROGRAM
1923  006170  112737  000020  004105          MOVB    #FMT22/256.,DPB.A+1  ;ASSUME 16 BIT FORMAT
1924  006176  032737  000001  001220          BIT     #BIT00,C.SWR    ;16 BIT FORMAT REQUESTED ?
1925  006204  001402                          BEQ     2$              ;BR IF YES
1926  006206  105037  004105                  CLRB    DPB.A+1         ;CLEAR THE 'FMT22' BIT
1927  006212  112737  000143  004106  2$:     MOVB    #SETFORM,DPB.A+2  ;SET THE FORMAT BIT PER DPB.A+1
1928  006220  004037  025002                  JSR     R0,@#CALL.A     ;GO EXECUTE THE COMMAND
```

```
1929  006224  112737  000107  004106          MOVB    @RECAL,@#DPB.A+2        ;RECAL=COMMAND
1930  006232  004037  025002                  JSR     R0,@#CALL.A     ;GO EXECUTE THE COMMAND
1931  006236  104401  043327                  TYPE    ,TESTNG         ;'TESTING DRIVE '
1932  006242  010246                          MOV     R2,-(SP)        ;;SAVE R2 FOR TYPEOUT
1933  006244  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1934  006246     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
1935  006247     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1936  006250  104401  044010                  TYPE    ,MSG.SP         ;TYPE SPACES
1937  006254  104401  043351                  TYPE    ,SERIAL         ;'SERIAL NUMBER '
1938  006260  012700  000004                  MOV     #4,R0           ;FOUR DIGITS TO TYPE
1939  006264  013701  004234                  MOV     RP.REG+30,R1    ;SERIAL NUMBER
1940  006270  005002                  3$:     CLR     R2              ;ZERO
1941  006272  006101                          ROL     R1              ;PUT THE NEXT DIGIT
1942  006274  006102                          ROL     R2              ;INTO R2
1943  006276  006101                          ROL     R1
1944  006300  006102                          ROL     R2
1945  006302  006101                          ROL     R1
1946  006304  006102                          ROL     R2
1947  006306  006101                          ROL     R1
1948  006310  006102                          ROL     R2
1949  006312  062702  000060                  ADD     #'0,R2          ;MAKE IT ASCII
1950  006316  010227                          MOV     R2,(PC)+        ;SAVE IT
1951  006320  000000                  4$:     .WORD   0
1952  006322  104401  006320                  TYPE    ,4$             ;TYPE
1953  006326  005300                          DEC     R0              ;ALL DIGITS TYPED?
1954  006330  003357                          BGT     3$              ;NO -- BRANCH
1955  006332  104401  001215                  TYPE    ,$CRLF
1956  006336  113737  001364  001115          MOVB    ERR.CT,SERMAX   ;SETUP MAX ERROR COUNT
```

```
1957
1958                                    .SBTTL  **** TESTS ****
1959

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

        ;*IN THE DESCRIPTIONS OF THE BELOW TESTS THE VARIABLES USED
        ;*AND THEIR DEFAULT VALUES (UNLESS SPECIFIED OTHERWISE) ARE:
        ;*
        ;*MNEMONIC      VALUE           VARIABLE
        ;*--------      -----           --------
        ;*
        ;*R             1               ITERATIONS (REPEATS)
        ;*FC            0               FIRST CYLINDER ADDRESS
        ;*LC            410 OR 814      LAST CYLINDER ADDRESS
        ;*IC            1               INCREMENT VALUE
        ;*NC OF NC1     FC+IC           NEW OR MODIFIED CYLINDER
        ;*                              ADDRRESS
        ;*NC2           LC-IC           NEW OR MODIFIED CYLINDER
        ;*                              ADDRESS
        ;*
        ;*FT            0               FIRST TRACK ADDRESS
        ;*LT            18              LAST TRACK ADDRESS
        ;*IT            1               INCREMENT VALUE
        ;*NT            FT+IT           NEW OR MODIFIED TRACK ADDRESS
        ;*
        ;*FS            0               FIRST SECTOR ADDRESS
        ;*LS            21              LAST SECTOR ADDRESS

        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1992
1993                                    .SBTTL  *** SEEK TESTS ***
1994

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

        ;*THE SEEK TESTS WILL BE EXCUTED USING IMPLIED SEEKS. THESE
        ;*IMPLIED SEEKS WILL BE PERFORMED BY "READ HEADER AND
        ;*DATA" COMMANDS TO TRACK "FT" SECTOR "FS" OF THE DESIRED CYLINDER.
        ;*THE WORD COUNT WILL BE SET SUCH THAT ONLY THE CYLINDER AND
        ;*TRACK/SECTOR WORDS OF THE HEADER ARE READ.

        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

2010
2011                                    ;;*************************************************************
2012                                    ;*TEST 0        RECAL/SEEK TEST
```

```
                                        ;*      THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A RECALIBRATE
                                        ;*      COMMAND CYCLE AND THEN SEEK FORWARD TO CYLINDER "LC" AT
                                        ;*      THE COMPLETION OF BOTH COMMANDS STATUS INDICATIONS ARE
                                        ;*      CHECKED TO ENSURE NO ERRORS OCCURRED.
                                        ;;*****************************************************************
006344                                  TST0:
006344  000240                                  NOP
006346  033737  001424  001234                  BIT     @#BITS+(0*2),TSTNMS ;DO THIS TEST?
006354  001002                                  BNE     64$             ;YES--BRANCH
006356  000137  006506                          JMP     TST1            ;NO--GO TO THE NEXT TEST
006362  012737  000000  001102  64$:     MOV     #0,@#STSTNM             ;SET UP TEST NUMBER AND
                                                                        ;CLEAR THE ERROR FLAG (SERFLG)
006370  004737  024560                          JSR     PC,LOOPRM       ;LOAD THE PARMETERS FOR THE TEST
006374  012737  006470  001110                  MOV     #TEST0,@#SLPERR  ;SETUP THE LOOP ON ERROR ADDRESS
006402  013777  001102  172532                  MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
006410  013737  001506  001204                  MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
006416  112737  000031  001115                  MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
006424  112737  000107  004106                  MOVB    #RECAL,@#DPB.A+2        ;RECAL=COMMAND
006432  113737  001524  004134                  MOVB    @#FS,@#DPB.B+10 ;FS
006440  113737  001516  004135                  MOVB    @#FT,@#DPB.B+11 ;FT
006446  013737  001512  004136                  MOV     @#LC,@#DPB.B+12 ;LC
006454  012737  006504  001252                  MOV     #EXIT0,@#BYPASS ;GO TO EXIT0 ON ERROR
006462  012737  006470  001106                  MOV     #TEST0,$LPADR   ;SETUP LOOP ADDRESS
006470  012706  001100          TEST0:  MOV     #STACK,SP               ;SET UP STACK POINTER
006474  004037  025002                          JSR     R0,@#CALL.A     ;GO EXECUTE THE COMMAND
006500  004037  025114                          JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
006504  000004                  EXIT0:  SCOPE                           ;LOOP
                                        ;;*****************************************************************
                                        ;*TEST 1         SEEK/SEEK TEST
                                        ;*      THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A FORWARD SEEK
                                        ;*      CYCLE TO "LC", "LT", "LS" FOLLOWED BY A REVERSE SEEK CYCLE TO
                                        ;*      "FC", "FT", "FS".  AT THE COMPLETION OF EACH SEEK, THE PROPER
                                        ;*      INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.
                                        ;*      "LC" WILL DEFAULT TO 128 AND "FC", "FT", "LT", "FS", AND "LS"
                                        ;*      WILL DEFAULT TO 0
                                        ;;*****************************************************************
006506                                  TST1:
006506  000240                                  NOP
006510  033737  001426  001234                  BIT     @#BITS+(1*2),TSTNMS ;DO THIS TEST?
006516  001002                                  BNE     64$             ;YES--BRANCH
006520  000137  006664                          JMP     TST2            ;NO--GO TO THE NEXT TEST
006524  012737  000001  001102  64$:     MOV     #1,@#STSTNM             ;SET UP TEST NUMBER AND
                                                                        ;CLEAR THE ERROR FLAG (SERFLG)
006532  004737  024560                          JSR     PC,LOOPRM       ;LOAD THE PARMETERS FOR THE TEST
006536  012737  006646  001110                  MOV     #TEST1,@#SLPERR  ;SETUP THE LOOP ON ERROR ADDRESS
006544  013777  001102  172370                  MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
006552  013737  001506  001204                  MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
006560  112737  000031  001115                  MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
006566  113737  001524  004134                  MOVB    @#FS,@#DPB.B+10 ;FS
006574  113737  001526  004154                  MOVB    @#LS,@#DPB.C+10 ;LS
006602  113737  001516  004135                  MOVB    @#FT,@#DPB.B+11 ;FT
```

```
2069  006610  113737  001520  004155          MOVB    @#LT,@#DPB.C+11 ;LT
2070  006616  013737  001510  004136          MOV     @#FC,@#DPB.B+12 ;FC
2071  006624  013737  001512  004156          MOV     @#LC,@#DPB.C+12 ;LC
2072  006632  012737  006662  001252          MOV     #EXIT1,@#BYPASS ;GO TO EXIT1 ON ERROR
2073  006640  012737  006646  001106          MOV     #TEST1,$LPADR   ;SETUP LOOP ADDRESS
2074  006646  012706  001100          TEST1:  MOV     #STACK,SP       ;SET THE STACK POINTER
2075  006652  004037  025304                  JSR     R0,@#CALL.C     ;GO EXECUTE THE COMMAND
2076  006656  004037  025114                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2077  006662  000004                  EXIT1:  SCOPE                   ;LOOP
2078
2079                                  ;;*****************************************************************
2080                                  ;*TEST 2         INCREMENT/SEEK TEST
2081
2082                                  ;*      THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
2083                                  ;*      CYLINDER ADDRESS FROM "FC" TO "LC" BY THE INCREMENT "IC".
2084                                  ;*      WHEN THE RESULTANT CYLINDER ADDRESS (NC) EXCEEDS
2085                                  ;*      "LC" REVERSE SEEK CYCLES ARE INITIATED; STARTING
2086                                  ;*      AT THE LAST LEGAL "NC" AND DECREMENTING BY "IC"
2087                                  ;*      UNTIL "NC" IS LESS THAN "FC". AT THE COMPLETION OF EACH
2088                                  ;*      SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
2089                                  ;*      ENSURE PROPER OPERATION.
2090
2091                                  ;;*****************************************************************
2092  006664                          TST2:
2093  006664  000240                          NOP
2094  006666  033737  001430  001234          BIT     @#BITS+(2*2),TSTNMS ;DO THIS TEST?
2095  006674  001002                          BNE     64$             ;YES--BRANCH
2096  006676  000137  007102                  JMP     TST3            ;NO--GO TO THE NEXT TEST
2097  006702  012737  000002  001102  64$:    MOV     #2,@#STSTNM     ;SET UP TEST NUMBER AND
2098                                                                  ;CLEAR THE ERROR FLAG (SERFLG)
2099  006710  004737  024560                  JSR     PC,LOOPRM       ;LOAD THE PARMETERS FOR THE TEST
2100  006714  012737  006774  001110          MOV     #TEST2,@#$LPERR ;SETUP THE LOOP ON ERROR ADDRESS
2101  006722  013777  001102  172212          MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2102  006730  013737  001506  001204          MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
2103  006736  112737  000031  001115          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
2104  006744  012737  006752  001106          MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
2105  006752  113737  001524  004134  1$:     MOVB    @#FS,@#DPB.B+10 ;FS
2106  006760  113737  001516  004135          MOVB    @#FT,@#DPB.B+11 ;FT
2107  006766  012737  007100  001252          MOV     #EXIT2,@#BYPASS ;GO TO EXIT2 ON ERROR
2108  006774  013737  001510  004136  TEST2:  MOV     @#FC,@#DPB.B+12 ;FC
2109  007002  012737  007002  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
2110  007010  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
2111  007014                          INCSK:
2112  007014  004037  025114                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2113  007020  063737  001514  004136          ADD     @#IC,@#DPB.B+12 ;MOVE TO NEXT CYLINDER
2114  007026  023737  001512  004136          CMP     @#LC,@#DPB.B+12 ;OUT OF CYLINDERS?
2115  007034  002367                          BGE     INCSK           ;NO--BRANCH
2116  007036  013737  001512  004136          MOV     @#LC,@#DPB.B+12
2117  007044  012737  007044  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
2118  007052  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
2119  007056                          DECSK:
2120  007056  004037  025114                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2121  007062  163737  001514  004136          SUB     @#IC,@#DPB.B+12
2122  007070  023737  001510  004136          CMP     @#FC,@#DPB.B+12
2123  007076  003767                          BLE     DECSK
2124  007100  000004                  EXIT2:  SCOPE                   ;LOOP
```

```
                                        ;;*****************************************************************
                                        ;*TEST 3         STEPPING SEEK TEST
                                        ;
                                        ;*      THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0,1,2,4,
                                        ;*      8,16,32,64,128, AND 256.  AT THE COMPLETION OF EACH SEEK
                                        ;*      COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER
                                        ;*      OPERATION.
                                        ;
                                        ;;*****************************************************************
007102                                  TST3:
007102  000240                                  NOP
007104  033737  001432  001234                  BIT     @#BITS+(3*2),TSTNMS ;DO THIS TEST?
007112  001002                                  BNE     64$             ;YES--BRANCH
007114  000137  007276                          JMP     TST4            ;NO--GO TO THE NEXT TEST
007120  012737  000003  001102  64$:     MOV     #3,@#STSTNM             ;SET UP TEST NUMBER AND
                                                                        ;CLEAR THE ERROR FLAG (SERFLG)
007126  004737  024560                          JSR     PC,LOOPRM       ;LOAD THE PARMETERS FOR THE TEST
007132  012737  007212  001110                  MOV     #TEST3,@#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
007140  013777  001102  171774                  MOV     STSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
007146  013737  001206  001204                  MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
007154  112737  000031  001115                  MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
007162  012737  007170  001106                  MOV     #1$,$LPADR      ;SETUP TEST LOOP ADDRESS
007170  113737  001524  004134  1$:     MOVB    @#FS,@#DPB.B+10 ;FS
007176  113737  001516  004135                  MOVB    @#FT,@#DPB.B+11 ;FT
007204  012737  007274  001252                  MOV     #EXIT3,@#BYPASS ;GO TO BYPASS ON ERROR
007212  013737  001510  004136  TEST3:  MOV     @#FC,@#DPB.B+12 ;FC
007220  012737  007220  001110                  MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
007226  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
007232  004037  025114                          JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
007236  013701  001514                          MOV     IC,R1           ;CYLINDER 1
007242  012737  007242  001110                  MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
007250  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
007254  010137  004136          1$:     MOV     R1,@#DPB.B+12   ;DESIRED CYLINDER
007260  004037  025114                          JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
007264  006101                                  ASL     R1              ;MOVE TO NEXT CYLINDER
007266  020137  001512                          CMP     R1,@#LC         ;DONE?
007272  003770                                  BLE     1$              ;NO--LOOP
007274  000004                          EXIT3:  SCOPE
                                        ;
                                        ;;*****************************************************************
                                        ;*TEST 4         OSCILLATING SEEK TEST
                                        ;
                                        ;*      THIS TEST WILL COMMAND SEEK CYCLES FROM "FC" TO "NC" AND BACK
                                        ;*      TO "FC".  "NC" STARTS AT "FC" AND INCREMENTS BY "IC" UP TO CYLINDER
                                        ;*      "LC", THEN IS DECREMENTED BY "IC" BACK TO CYLINDER "FC".  AT THE
                                        ;*      COMPLETION OF EVERY SEEK COMMAND THE PROPER INDICATORS ARE
                                        ;*      EXAMINED TO ENSURE PROPER OPERATION.
                                        ;
                                        ;;*****************************************************************
007276                                  TST4:
007276  000240                                  NOP
007300  033737  001434  001234                  BIT     @#BITS+(4*2),TSTNMS ;DO THIS TEST?
007306  001002                                  BNE     64$             ;YES--BRANCH
007310  000137  007664                          JMP     $STS            ;NO--GO TO THE NEXT TEST
007314  012737  000004  001102  64$:     MOV     #4,@#STSTNM             ;SET UP TEST NUMBER AND
```

```
2181
2182  007322  004737  024560                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
2183  007326  012737  007422  001110          MOV     #TEST4,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
2184  007334  013777  001102  171600          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2185  007342  013737  001506  001204          MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
2186  007350  112737  000031  001115          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
2187  007356  012737  007364  001106          MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
2188  007364  113737  001524  004134  1$:     MOVB    @#FS,@#DPB.B+10 ;FS
2189  007372  113737  001516  004135          MOVB    @#FT,@#DPB.B+11 ;FT
2190  007400  012737  007662  001252          MOV     #EXIT4,@#BYPASS ;GO TO EXIT4 ON ERROR
2191  007406  005002                          CLR     R2              ;CLEAR STALL SWITCH (NO STALL)
2192  007410  032737  010000  001220          BIT     #SW12,@#C.SWR   ;STALL REQUIRED?
2193  007416  001401                          BEQ     TEST4           ;NO--BRANCH
2194  007420  005102                          COM     R2              ;YES--SET SWITCH
2195  007422  013701  001510          TEST4:  MOV     @#FC,R1         ;SET NC TO FC
2196  007426  005037  001336                  CLR     @#STALLO        ;START AT ZERO IF STALLS REQUIRED
2197  007432  012737  007432  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
2198  007440  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
2199  007444  010137  004136          1$:     MOV     R1,@#DPB.B+12   ;NC
2200  007450  004037  025114                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2201  007454  005702                          TST     R2              ;STALL?
2202  007456  001403                          BEQ     2$              ;NO--BRANCH
2203  007460  004037  026334                  JSR     R0,@#STALL      ;YES-GO TO STALL ROUTINE
2204  007464  001336                          .WORD   STALLO          ;TIME POINTER
2205  007466  013737  001510  004136  2$:     MOV     FC,@#DPB.B+12   ;FC
2206  007474  004037  025114                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2207  007500  005702                          TST     R2              ;STALL?
2208  007502  001413                          BEQ     3$              ;NO--BRANCH
2209  007504  004037  026334                  JSR     R0,@#STALL      ;YES--GO TO STALL ROUTINE
2210  007510  001336                          .WORD   STALLO          ;TIME POINTER
2211  007512  005237  001336                  INC     @#STALLO        ;UPDATE THE TIME
2212  007516  023737  001362  001336          CMP     @#MXSTAL,@#STALLO ;TIME TO BIG?
2213  007524  003347                          BGT     1$              ;NO--BRANCH
2214  007526  005037  001336                  CLR     @#STALLO        ;YES--START OVER AT ZERO
2215  007532  063701  001514          3$:     ADD     @#IC,R1         ;MOVE TO NEXT CYLINDER
2216  007536  020137  001512                  CMP     R1,@#LC         ;LAST CYLINDER COMPLETED?
2217  007542  003740                          BLE     1$              ;NO--BRANCH
2218  007544  013701  001512                  MOV     @#LC,R1         ;SET NC TO LC
2219  007550  012737  007550  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
2220  007556  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
2221  007562  010137  004136          4$:     MOV     R1,@#DPB.B+12   ;NC
2222  007566  004037  025114                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2223  007572  005702                          TST     R2              ;STALL?
2224  007574  001403                          BEQ     5$              ;NO--BRANCH
2225  007576  004037  026334                  JSR     R0,@#STALL      ;YES--GO TO STALL ROUTINE
2226  007602  001336                          .WORD   STALLO          ;TIME POINTER
2227  007604  013737  001512  004136  5$:     MOV     @#LC,@#DPB.B+12 ;LC
2228  007612  004037  025114                  JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2229  007616  005702                          TST     R2              ;STALL?
2230  007620  001413                          BEQ     6$              ;NO--BRANCH
2231  007622  004037  026334                  JSR     R0,@#STALL      ;YES--GO TO STALL ROUTINE
2232  007626  001336                          .WORD   STALLO          ;TIME POINTER
2233  007630  005237  001336                  INC     @#STALLO        ;UPDATE STALL TIME
2234  007634  023737  001362  001336          CMP     @#MXSTAL,@#STALLO ;TIME TOO BIG?
2235  007642  003347                          BGT     4$              ;NO--BRANCH
2236  007644  005037  001336                  CLR     @#STALLO        ;YES--SET STALL TIME BACK TO ZERO
```

```
                                T4      OSCILLATING SEEK TEST

2237  007650  163701  001514    6$:     SUB     @#IC,R1         ;NEXT CYLINDER
2238  007654  020137  001510            CMP     R1,@#FC         ;DONE?
2239  007660  002340                    BGE     4$              ;NO--BRANCH
2240  007662  000004            EXIT4:  SCOPE                   ;LOOP
2241
2242                            ;;**************************************************************
2243                            ;*TEST 5         CONVERGING/DIVERGING SEEK TEST
2244
2245                            ;*      THIS TEST WILL CAUSE THE DRIVE TO EXECUTE FORWARD AND REVERSE
2246                            ;*      SEEKS FROM "NC1" AND "NC2" RESPECTIVELY, "NC1" WILL BE INCREMENTED
2247                            ;*      BY "IC" AND "NC2" WILL BE DECREMENTED BY "IC" UNTIL "NC1" IS
2248                            ;*      GREATER THAN THE INITIAL VALUE OF "NC2" AND "NC2" IS
2249                            ;*      LESS THAN THE INITIAL VALUE OF "NC1".  AT THE COMPLETION OF
2250                            ;*      EACH SEEK COMPARE THE PROPER INDICATORS ARE EXAMINED TO
2251                            ;*      ENSURE PROPER OPERATION.  "NC1" AND "NC2" DEFAULT TO
2252                            ;*      "FC" AND "LC" RESPECTIVELY.
2253
2254                            ;;**************************************************************
2255  007664                    TST5:
2256  007664  000240                    NOP
2257  007666  033737  001436  001234    BIT     @#BITS+(5*2),TSTNMS ;DO THIS TEST?
2258  007674  001002                    BNE     64$             ;YES--BRANCH
2259  007676  000137  010064            JMP     TST6            ;NO--GO TO THE NEXT TEST
2260  007702  012737  000005  001102  64$:     MOV     #5,@#STSTNM     ;SET UP TEST NUMBER AND
2261                                                            ;CLEAR THE ERROR FLAG (SERFLG)
2262  007710  004737  024560            JSR     PC,LOOPRM       ;LOAD THE PARAMETERS FOR THE TEST
2263  007714  012737  007774  001110    MOV     #TEST5,@#SLPERR ;SETUP THE LOOP ON ERROR ADDRESS
2264  007722  013777  001102  171212    MOV     STSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2265  007730  013737  001506  001204    MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
2266  007736  112737  000031  001115    MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
2267  007744  012737  007752  001106    MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
2268  007752  113737  001524  004134  1$:     MOVB    @#FS,@#DPB.B+10 ;FS
2269  007760  113737  001516  004135            MOVB    @#FT,@#DPB.B+11 ;FT
2270  007766  012737  010062  001252            MOV     #EXIT5,@#BYPASS ;GO TO EXIT5 ON ERROR
2271  007774  013701  001510            TEST5:  MOV     @#FC,R1         ;START NC1 AT FC
2272  010000  013702  001512                    MOV     @#LC,R2         ;START NC2 AT LC
2273  010004  012737  010004  001110            MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
2274  010012  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
2275  010016  010137  004136            1$:     MOV     R1,@#DPB.B+12   ;NC1
2276  010022  004037  025114                    JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2277  010026  010237  004136                    MOV     R2,@#DPB.B+12   ;NC2
2278  010032  004037  025114                    JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2279  010036  063701  001514                    ADD     @#IC,R1         ;NEXT NC1
2280  010042  163702  001514                    SUB     @#IC,R2         ;NEXT NC2
2281  010046  020137  001512                    CMP     R1,@#LC         ;DONE?
2282  010052  003003                            BGT     EXIT5           ;YES--BRANCH
2283  010054  020237  001510                    CMP     R2,@#FC ;?
2284  010060  002356                            BGE     1$              ;NO--BRANCH
2285  010062  000004            EXIT5:  SCOPE                   ;LOOP
2286
2287                            ;;**************************************************************
2288                            ;*TEST 6         SERVO ADDRESSING LOGIC NOISE GENERATOR
2289
2290                            ;*      IN THIS TEST A SEEK IS DONE TO CYL "NC" THEN A SEEK TO
2291                            ;*      NC+4 THEN NC+1 THEN NC+3 THEN NC+2 THEN NC+5.  NOW "NC" IS UPDATED
2292                            ;*      BY "IC" AND THE ABOVE SEQUENCE IS REPEATED UNITL "LC" IS
```

```
                                        ;*      EXCEEDED BY ANY OF THE ABOVE VALUES.  THE INITIAL VALUE OF "NC"
                                        ;*      IS "FC".  AT THE COMPLETION OF EACH SEEK COMMAND THE
                                        ;*      PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.
                                        ;
                                        ;;*****************************************************************
010064                                  ;ST6:
010064  000240                                 NOP
010066  033737  001440  001234                 BIT     @#BITS+(6*2),TSTNMS ;DO THIS TEST?
010074  001002                                 BNE     64$                 ;YES--BRANCH
010076  000137  010330                         JMP     TST7                ;NO--GO TO THE NEXT TEST
010102  012737  000006  001102  64$:    MOV     #6,@#$TSTNM             ;SET UP TEST NUMBER AND
                                                                        ;CLEAR THE ERROR FLAG (SERFLG)
010110  004737  024560                         JSR     PC,LOOPRM           ;LOAD THE PARMETERS FOR THE TEST
010114  012737  010174  001110                 MOV     #TEST6,@#$LPERR     ;SETUP THE LOOP ON ERROR ADDRESS
010122  013777  001102  171012                 MOV     $TSTNM,@DISPLAY     ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
010130  013737  001506  001204                 MOV     @#RPT,$TIMES        ;GET THE ITERATION COUNT
010136  112737  000031  001115                 MOVB    #25.,$ERMAX         ;MAX ERRORS ALLOWED FOR TEST
010144  012737  010152  001106                 MOV     #1$,$LPADR          ;SETUP LOOP ADDRESS
010152  113737  001524  004134  1$:     MOVB    @#FS,@#DPB.B+10     ;FS
010160  113737  001516  004135                 MOVB    @#FT,@#DPB.B+11     ;FT
010166  012737  010326  001252                 MOV     #EXIT6,@#BYPASS     ;GO TO EXIT6 ON ERROR
010174  013701  001510          TEST6:  MOV     @#FC,R1             ;PICKUP "FC"
010200  013702  001512                 MOV     @#LC,R2             ;FORM LAST CYLINDER THAT
010204  162702  000005                 SUB     #5,R2               ;IS AVAILABLE FOR TESTING
010210  012737  010210  001110                 MOV     #.,$LPERR           ;SETUP THE ERROR LOOP ADDRESS
010216  012706  001100                 MOV     #STACK,SP           ;LOAD THE STACK POINTER
010222  020102                  1$:     CMP     R1,R2               ;LAST CYLINDER
010224  003040                         BGT     EXIT6               ;YES--BRANCH
010226  010137  004136                 MOV     R1,@#DPB.B+12       ;NC
010232  004037  025114                 JSR     R0,@#CALL.B         ;GO EXECUTE THE COMMAND
010236  062737  000004  004136         ADD     #4,@#DPB.B+12       ;NC+4
010244  004037  025114                 JSR     R0,@#CALL.B         ;GO EXECUTE THE COMMAND
010250  162737  000003  004136         SUB     #3,@#DPB.B+12       ;NC+1
010256  004037  025114                 JSR     R0,@#CALL.B         ;GO EXECUTE THE COMMAND
010262  062737  000002  004136         ADD     #2,@#DPB.B+12       ;NC+3
010270  004037  025114                 JSR     R0,@#CALL.B         ;GO EXECUTE THE COMMAND
010274  162737  000001  004136         SUB     #1,@#DPB.B+12       ;NC+2
010302  004037  025114                 JSR     R0,@#CALL.B         ;GO EXECUTE THE COMMAND
010306  062737  000003  004136         ADD     #3,@#DPB.B+12       ;NC+5
010314  004037  025114                 JSR     R0,@#CALL.B         ;GO EXECUTE THE COMMAND
010320  063701  001514                 ADD     @#IC,R1
010324  000736                         BR      1$
010326  000004                  EXIT6:  SCOPE                       ;LOOP

                                        ;;*****************************************************************
                                        ;*TEST 7        RANDOM SEEK TEST

                                        ;*      THIS TEST PERFORMS RANDOM SEEK OPERATIONS BETWEEN CYLINDERS 'FC'
                                        ;*      'LC'.  AFTER EACH SEEK, THE POSITION OF THE DRIVE IS VERIFIED BY
                                        ;*      READING A SECTOR FROM THE CURRENTLY ADDRESSED CYLINDER AND TRACK.
                                        ;*      THE TRACK ADDRESS IS INCREMENTED FOR EACH SEEK SO THAT VERIFICATION
                                        ;*      OF POSITIONING OCCURS USING EACH HEAD.  TRACK ADDRESSES ARE INCREMENTED
                                        ;*      BETWEEN PARAMTERS 'FT' AND 'LT'.

                                        ;;*****************************************************************
010330                                  ;TST7:
```

```
2349  010330  000240                          NOP
2350  010332  033737  001442  001234          BIT     @#BITS+<7*2>,TSTNM$ ;DO THIS TEST?
2351  010340  001002                          BNE     64$                 ;YES--BRANCH
2352  010342  000137  010710                  JMP     TST10               ;NO--GO TO THE NEXT TEST
2353  010346  012737  000007  001102  64$:    MOV     #7,@#STSTNM         ;SET UP TEST NUMBER AND
2354                                                                      ;CLEAR THE ERROR FLAG (SERFLG)
2355  010354  004737  024560                  JSR     PC,LOOPRM           ;LOAD THE PARMETERS FOR THE TEST
2356  010360  012737  010452  001110          MOV     #TEST7,@#SLPERR     ;SETUP THE LOOP ON ERROR ADDRESS
2357  010366  013777  001102  170546          MOV     $TSTNM,@DISPLAY     ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2358  010374  013737  001506  001204          MOV     @#RPT,$TIMES        ;GET THE ITERATION COUNT
2359  010402  112737  000031  001115          MOVB    #25.,$ERMAX         ;MAX ERRORS ALLOWED FOR TEST
2360  010410  113737  001516  004135          MOVB    FT,DPB.B+11         ;LOAD STARTING TRACK ADDRESS
2361  010416  112737  000105  004106          MOVB    #SEEK,@#DPB.A+2     ;SEEK=COMMAND
2362  010424  112737  000173  004126          MOVB    #READHD,DPB.B+2     ;READ HEADER & DATA COMMAND
2363  010432  013704  034464                  MOV     RPADR,R4            ;UNIBUS ADDRESS OF THE RH11
2364  010436  012737  010706  001252          MOV     #EXIT7,BYPASS       ;ERROR TERMINATION ADDRESS
2365  010444  012737  010452  001106          MOV     #TEST7,$LPADR       ;SETUP THE LOOP ON TEST ADDRESS
2366  010452  012706  001100          TEST7:  MOV     #STACK,SP           ;SETUP THE STACK POINTER
2367  010456  013737  001510  004136          MOV     FC,DPB.B+12         ;INITIAL CYLINDER ADDRESS
2368  010464  023737  001510  001512          CMP     FC,LC               ;CYLINDER LIMITS THE SAME ?
2369  010472  001422                          BEQ     1$                  ;BR IF THEY ARE
2370  010474  004737  023460                  JSR     PC,$RAND            ;CYCYLE THE RANDOM NUMBER GENERATOR
2371  010500  013746  023556                  MOV     $HINUM,-(SP)        ;USE THE HIGH RANDOM NUMBER
2372  010504  005046                          CLR     -(SP)               ;UPPER DIVIDEND
2373  010506  013746  001512                  MOV     LC,-(SP)            ;FORM THE DIVISOR
2374  010512  005216                          INC     (SP)                ;INCREMENT
2375  010514  163716  001510                  SUB     FC,(SP)             ;SUBTRACT THE LOWER LIMIT
2376  010520  004737  023562                  JSR     PC,$DIV             ;DIVIDE
2377  010524  062637  004136                  ADD     (SP)+,DPB.B+12      ;ADD THE REMAINDER TO THE INITIAL CYLINDER
2378  010530  005726                          TST     (SP)+               ;DISCARD THE QUOTENT
2379  010532  013737  004136  004116          MOV     DPB.B+12,DPB.A+12   ;COPY NEW CYLINDER ADDRESS
2380  010540                          1$:
2381  010540  012737  010540  001110          MOV     #.,$LPERR           ;SETUP THE ERROR LOOP ADDRESS
2382  010546  012706  001100                  MOV     #STACK,SP           ;LOAD THE STACK POINTER
2383  010552  004037  025002                  JSR     R0,@#CALL.A         ;GO EXECUTE THE COMMAND
2384  010556  012737  010556  001110          MOV     #.,$LPERR           ;SETUP THE ERROR LOOP ADDRESS
2385  010564  012706  001100                  MOV     #STACK,SP           ;LOAD THE STACK POINTER
2386  010570  113764  004104  000010          MOVB    DPB.A,RPCS2(R4)     ;SELECT THE DRIVE
2387  010576  014446  000020                  MOV     RPLA(R4),-(SP)      ;GET THE LOOK AHEAD REGISTER
2388  010602  006316                          ASL     (SP)                ;ALIGN THE SECTOR ADDRESS
2389  010604  006316                          ASL     (SP)                ;ALIGN THE SECTOR ADDRESS
2390  010606  000316                          SWAB    (SP)                ;PUT ADDRESS IN LOWER BYTE
2391  010610  135766  000001                  TSTB    1(SP)               ;IN THE 1ST 20% OF SECTOR ?
2392  010614  001401                          BEQ     2$                  ;BR IF YES
2393  010616  105216                          INCB    (SP)                ;INCREMENT THE SECTOR ADDRESS
2394  010620  105216                  2$:     INCB    (SP)                ;INCREMENT THE SECTOR ADDRESS
2395  010622  112637  004174                  MOVB    (SP)+,DTADPB+10     ;LOAD THE DPB
2396  010626  013746  001630                  MOV     PRMLMT+22,-(SP)     ;PUT LAST SECTOR ADDRESS ON THE STACK
2397  010632  005216                          INC     (SP)                ;INCREMENT IT
2398  010634  122637  004174                  CMPB    (SP)+,DTADPB+10     ;NEW SECTOR ADDRESS TOO LARGE ?
2399  010640  103007                          BHIS    4$                  ;BR IF NOT
2400  010642  103403                          BLO     3$                  ;BR IF ADDRESS IS 2 GREATER
2401  010644  105037  004174                  CLRB    DTADPB+10           ;RESET TO SECTOR ADDRESS 0
2402  010650  000403                          BR      4$                  ;CONTINUE
2403  010652  112737  000001  004174  3$:     MOVB    #1,DTADPB+10        ;RESET ADDRESS TO SECTOR 1
2404  010660                          4$:
```

```
2405  010660  004037  025114                 JSR     R0,@#CALL.B     ;GO EXECUTE THE COMMAND
2406  010664  105237  004135                 INCB    DPB.B+11        ;INCREMENT THE TRACK ADDRESS
2407  010670  123737  004135  001520         CMPB    DPB.B+11,LT     ;MAXIMUM ?
2408  010676  101403                         BLOS    EXIT7           ;BR IF NOT
2409  010700  113737  001516  004135         MOVB    FT,DPB.B+11     ;RELOAD STARTING TRACK ADDRESS
2410  010706  000004                 EXIT7:  SCOPE                   ;LOOP ?
2411
2412
2413                                 ;;*****************************************************************
2414                                 ;*TEST 10       SERVO SETTLE DOWN TEST
2415
2416                                 ;*      THIS TEST VERIFIES THAT THE SERVO HAS SETTLED DOWN AND THAT
2417                                 ;*      THE DRIVE IS ON CYLINDER WHEN THE DRIVE INDICATES SEEK COMPLETE.
2418                                 ;*      RANDOM SEEKS ARE ISSUED BETWEEN CYLINDERS 'NC1' AND 'NC1+IC'
2419                                 ;*      ('NC1' STARTS AT VALUE 'FC').  AT THE COMPLETION OF 1000 (10) SEEKS,
2420                                 ;*      'NC1' IS INCREMENTED BY VALUE 'IC' AND THE SEQUENCE IS REPEATED.
2421                                 ;*      THE TEST IS COMPLETED WHEN 'NC1' HAS BEEN INCREMENTED BEYOND 'LC'.
2422                                 ;*
2423                                 ;*      WHEN THE SEEK COMPLETES, THE PROGRAM READS THE DRIVE'S LOOK-AHEAD
2424                                 ;*      REGISTER (RPLA) TO DETERMINE THE ADDRESS OF THE SECTOR ROTATING  INTO
2425                                 ;*      POSITION.  THE PROGRAM THEN ISSUES A WRITE HEADER AND DATA COMMAND
2426                                 ;*      FOR THAT SECTOR.  IF THE DRIVE'S POSITIONER HAS NOT SETTLED DOWN OR
2427                                 ;*      IF THE POSITIONER IS NOT ON CYLINDER (IF THE DRIVE IS AN RP04, THE
2428                                 ;*      OFF CYLINDER CONDITION MUST LAST FOR AT LEAST 800 USEC, THE DRIVE
2429                                 ;*      WILL REPORT A 'WRU' ERROR.  (RP05/6'S MAY ALSO REPORT 'NHS' ERROR UNDER
2430                                 ;*      ERRORS IN THIS TEST INDICATE THAT THE SERVO SYSTEM MAY NOT BE ADJUSTED
2431                                 ;*      CORRECTLY, THAT THE DRIVE IS MALFUNCTIONING, OR THAT A PACK WITH
2432                                 ;*      MARGINAL SERVO TRACKS IS MOUNTED ON THE DRIVE.
2433                                 ;*
2434                                 ;*      THIS TEST USES THE EXTENTION BITS IN THE LOOK-AHEAD REGISTER TO DETERMINE
2435                                 ;*      WHETHER OR NOT IT CAN PICK UP THE SECTOR ROTATING INTO POSITION.  THE
2436                                 ;*      TEST IS OPTIMIZED SUCH THAT IF THE DRIVE SIGNALS SEEK DONE WITHIN
2437                                 ;*      THE FIRST 80% OF THE SECTOR CURRENTLY UNDER THE HEAD, THE TEST WILL
2438                                 ;*      TRY TO ADDRESS THE NEXT SECTOR.  BASED ON OBSERVATION, THE PROGRAM
2439                                 ;*      IS ABLE TO START THE OPERATION WITHOUT LOSING A REVOLUTION MOST OF
2440                                 ;*      THE TIME.
2441                                 ;*
2442                                 ;*      THIS TEST IS VALID ONLY IF THE OPERATION IS STARTED WITHIN A FEW
2443                                 ;*      HUNDRED MICRO-SECONDS AFTER SEEK DONE OCCURS.  THE NECESSARY TIME
2444                                 ;*      TIME DEPENDENT PARAMETERS OCCUR FREQUENTLY ENOUGH WITHIN THE REQUIRED
2445                                 ;*      RANGE TO PERMIT THIS TEST TO BE EFFECTIVE.
2446
2447                                 ;;*****************************************************************
2448  010710                         TST10:
2449  010710  000240                         NOP
2450  010712  033737  001444  001234         BIT     @#BITS+(10*2),TSTNMS ;DO THIS TEST?
2451  010720  001002                         BNE     64$             ;YES--BRANCH
2452  010722  000137  012014                 JMP     TST11           ;NO--GO TO THE NEXT TEST
2453  010726  012737  000010  001102  64$:   MOV     #10,@#STSTNM    ;SET UP TEST NUMBER AND
2454                                                                 ;CLEAR THE ERROR FLAG (SERFLG)
2455  010734  004737  024560                 JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
2456  010740  012737  011116  001110         MOV     #TEST10,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
2457  010746  013777  001102  170166         MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2458  010754  013737  001506  001204         MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
2459  010762  112737  000031  001115         MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
2460  010770  012737  010776  001106         MOV     #1$,$LPADR      ;SETUP THE LOOP ADDRESS
```

```
2461  010776                          1$:
2462  010776  112737  000105  004106          MOVB    #SEEK,@#DPB.A+2 ;SEEK=COMMAND
2463  011004  112737  000161  004166          MOVB    #WRI.E,DTADPB+2 ;COMMAND
2464  011012  113737  001516  004175          MOVB    FT,DTADPB+11    ;TRACK ADDRESS FOR THE WRITE
2465  011020  013737  001510  004116          MOV     FC,DPB.A+12     ;CYLINDER ADDRESS FOR THE SEEK
2466  011026  013737  001510  004176          MOV     FC,DTADPB+12    ;CYLINDER ADDRESS FOR THE WRITE
2467  011034  013737  001510  001532          MOV     FC,NC1          ;STARTING CYLINDER
2468  011042  013737  001514  001350          MOV     IC,DELTA        ;CYLINDER INCREMENT VALUE
2469  011050  012737  176000  004170          MOV     #-(256.*4.),DTADPB+4  ;WORD COUNT
2470  011056  012737  047662  004172          MOV     #BUFFER,DTADPB+6  ;BUFFER ADDRESS
2471  011064  005000                          CLR     R0              ;PATTERN POINTER (HC PATTERN)
2472  011066  004737  030330                  JSR     PC,SETBUF       ;LOAD THE WRITE BUFFER
2473  011072  005001                          CLR     R1              ;CLEAR REGISTER
2474  011074  113701  004104                  MOVB    DPB.A,R1        ;LOAD DRIVE ADDRESS
2475  011100  013704  034464                  MOV     RPADR,R4        ;UNIBUS ADDRESS OF THE RH11
2476  011104  004737  042466                  JSR     PC,CLRQUE       ;CLEAR THE OPERATION QUEUES
2477  011110  012737  012012  001252          MOV     #EXIT10,BYPASS  ;ERROR EXIT FROM TEST
2478  011116                          TEST10:
2479  011116  012737  011116  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
2480  011124  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
2481  011130  012737  000340  177776          MOV     #PR7,@#PS       ;SET PRIORITY TO 7
2482  011136  005737  001244                  TST     CLKSTA          ;SEE WHICH CLOCK ON SYSTEM
2483  011142  001415                          BEQ     3$              ;BR IF NO CLOCK
2484  011144  100405                          BMI     1$              ;BR IF KW11-L CLOCK
2485  011146  017746  170222                  MOV     @PKV,-(SP)      ;SAVE THE VECTOR
2486  011152  013746  001374                  MOV     PKV,-(SP)       ;SAVE THE VECTOR ADDRESS
2487  011156  000404                          BR      2$              ;CONTINUE
2488  011160  017746  170222          1$:     MOV     @LKV,-(SP)      ;SAVE THE 'L' CLOCK VECTOR
2489  011164  013746  001406                  MOV     LKV,-(SP)       ;SAVE THE VECTOR ADDRESS
2490  011170  012776  011746  000100  2$:     MOV     #TST10B,@(SP)   ;CHANGE THE VECTOR
2491  011176  012777  027022  000232  3$:     MOV     #DORTI,@RPVEC   ;CHANGE THE RP04/RP05 VECTOR
2492  011204  012737  000010  001344          MOV     #8.,SEKTMR      ;LOAD THE SEEK TIMER
2493  011212  012764  000040  000010          MOV     #BIT05,RPCS2(R4)  ;INIT THE MASSBUS
2494  011220  110164  000010                  MOVB    R1,RPCS2(R4)    ;RESELECT THE DRIVE
2495  011224  013764  004116  000034          MOV     DPB.A+12,RPCA(R4)  ;LOAD THE CYLINDER ADDRESS
2496  011232  013737  004116  001270          MOV     DPB.A+12,CYL.DS ;CYLINDER ADDRESS FOR ERROR MESSAGE
2497  011240  012764  000105  000000          MOVB    #SEEK,RPCS1(R4) ;START THE SEEK
2498  011246  005037  177776                  CLR     @#PS            ;CLEAR THE PRIORITY
2499  011252  105764  000012          4$:     TSTB    RPDS1(R4)       ;HAS THE DRIVE FINISHED ?
2500  011256  100402                          BMI     5$              ;BR IF IT HAS
2501  011260  000001                          WAIT                    ;WAIT FOR THE OPERATION TO COMPLETE
2502  011262  000773                          BR      4$              ;CONTINUE
2503  011264  012737  000340  177776  5$:     MOV     #PR7,@#PS       ;CHANGE PRIORITY TO MAX
2504  011272  032764  040000  000012          BIT     #BIT14,RPDS1(R4)  ;ERROR ?
2505  011300  001412                          BEQ     6$              ;BR IF NOT
2506  011302  012702  004104                  MOV     #DPB.A,R2       ;DPB POINTER
2507  011306  004737  042004                  JSR     PC,SVRH11       ;SAVE THE REGISTERS
2508  011312  104023                          ERROR   23              ;ERROR DURING SEEK
2509  011314  012764  000040  000010          MOV     #BIT05,RPCS2(R4)  ;INIT THE MASSBUS
2510  011322  110164  000010                  MOVB    R1,RPCS2(R4)    ;RESELECT THE DRIVE
2511  011326  012777  037346  023132  6$:     MOV     #ISR,@RPVEC     ;SETUP THE RP04/RP05 VECTOR
2512  011334  005737  001244                  TST     CLKSTA          ;WHICH CLOCK
2513  011340  001405                          BEQ     TST10A          ;BR IF NONE
2514  011342  016676  000002  000000          MOV     2(SP),@(SP)     ;RELOAD THE CLOCK VECTOR
2515  011350  062706  000004                  ADD     #4,SP           ;CORRECT THE STACK POINTER
2516  011354                          TST10A:
```

```
2517  011354  012737  011354  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
2518  011362  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
2519  011366  110164  000010                  MOVB    R1,RPCS2(R4)    ;SELECT THE DRIVE
2520  011372  016446  000020                  MOV     RPLA(R4),-(SP)  ;GET THE LOOK AHEAD REGISTER
2521  011376  006316                          ASL     (SP)            ;ALIGN THE SECTOR ADDRESS
2522  011400  006316                          ASL     (SP)            ;ALIGN THE SECTOR ADDRESS
2523  011402  000316                          SWAB    (SP)            ;PUT ADDRESS IN LOWER BYTE
2524  011404  122766  000300  000001          CMPB    #300,1(SP)      ;IN THE LAST 20% OR SECTOR ?
2525  011412  001001                          BNE     2$              ;BR IF NOT
2526  011414  105216                          INCB    (SP)            ;INCREMENT THE SECTOR ADDRESS
2527  011416  105216                  2$:     INCB    (SP)            ;INCREMENT THE SECTOR ADDRESS
2528  011420  112637  004174                  MOVB    (SP)+,DTADPB+10 ;LOAD THE DPB
2529  011424  013746  001630                  MOV     PRMLMT+22,-(SP) ;PUT MAXIMUM SECTOR ADDRESS ON THE STACK
2530  011430  005216                          INC     (SP)            ;INCREMENT PAST THE MAXIMUM ADDRESS
2531  011432  122637  004174                  CMPB    (SP)+,DTADPB+10 ;NEW SECTOR ADDRESS TOO LARGE ?
2532  011436  101007                          BHI     4$              ;BR IF NOT
2533  011440  103403                          BLO     3$              ;BR IF ADDRESS IS 2 GREATER THAN MAXIMUM
2534  011442  105037  004174                  CLRB    DTADPB+10       ;RESET TO SECTOR ADDRESS 0
2535  011446  000403                          BR      4$              ;CONTINUE
2536  011450  112737  000001  004174  3$:     MOVB    #1,DTADPB+10    ;RESET ADDRESS TO SECTOR 1
2537  011456  012703  004170          4$:     MOV     #DTADPB+4,R3    ;POINTER
2538  011462  012764  000111  000000          MOV     #DRVCLR,RPCS1(R4)  ;CLEAR THE DRIVE
2539  011470  012364  000002                  MOV     (R3)+,RPWC(R4)  ;LOAD THE WORD COUNT
2540  011474  012364  000004                  MOV     (R3)+,RPBA(R4)  ;LOAD THE BUFFER ADDRESS
2541  011500  012364  000006                  MOV     (R3)+,RPDA(R4)  ;LOAD THE TRACK/SECTOR ADDR
2542  011504  005037  004202                  CLR     DTADPB+16       ;RESET 'DONE' INDICATOR
2543  011510  012737  004164  034376          MOV     #DTADPB,TRNSWT  ;LOAD 'TRANSFER' DPB ADDRESS
2544  011516  010137  034450                  MOV     R1,DTUN         ;ADDRESS OF DRIVE TRANSFERING
2545  011522  112761  000001  034326          MOVB    #1,DRVACT(R1)   ;SET DRIVE ACTIVE INDICATOR
2546  011530  006301                          ASL     R1              ;SHIFT DRIVE ADDRESS
2547  011532  012761  001750  034430          MOV     #1000.,TIMER(R1)  ;SETUP THE OPERATION TIMER
2548  011540  006201                          ASR     R1              ;RESTORE R1
2549  011542  013764  004166  000000          MOV     DTADPB+2,RPCS1(R4)  ;START THE OPERATION
2550  011550  005037  177776                  CLR     @#PS            ;CLEAR THE PRIORITY
2551  011554  004037  025514                  JSR     R0,DRVCL1       ;WAIT FOR OPERATION TO COMPLETE
2552  011560  023727  001346  001750  5$:     CMP     SEKCNT,#1000.   ;FINISHED SEEKS ?
2553  011566  001026                          BNE     6$              ;BR IF NOT
2554  011570  005037  001346                  CLR     SEKCNT          ;CLEAR THE SEEK COUNT
2555  011574  063737  001514  001532          ADD     IC,NC1          ;ADD THE INCREMENT
2556  011602  023737  001532  001512          CMP     NC1,LC          ;EXCEEDED THE CYLINDER LIMIT ?
2557  011610  103100                          BHIS    EXIT10          ;BR IF IT HAS
2558  011612  013737  001512  001350          MOV     LC,DELTA        ;GET THE NEXT 'ZONE' ADDRESS
2559  011620  163737  001532  001350          SUB     NC1,DELTA       ;CHECK THE DIFFERENCE
2560  011626  023737  001514  001350          CMP     IC,DELTA        ;DIFFERENCE GREATER THAN THE INCREMENT ?
2561  011634  101003                          BHI     6$              ;BR IF IT IS
2562  011636  013737  001514  001350          MOV     IC,DELTA        ;USE THE ICREMENT PARAMETER
2563  011644  005237  001346          6$:     INC     SEKCNT          ;COUNT THE NEXT SEEK
2564  011650  023737  001510  001512          CMP     FC,LC           ;BEGINNING AND ENDING CYLINDERS THE SAME ?
2565  011656  001002                          BNE     7$              ;BR IF NOT
2566  011660  000137  011116                  JMP     TEST10          ;BR IF THEY ARE
2567  011664  013737  001532  004116  7$:     MOV     NC1,DPB.A+12    ;RESET THE CYLINDER ADDRESS
2568  011672  004737  023460                  JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
2569  011676  013746  023556                  MOV     $HINUM,-(SP)    ;USE THE HIGH RANDOM NUMBER
2570  011702  005046                          CLR     -(SP)           ;CLEAR THE UPPER DIVIDEND
2571  011704  013746  001350                  MOV     DELTA,-(SP)     ;FORM THE DIVISOR
2572  011710  005216                          INC     (SP)            ;INCREMENT
```

```
2573  011712  004737  023562                  JSR     PC,$DIV          ;DIVIDE
2574  011716  062637  004116                  ADD     (SP)+,DPB.A+12   ;ADD THE REMAINDER TO THE INITIAL CYLINDER
2575  011722  005726                          TST     (SP)+            ;DISCARD THE QUOTENT
2576  011724  023737  004116  004176          CMP     DPB.A+12,DTADPB+12  ;SAME CYLINDER SELECTED AS LAST TIME ?
2577  011732  001754                          BEQ     7$               ;BR IF IT WAS
2578  011734  013737  004116  004176          MOV     DPB.A+12,DTADPB+12  ;COPY NEW CYLINDER ADDRESS
2579  011742  000137  011116                  JMP     TEST10           ;CONTINUE
2580  011746  005337  001344          TST10B: DEC     SEKTMR           ;DECREMENT THE SEEK TIMER
2581  011752  001016                          BNE     1$               ;CONTINUE IF NOT DONE
2582  011754  012702  004104                  MOV     #DPB.A,R2        ;DPB ADDRESS
2583  011760  004737  042004                  JSR     PC,SVRH11        ;SAVE THE REGISTERS
2584  011764  104024                          ERROR   24               ;TIMEOUT DURING SEEK
2585  011766  012764  000040  000010          MOV     #BIT05,RPCS2(R4)   ;INIT THE MASSBUS
2586  011774  110164  000010                  MOVB    R1,RPCS2(R4)     ;RESELECT THE DRIVE
2587  012000  016676  000002  000000          MOV     2(SP),@(SP)      ;RESTORE THE CLOCK VECTOR ADDRESS
2588  012006  000401                          BR      EXIT10           ;ABORT THE TEST
2589  012010  000002                  1$:     RTI                      ;RETURN
2590  012012  000004                  EXIT10: SCOPE                    ;LOOP ?
2591
2592                                  ;;*****************************************************************
2593                                  ;*TEST 11       ALL SEEKS TEST
2594
2595                                  ;*      THIS TEST VERIFIES THAT THE DISK DRIVE CAN SEEK FROM EACH CYLINDER
2596                                  ;*      TO ALL OTHER CYLINDERS.
2597                                  ;*
2598                                  ;*      BEGINNING WITH CYLINDER 'FC', THE TEST SEEKS TO EACH CYLINDER
2599                                  ;*      BETWEEN 'FC' AND 'LC' FROM CYINDER 'FC'.  THE BEGINNING CYLINDER
2600                                  ;*      ADDRESS IS INCREMENTED AND THE TEST SEEKS BETWEEN THE NEW CYLINDER
2601                                  ;*      ADDRESS AND ALL CYLINDERS BETWEEN 'FC' AND 'LC'.  THE SEQUENCE
2602                                  ;*      CONTINUES UNTIL ALL CYLINDERS HAVE BEEN CHECKED.
2603
2604                                  ;;*****************************************************************
2605  012014                          TST11:
2606  012014  000240                          NOP
2607  012016  033737  001446  001234          BIT     @#BITS+(11*2),TSTNMS ;DO THIS TEST?
2608  012024  001002                          BNE     64$              ;YES--BRANCH
2609  012026  000137  012234                  JMP     TST12            ;NO--GO TO THE NEXT TEST
2610  012032  012737  000011  001102  64$:    MOV     #11,@#STSTNM     ;SET UP TEST NUMBER AND
2611                                                                   ;CLEAR THE ERROR FLAG (SERFLG)
2612  012040  004737  024560                  JSR     PC,LODPRM        ;LOAD THE PARMETERS FOR THE TEST
2613  012044  012737  012154  001110          MOV     #TEST11,@#LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
2614  012052  013777  001102  167062          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2615  012060  013737  001506  001204          MOV     @#RPT,$TIMES     ;GET THE ITERATION COUNT
2616  012066  112737  000031  001115          MOVB    #25.,$ERMAX      ;MAX ERRORS ALLOWED FOR TEST
2617  012074  012737  012102  001106          MOV     #1$,$LPADR       ;SETUP THE LOOP ADDRESS
2618  012102  113737  001524  004134  1$:     MOVB    FS,DPB.B+10      ;SECTOR ADDRESS
2619  012110  113737  001524  004154          MOVB    FS,DPB.C+10      ;SECTOR ADDRESS
2620  012116  113737  001516  004135          MOVB    FT,DPB.B+11      ;TRACK ADDRESS
2621  012124  113737  001516  004155          MOVB    FT,DPB.C+11      ;TRACK ADDRESS
2622  012132  013737  001510  004136          MOV     FC,DPB.B+12      ;STARTING CYLINDER ADDRESS
2623  012140  013737  001510  004156          MOV     FC,DPB.C+12      ;STARTING CYLINDER ADDRESS
2624  012146  012737  012232  001252          MOV     #EXIT11,BYPASS   ;TEST ABORT EXIT
2625  012154  012706  001100          TEST11: MOV     #STACK,SP        ;SETUP THE STACK POINTER
2626  012160                          1$:
2627  012160  004037  025304                  JSR     R0,@#CALL.C      ;GO EXECUTE THE COMMAND
2628  012164  004037  025114                  JSR     R0,@#CALL.B      ;GO EXECUTE THE COMMAND
```

```
 2629  012170  063737  001514  004156            ADD     IC,DPE C+12     ;INCREMENT THE ENDING CYLINDER ADDRESS
 2630  012176  023737  001512  004156            CMP     LC,DPB.C+12     ;CHECK IF EXCEEDING MAXIMUM
 2631  012204  002365                            BGE     1$              ;BR IF NOT
 2632  012206  013737  001510  004156            MOV     FC,DPB.C+12     ;RESET ENDING CYLINDER ADDRESS
 2633  012214  063737  001514  004136            ADD     IC,DPB.B+12     ;INCREMENT THE STARTING ADDRESS
 2634  012222  023737  001512  004136            CMP     LC,DPB.B+12     ;EXCEEDING MAXIMUM ?
 2635  012230  002353                            BGE     1$              ;BR IF NOT
 2636  012232  000004                  EXIT11:   SCOPE                   ;LOOP ?
 2637
```

```
2638
2639                                    .SBTTL  *** TIMING TESTS ***
2640

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

        ;*THE TIMING TESTS WILL ENSURE THAT THOSE FUNCTIONS BEING
        ;*TIMED ARE WITHIN THE TOLERANCES SPECIFIED IN THE "RP04
        ;*ENGINEERING SPECIFICATIONS".
        ;*THE SEEK TIMING WILL BE PERFORMED USING EXPLICIT SEEK
        ;*OPERATIONS. AT THE COMPLETION OF EACH OF THE TIMING
        ;*TESTS THE MINIMUM, MAXIMUM AND AVERAGE TIMES WILL BE
        ;*TYPED.

        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:  */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

2658
2659                                    ;;*****************************************************************
2660                                    ;*TEST 12       ROTATIONAL SPEED TIMING TEST
2661
2662                                    ;*      THIS TEST WILL START A SEARCH TO CYLINDER 0, TRACK 0, SECTOR
2663                                    ;*      0.  AS SOON AS THE INTERRUPT OCCURS, THE GO BIT IS SET AGAIN
2664                                    ;*      AND THE OPERATION IS TIMED.  THIS PROCEDURE IS REPEATED 10
2665                                    ;*      TIMES THEN THE AVERAGE TIME IS CALCULATED AND CHECKED TO
2666                                    ;*      ENSURE IT IS WITHIN TOLERANCE:
2667                                    ;*              16.67 MS/REV + OR - 2% IF 60HZ
2668                                    ;*              16.67 MS/REV + OR - 2.5% IF 50HZ.
2669
2670                                    ;;*****************************************************************
2671   012234                           TST12:
2672   012234   000240                          NOP
2673   012236   033737   001450   001234       BIT     @#BITS+(12*2),TSTNMS ;DO THIS TEST?
2674   012244   001002                          BNE     64$             ;YES--BRANCH
2675   012246   000137   012772               JMP     TST13           ;NO--GO TO THE NEXT TEST
2676   012252   012737   000012   001102  64$: MOV     #12,@#STSTNM    ;SET UP TEST NUMBER AND
2677                                                                    ;CLEAR THE ERROR FLAG (SERFLG)
2678   012260   004737   024560               JSR     PC,LOOPRM       ;LOAD THE PARAMETERS FOR THE TEST
2679   012264   012737   012234   001110      MOV     #TST12,@#SLPERR ;SETUP THE ERROR LOOP ADDRESS
2680   012272   013777   001102   166642      MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2681   012300   013737   001506   001204      MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
2682   012306   112737   000031   001115      MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
2683   012314   005737   001244               TST     @#CLKSTA        ;KW11-P CLOCK?
2684   012320   003002                          BGT     1$              ;YES--START TEST
2685   012322   000137   012772               JMP     TST13           ;NO--GO TO NEXT TEST
2686   012326   012737   012326   001106  1$: MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
2687   012334   004037   026640               JSR     R0,@#SRCH00     ;DO A MASSBUS INIT & RECAL
2688   012340   000402                          BR      2$              ;RETURN HERE IF NO ERROR
2689   012342   000137   012770               JMP     EXIT12          ;RETURN HERE IF ERROR
2690   012346   013764   001510   000034  2$: MOV     @#FC,RPCA(R4)   ;FC
2691   012354   013746   001524               MOV     @#FS,-(SP)      ;  S
2692   012360   113766   001516   000001      MOVB    @#FT,1(SP)      ;FT
2693   012366   012664   000006               MOV     (SP)+,RPDA(R4)  ;LOAD FT/FS
```

```
2694  012372  012737  012770  001206          MOV     #EXIT12,$ESCAPE ;;ESCAPE TO EXIT12 ON ERROR
2695  012400  005005                          CLR     R5              ;COUNT UP
2696  012402  012703  002774                  MOV     #174,R3         ;60HZ PARAMETERS
2697  012406  032737  000100  001220          BIT     #SW06,@#C.SWR   ;60 HZ?
2698  012414  001402                          BEQ     TEST12          ;YES--BRANCH
2699  012416  012703  003004                  MOV     #178,R3         ;NO--50 HZ PARAMETERS
2700  012422  012706  001100          TEST12: MOV     #STACK,SP       ;SETUP STACK
2701  012426  012701  000012                  MOV     #10.,R1         ;TIME 10 SEARCHES
2702  012432  004737  027024                  JSR     PC,@#STRTMR     ;INITIALIZE THE TIMERS
2703  012436  012777  012644  166730          MOV     #7$,@PKV        ;SETUP VECTOR IN CASE OF OVERFLOW
2704  012444  012777  027022  022014          MOV     #DOARTI,@RPVEC  ;SETUP RP04/5/6 VECTOR
2705  012452  005077  166724          1$:     CLR     @PKB            ;START COUNTING AT ZERO
2706  012456  012777  000131  166714          MOV     #131,@PKCS      ;INT.EN., COUNT UP AT 100KHZ
2707  012464  012714  000132                  MOV     #SEARCH,(R4)    ;START A SEARCH
2708  012470  000001                          WAIT                    ;WAIT ON INTERRUPT
2709  012472  042777  000101  166700          BIC     #101,@PKCS      ;STOP THE CLOCK
2710  012500  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ;ERROR?
2711  012506  001415                          BEQ     2$              ;NO--BRANCH
2712  012510  104412                          SAVREG                  ;SAVE R0-R5
2713  012512  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
2714  012516  004737  042004                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
2715  012522  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2716  012530  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
2717  012536  104413                          RESREG                  ;RESTORE R0-R5
2718  012540  104017                          ERROR   17
2719  012542  005077  166634          2$:     CLR     @PKB            ;START THE COUNT AT ZERO
2720  012546  012714  000131                  MOV     #SEARCH,(R4)    ;START A SEARCH
2721  012552  012777  000131  166620          MOV     #131,@PKCS      ;START THE CLOCK
2722  012560  000001                          WAIT                    ;WAIT ON INTERRUPT
2723  012562  042777  000101  166610          BIC     #101,@PKCS      ;STOP THE CLOCK
2724  012570  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ;IS "ERR=1"?
2725  012576  001415                          BEQ     3$              ;NO--BRANCH
2726  012600  104412                          SAVREG                  ;SAVE R0-R5
2727  012602  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
2728  012606  004737  042004                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
2729  012612  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2730  012620  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
2731  012626  104413                          RESREG                  ;RESTORE R0-R5
2732  012630  104017                          ERROR   17              ;DISK ERROR OCCURRED
2733  012632  004737  027070          3$:     JSR     PC,@#COUNT      ;UPDATE THE COUNT
2734  012636  005301                          DEC     R1              ;DONE?
2735  012640  003304                          BGT     1$              ;NO--BRANCH
2736  012642  000424                          BR      8$              ;YES--GO TO THE EXIT
2737  012644  042777  000101  166526  7$:     BIC     #101,@PKCS      ;STOP THE CLOCK
2738  012652  005037  177776                  CLR     @#PS            ;DROP THE PRIORITY
2739  012656  012600                          MOV     (SP)+,R0        ;PC OF WAIT+2
2740  012660  005726                          TST     (SP)+           ;POP THE PS FROM THE STACK
2741  012662  104412                          SAVREG                  ;SAVE R0-R5
2742  012664  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
2743  012670  004737  042004                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
2744  012674  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2745  012702  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
2746  012710  104413                          RESREG                  ;RESTORE R0-R5
2747  012712  104020                          ERROR   20              ;CLOCK OVERFLOWED
2748  012714                          8$:
2749  012714  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS INIT.
```

```
012722  013764  004164  000010          MOV     #DTADPB,RPCS2(R4) ;SELECT DRIVE
012726  004737  024044                  JSR     PC,#ST.CLK        ;INITIALIZE THE CLOCK
012732  012777  037346  021524          MOV     #ISR,@#VPVEC      ;RESTORE RH11/RP04/5/6 INT. VECTOR
012740  032737  000100  001220          BIT     #SW06,@#C.SWR     ;60 HZ?
012746  001004                          BNE     9$                ;NO -- BRANCH
012750  004737  027222                  JSR     R0,@#TYPTIM       ;GO TYPE THE TIMES
012754  000774                          T7A                       ;POINTER
012756  000403                          BR      EXIT12            ;GO TO EXIT
012760                          9$:
012760  004037  027222                  JSR     R0,@#TYPTIM       ;GO TYPE THE TIMES
012764  003004                          T7B                       ;POINTER
012766  000004                  EXIT12: SCOPE                     ;LOOP ?

                                ;;*****************************************************************
                                ;*TEST 13        ONE CYLINDER SEEK TIMING TEST

                                ;*      THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
                                ;*      CYLINDER BY ONE UNTIL THE INCREMENT IS GREATER THAN THE
                                ;*      CYLINDER 'LC', THEN REVERSE SEEK TO CYLINDER 'FC'.  THE
                                ;*      TIME TO PERFORM EACH SEEK IS CHECKED TO ENSURE IT DOES NOT
                                ;*      EXCEED THE MAXIMUM TIME PERMITTED FOR A ONE CYLINDER SEEK.
                                ;*      THE TIME MUST BE LESS THAN 10MS.

                                ;;*****************************************************************
012772                  TST13:
012772  000240                  NOP
012774  033737  001452  001234          BIT     @#BITS+(13*2),TSTNMS ;DO THIS TEST?
013002  001002                          BNE     64$               ;YES--BRANCH
013004  000137  013436                  JMP     TST14             ;NO--GO TO THE NEXT TEST
013010  012737  000013  001102  64$:     MOV     #13,@#STSTNM      ;SET UP TEST NUMBER AND
                                                                  ;CLEAR THE ERROR FLAG (SERFLG)
013016  004737  024560                  JSR     PC,LOOPRM         ;LOAD THE PARMETERS FOR THE TEST
013022  012737  012772  001110          MOV     #TST13,@#SLPERR   ;SETUP THE ERROR LOOP ADDRESS
013030  013777  001102  166104          MOV     STSTNM,@DISPLAY   ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
013036  013737  001506  001204          MOV     @#RPT,$TIMES      ;GET THE ITERATION COUNT
013044  112737  000031  001115          MOVB    #25.,$ERMAX       ;MAX ERRORS ALLOWED FOR TEST
013052  005737  001244                  TST     @#CLKSTA          ;KW11-P CLOCK?
013056  003002                          BGT     1$                ;YES--START TEST
013060  000137  013436                  JMP     TST14             ;NO--GO TO NEXT TEST
013064  012737  013064  001106  1$:     MOV     #1$,SLPADR        ;SETUP THE LOOP ADDRESS
013072  004037  026640                  JSR     R0,@#SRCH00       ;DO A MASSBUS INIT. AND RECAL
013076  000402                          BR      2$                ;NO ERROR RETURN
013100  000137  013434                  JMP     EXIT13            ;ERROR RETURN--SCOPE LOOP CALL
013104  012703  001014          2$:     MOV     #T10,R3           ;PARAMETER POINTER
013110  012737  013434  001206          MOV     #EXIT13,$ESCAPE   ;ESCAPE TO EXIT13 ON ERROR
013116  012706  001100          TEST13: MOV     #STACK,SP         ;SETUP STACK
013122  013737  001510  004176          MOV     FC,@#DTADPB+12    ;START WITH BEGINNING CYLINDER
013130  005237  004176                  INC     DTADPB+12         ;INCREMENT THE BEGINNING CYLINDER
013134  005005                          CLR     R5                ;SET THE UP/DOWN SWITCH TO UP
013136  004737  027024                  JSR     PC,@#STRTMR       ;INITIALIZE THE TIMERS
013142  012777  013330  166224          MOV     #7$,@PKV          ;SETUP INCASE OF OVERFLOW
013150  012777  027022  021310          MOV     #DORTI,@RPVEC     ;SET RP04/5/6 VECTOR
013156  005077  166220          1$:     CLR     @PKB              ;START THE COUNTER AT ZERO
013162  013764  004176  000034          MOV     @#DTADPB+12,RPCA(R4) ;LOAD DESIRED CYLINDER
013170  012714  000105                  MOV     #SEEK,(R4)        ;START A SEEK
```

```
2806  013174  012777  000131  166176          MOV     #131,@PKCS      ;START THE CLOCK
2807  013202  000001                          WAIT                    ;WAIT ON INTERRUPT
2808  013204  042777  000101  166166          BIC     #101,@PKCS      ;STOP THE CLOCK
2809  013212  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ;ANY DISK ERRORS?
2810  013220  001415                          BEQ     2$              ;NO--BRANCH
2811  013222  104412                          SAVREG                  ;SAVE R0-R5
2812  013224  012703  004164                  MOV     #DTADPB,R2      ;DPB POINTER
2813  013230  004737  042004                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
2814  013234  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2815  013242  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
2816  013250  104413                          RESREG                  ;RESTORE R0-R5
2817  013252  104017                          ERROR   17              ;REPORT THE ERROR
2818  013254  004737  027070          2$:     JSR     PC,@#COUNT      ;COUNT THIS SEEKS TIME
2819  013260  004737  026416                  JSR     PC,@#TWOMS      ;STALL FOR 2 MILLISECONDS
2820  013264  005705                          TST     R5              ;UP OR DOWN?
2821  013266  001011                          BNE     4$              ;DOWN--BRANCH
2822  013270  005237  004176          3$:     INC     @#DTADPB+12     ;MOVE TO NEXT CYLINDER
2823  013274  023737  004176  001512          CMP     @#DTADPB+12,LC ;OUT OF CYLINDERS?
2824  013302  002725                          BLT     1$              ;NO--GO DO THE NEXT SEEK
2825  013304  012705  177777                  MOV     #-1,R5          ;SET UP/DOWN SWITCH TO DOWN
2826  013310  000722                          BR      1$              ;GO DO THE NEXT SEEK
2827  013312  005337  004176          4$:     DEC     @#DTADPB+12     ;MOVE TO NEXT CYLINDER
2828  013316  023727  004176  000000          CMP     @#DTADPB+12,#0  ;OUT OF CYLINDERS?
2829  013324  003314                          BGT     1$              ;NO--GO DO THE NEXT SEEK
2830  013326  000424                          BR      8$              ;GO TO THE EXIT
2831  013330  042777  000101  166042  7$:     BIC     #101,@PKCS      ;STOP THE CLOCK
2832  013336  005037  177776                  CLR     @#PS            ;DROP THE PRIORITY
2833  013342  012600                          MOV     (SP)+,R0        ;PC OF WAIT+2
2834  013344  005726                          TST     (SP)+           ;POP THE PS FROM THE STACK
2835  013346  104412                          SAVREG                  ;SAVE R0-R5
2836  013350  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
2837  013354  004737  042004                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
2838  013360  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2839  013366  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
2840  013374  104413                          RESREG                  ;RESTORE R0-R5
2841  013376  104020                          ERROR   20              ;REPORT CLOCK OVERFLOW
2842  013400                          8$:
2843  013400  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS INIT.
2844  013406  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
2845  013414  004737  024044                  JSR     PC,@#ST.CLK     ;INITIALIZE THE CLOCK
2846  013420  012777  037346  021040          MOV     #ISR,@#RPVEC    ;RESTORE RH11/RP04/5/6 INT. VECTOR
2847  013426  004037  027222                  JSR     R0,@#TYPTIM     ;GO TYPE THE TIMES
2848  013432  003014                          T10                     ;POINTER
2849  013434  000004                  EXIT13: SCOPE                   ;LOOP ?
2850
2851
2852                                  ;;*****************************************************************
2853                                  ;*TEST 14       ACCESS TIME MEASURMENT TEST
2854
2855                                  ;*      THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO
2856                                  ;*      CYLINDER 'LC', THEN A REVERSEK FROM CYLINDER 'LC' TO
2857                                  ;*      CYLINDER 0.  BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY
2858                                  ;*      ARE WITHIN THE TOLERANCE ALLOWED FOR THE ACCESS TIME MEASURMENT.
2859                                  ;*      THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL
2860                                  ;*      OF 256 SEEKS).  THE AVERAGE ACCESS TIME MUST BE LESS THAN 30 MS.
2861                                  ;*      CYLINDER 'LC' DEFAULTS TO 136 (10) FOR AN RP04/5 OR TO 255 (10)
```

```
                                        ;*      FOR AN RP06.
                                        ;;*****************************************************************
013436                                  TST14:
013436  000240                          NOP
013440  033737  001454  001234          BIT     @#BITS+<14*2>,TSTNMS ;DO THIS TEST?
013446  001002                          BNE     64$                 ;YES--BRANCH
013450  000137  014154                  JMP     TST15               ;NO--GO TO THE NEXT TEST
013454  012737  000014  001142  64$:    MOV     #14,@#STSTNM        ;SET UP TEST NUMBER AND
                                                                    ;CLEAR THE ERROR FLAG (SERFLG)
013462  004737  024560                  JSR     PC,LOOPRM           ;LOAD THE PARAMETERS FOR THE TEST
013466  012737  013436  001110          MOV     #TST14,@#SLPERR     ;SETUP THE ERROR LOOP ADDRESS
013474  013777  001102  165440          MOV     $TSTNM,@DISPLAY     ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
013502  013737  001506  001204          MOV     @#RPT,$TIMES        ;GET THE ITERATION COUNT
013510  112737  000031  001115          MOVB    #25.,$ERMAX         ;MAX ERRORS ALLOWED FOR TEST
013516  005737  001244                  TST     @#CLKSTA            ;KW11-P CLOCK?
013522  003002                          BGT     1$                  ;YES--START TEST
013524  000137  014154                  JMP     TST15               ;NO--GO TO NEXT TEST
013530  012737  013530  001106  1$:     MOV     #1$,$LPADR          ;SET THE LOOP ADDRESS
013536  004037  026640                  JSR     R0,@#SRCH00         ;DO A MASSBUS INIT & RECAL
013542  000402                          BR      2$                  ;RETURN HERE IF NO ERROR
013544  000137  014152                  JMP     EXIT14              ;RETURN HERE ON ERROR
013550  012703  003024          2$:     MOV     #T11,R3             ;PARAMETER POINTER
013554  012737  014152  001206          MOV     #EXIT14,$ESCAPE     ;ESCAPE TO EXIT14 ON ERROR
013562  012706  001100          TEST14: MOV     #STACK,SP           ;SETUP STACK
013566  012701  000200                  MOV     #128.,R1            ;REPEAT "0-136-0" 128 TIMES
013572  004737  027024                  JSR     PC,@#STRTMR         ;INIT. THE COUNTERS
013576  012777  014046  165570          MOV     #7$,@PKV            ;SET UP VECTOR IN CASE OF OVERFLOW
013604  012777  027022  020654          MOV     #UORTI,@RPVEC       ;SETUP RP04/5/6 VECTOR
013612  005077  165564          1$:     CLR     @PKB                ;START COUNT AT ZERO
013616  013764  001512  000034          MOV     LC,RPCA(R4)         ;'MIDDLE' CYLINDER
013624  012764  000105  000000          MOV     #SEEK,RPCS1(R4)     ;START A SEEK
013632  012777  000131  165540          MOV     #131,@PKCS          ;START THE CLOCK
013640  000001                          WAIT                        ;WAIT ON INTERRUPT
013642  042777  000101  165530          BIC     #101,@PKCS          ;STOP CLOCK
013650  032764  040000  000012          BIT     #BIT14,RPDS1(R4)    ;ERR=1?
013656  001415                          BEQ     2$                  ;NO--BRANCH
013660  104412                          SAVREG                      ;SAVE R0-R5
013662  012702  004164                  MOV     #DTADPB,R2          ;DPB POINTER
013666  004737  042004                  JSR     PC,@#SVRH11         ;SAVE ALL THE RH11/RP04 REGISTERS
013672  012764  000040  000010          MOV     #BIT05,RPCS2(R4)    ;MASSBUS CLEAR
013700  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4)  ;SELECT DRIVE
013706  104413                          RESREG                      ;RESTORE R0-R5
013710  104017                          ERROR   17
013712  005005                  2$:     CLR     R5                  ;SET UP/DOWN SWITCH TO UP
013714  004737  027070                  JSR     PC,@#COUNT          ;UPDATE THE COUNT
013720  004737  026416                  JSR     PC,@#TWOMS          ;STALL FOR 2 MILLISECONDS
013724  005077  165452                  CLR     @PKB                ;START THE COUNT AT ZERO
013730  013764  001510  000034          MOV     FC,RPCA(R4)         ;BEGINNING CYLINDER
013736  012764  000105  000000          MOV     #SEEK,RPCS1(R4)     ;START A SEEK
013744  012777  000131  165426          MOV     #131,@PKCS          ;START THE CLOCK
013752  000001                          WAIT                        ;WAIT ON INTERRUPT
013754  042777  000101  165416          BIC     #101,@PKCS          ;STOP THE CLOCK
013762  032764  040000  000012          BIT     #BIT14,RPDS1(R4)    ;ERR=1?
013770  001415                          BEQ     3$                  ;NO--BRANCH
013772  104412                          SAVREG                      ;SAVE R0-R5
```

```
2918  013774  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
2919  014000  004737  042004                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RPO4 REGISTERS
2920  014004  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2921  014012  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
2922  014020  104413                          RESREG                  ;RESTORE R0-R5
2923  014022  104017                          ERROR   17
2924  014024  012705  177777          3$:     MOV     #-1,R5          ;SET UP/DOWN SWITCH TO DOWN
2925  014030  004737  027070                  JSR     PC,@#COUNT      ;UPDATE THE COUNT
2926  014034  004737  026416                  JSR     PC,@#TWOMS      ;STALL FOR 2 MILLISECONDS
2927  014040  005301                          DEC     R1              ;DONE?
2928  014042  003263                          BGT     1$              ;NO--BRANCH
2929  014044  000424                          BR      8$              ;YES--EXIT
2930  014046  042777  000101  165324  7$:     BIC     #101,@PKCS      ;STOP THE CLOCK
2931  014054  005037  177776                  CLR     @#PS            ;DROP THE PRIORITY
2932  014060  012600                          MOV     (SP)+,R0        ;PC OF WAIT+2
2933  014062  005726                          TST     (SP)+           ;POP THE PS FROM THE STACK
2934  014064  104412                          SAVREG                  ;SAVE R0-R5
2935  014066  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
2936  014072  004737  042004                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RPO4 REGISTERS
2937  014076  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
2938  014104  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
2939  014112  104413                          RESREG                  ;RESTORE R0-R5
2940  014114  104020                          ERROR   20              ;CLOCK OVERFLOWED
2941  014116                          8$:
2942  014116  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS INIT.
2943  014124  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
2944  014132  004737  024044                  JSR     PC,@#ST.CLK     ;INITIALIZE THE CLOCK
2945  014136  012777  037346  020322          MOV     #ISR,@RPVEC     ;RESTORE RH11/RPO4/5/6 INT. VECTOR
2946  014144  004037  027222                  JSR     R0,@#TYPTIM     ;GO TYPE THE TIMES
2947  014150  003024                          T11                     ;POINTER
2948  014152  000004                  EXIT14: SCOPE                   ;LOOP ?
2949
2950                                  ;;**************************************************************
2951                                  ;*TEST 15       MAXIMUM SEEK TIMING TEST
2952
2953                                  ;*      THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 0 TO
2954                                  ;*      CYLINDER 'LC', THEN A REVERSE SEEK FROM CYLINDER 'LC' TO
2955                                  ;*      CYLINDER 0.  BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE
2956                                  ;*      THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE MAXIMUM SEEK
2957                                  ;*      TIME.  THIS SEQUENCE IS REPEATED 128 TIMES (FOR
2958                                  ;*      A TOTAL OF 256 SEEKS).  THE MAXIMUM SEEK TIME MUST BE LESS THAN
2959                                  ;*      54 MS.  'LC' DEFAULTS TO 410 (10) FOR RPO4/5'S AND TO 814 (10)
2960                                  ;*      FOR RPO6'S.
2961
2962                                  ;;**************************************************************
2963  014154                          TST15:
2964  014154  000240                          NOP
2965  014156  033737  001456  001234          BIT     @#BITS+(15*2),TSTNMS ;DO THIS TEST?
2966  014164  001002                          BNE     64$             ;YES--BRANCH
2967  014166  000137  014672                  JMP     TST16           ;NO--GO TO THE NEXT TEST
2968  014172  012737  000015  001102  64$:     MOV     #15,@#STSTNM    ;SET UP TEST NUMBER AND
2969                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
2970  014200  004737  024560                  JSR     PC,LOOPRM       ;LOAD THE PARMETERS FOR THE TEST
2971  014204  012737  014154  001110          MOV     #TST15,@#$LPERR ;SETUP THE ERROR LOOP ADDRESS
2972  014212  013777  001102  164722          MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2973  014220  013737  001506  001204          MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
```

```
2974  014226  112737  000031  001115          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
2975  014234  005737  001244                  TST     @#CLKSTA        ;KW11-P CLOCK
2976  014240  003002                          BGT     1$              ;YES--START TEST
2977  014242  000137  014672                  JMP     TST16           ;NO--GO TO NEXT TEST
2978  014246  012737  014246  001106  1$:     MOV     #1$,$LPADR      ;SETUP THE LOOP ADDRESS
2979  014254  004037  026640                  JSR     R0,@#SRCH00     ;DO A MASSBUS INIT & RECAL
2980  014260  000402                          BR      2$              ;RETURN HERE IF NO ERROR
2981  014262  000137  014670                  JMP     EXIT15          ;RETURN HERE ON ERROR
2982  014266  012703  003034          2$:     MOV     #T12,R3         ;PARAMETER POINTER
2983  014272  012737  014670  001206          MOV     #EXIT15,$ESCAPE ;;ESCAPE TO EXIT15 ON ERROR
2984  014300  012706  001100          TEST15: MOV     #STACK,SP       ;SETUP STACK
2985  014304  012701  000200                  MOV     #128.,R1        ;REPEAT "0-'LC'-0" 128 TIMES
2986  014310  004737  027024                  JSR     PC,@#STRTMR     ;INIT. THE TIMERS
2987  014314  012777  014564  165052          MOV     #7$,@PKV        ;SETUP VECTOR IN CASE OF OVERFLOW
2988  014322  012777  027022  020136          MOV     #DORTI,@RPVEC   ;SETUP RP04/5/6 VECTOR
2989  014330  005077  165046          1$:     CLR     @PKB            ;START COUNTING FROM ZERO
2990  014334  013764  001512  000034          MOV     LC,RPCA(R4)     ;MAXIMUM CYLINDER
2991  014342  012764  000105  000000          MOV     #SEEK,RPCS1(R4) ;START A SEEK
2992  014350  012777  000131  165022          MOV     #131,@PKCS      ;START THE CLOCK
2993  014356  000001                          WAIT                    ;WAIT ON INTERRUPT
2994  014360  042777  000101  165012          BIC     #101,@PKCS      ;STOP THE CLOCK
2995  014366  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ;ERR=1?
2996  014374  001415                          BEQ     2$              ;NO--BRANCH
2997  014376  104412                          SAVREG                  ;SAVE R0-R5
2998  014400  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
2999  014404  004737  042004                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
3000  014410  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
3001  014416  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
3002  014424  104413                          RESREG                  ;RESTORE R0-R5
3003  014426  104017                          ERROR   17
3004  014430  005005                  2$:     CLR     R5              ;SET THE UP/DOWN SWITCH TO UP
3005  014432  004737  027070                  JSR     PC,@#COUNT      ;UP THE COUNT
3006  014436  004737  026416                  JSR     PC,@#TWOMS      ;STALL FOR 2 MILLISECONDS
3007  014442  005077  164734                  CLR     @PKB            ;START COUNT AT ZERO
3008  014446  013764  001510  000034          MOV     FC,RPCA(R4)     ;BEGINNING CYLINDER
3009  014454  012764  000105  000000          MOV     #SEEK,RPCS1(R4) ;START A SEEK
3010  014462  012777  000131  164710          MOV     #131,@PKCS      ;START THE CLOCK
3011  014470  000001                          WAIT                    ;WAIT ON INTERRUPT
3012  014472  042777  000101  164700          BIC     #101,@PKCS      ;STOP THE CLOCK
3013  014500  032764  040000  000012          BIT     #BIT14,RPDS1(R4) ;"ERR"=1?
3014  014506  001415                          BEQ     3$              ;NO--BRANCH
3015  014510  104412                          SAVREG                  ;SAVE R0-R5
3016  014512  012702  004164                  MOV     #DTADPB,R2      ;DPB POINTER
3017  014516  004737  042004                  JSR     PC,@#SVRH11     ;SAVE ALL THE RH11/RP04 REGISTERS
3018  014522  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS CLEAR
3019  014530  013764  004164  000010          MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
3020  014536  104413                          RESREG                  ;RESTORE R0-R5
3021  014540  104017                          ERROR   17              ;REPORT THE ERROR
3022  014542  012705  177777          3$:     MOV     #-1,R5          ;SET THE UP/DOWN SWITCH TO DOWN
3023  014546  004737  027070                  JSR     PC,@#COUNT      ;UPDATE THE COUNT
3024  014552  004737  026416                  JSR     PC,@#TWOMS      ;STALL FOR 2 MILLISECONDS
3025  014556  005301                          DEC     R1              ;DONE?
3026  014560  003363                          BGT     1$              ;NO--BRANCH
3027  014562  000424                          BR      8$              ;YES--EXIT
3028  014564  042777  000101  164606  7$:     BIC     #101,@PKCS      ;STOP THE CLOCK
3029  014572  005037  177776                  CLR     @#PS            ;DROP THE PRIORITY
```

```
 3030  014576  012600                        MOV     (SP)+,R0          ;PC OF WAIT+2
 3031  014600  005726                        TST     (SP)+             ;POP THE PS FROM THE STACK
 3032  014602  104412                        SAVREG                    ;SAVE R0-R5
 3033  014604  012702  004164                MOV     #DTADPB,R2        ;DPB POINTER
 3034  014610  004737  042004                JSR     PC,@#SVRH11       ;SAVE ALL THE RH11/RP04 REGISTERS
 3035  014614  012764  000040  000010        MOV     #BIT05,RPCS2(R4)  ;MASSBUS CLEAR
 3036  014622  013764  004164  000010        MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
 3037  014630  104413                        RESREG                    ;RESTORE R0-R5
 3038  014632  104020                        ERROR   20                ;CLOCK OVERFLOWED
 3039  014634                          BS:
 3040  014634  012764  000040  000010        MOV     #BIT05,RPCS2(R4)  ;MASSBUS INIT.
 3041  014642  013764  004164  000010        MOV     @#DTADPB,RPCS2(R4) ;SELECT DRIVE
 3042  014650  004737  024044                JSR     PC,@#ST.CLK       ;INITIALIZE THE CLOCK
 3043  014654  012777  037346  017604        MOV     #ISR,@RPVEC       ;RESTORE RH11/RP04/5/6 INT. VECTOR
 3044  014662  004037  027222                JSR     R0,@#TYPTIM       ;GO TYPE THE TIMES
 3045  014666  003034                        T12                       ;POINTER
 3046  014670  000004                  EXIT15: SCOPE                     ;LOOP ?
```

```
3047
3048                                    .SBTTL  *** ADDRESSING TESTS ***
3049

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\./*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

        ;*THE ADDRESSING TESTS ENSURES PROPER OPERATION OF THE TRACK
        ;*AND SECTOR ADDRESS CIRCUITRY.  BOTH ADDRESSING TESTS
        ;*WILL BE PERFORMED ON CYLINDER FC.

        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

3063
3064                                    ;;*****************************************************************
3065                                    ;*TEST 16       SECTOR ADDRESSING TEST
3066
3067                                    ;*      THIS TEST WRITES DATA INTO ALL SECTORS OF TRACK "FT".  THE
3068                                    ;*      DATA WILL BE 256 WORDS OF THE SECTOR ADDRESS OF THE SECTOR
3069                                    ;*      BEING WRITTEN.  A WRITE CHECK IS PERFORMED, THE BUFFER IS
3070                                    ;*      CLEARED (TO 177400) AND THE DATA IS READ AND COMPARED.  THEN SECTOR 0
3071                                    ;*      IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED.  THEN
3072                                    ;*      SECTOR 1 IS REWRITTEN AND SECTORS 0 - 21 ARE WRITE CHECKED.
3073                                    ;*      THIS REWRITE AND WRITE CHECK PROCEDURE IS CONTINUED UP THROUGH
3074                                    ;*      REWRITE SECTOR 21 AND WRITE CHECK SECTORS 0-21.
3075
3076                                    ;;*****************************************************************
3077  014672                            TST16:
3078  014672  000240                           NOP
3079  014674  033737  001460  001234           BIT     @#BIT^ .16*2),TSTNMS ;DO THIS TEST?
3080  014702  001002                           BNE     64$             ;YES--BRANCH
3081  014704  000137  015270                   JMP     TST17           ;NO--GO TO THE NEXT TEST
3082  014710  012737  000016  001102  64$:    MOV     #16,@#STSTNM    ;SET UP TEST NUMBER AND
3083                                                                    ;CLEAR THE ERROR FLAG (SERFLG)
3084  014716  004737  024560                   JSR     PC,LOOPRM       ;LOAD THE PARMETERS FOR THE TEST
3085  014722  012737  014766  001110           MOV     #TEST16,@#SLPERR  ;SETUP THE LOOP ON ERROR ADDRESS
3086  014730  013777  001102  164204           MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3087  014736  013737  001506  001204           MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
3088  014744  113737  001364  001115           MOVB    ERR.CT,SERMAX   ;MAX ERRORS ALLOWED FOR TEST
3089  014752  012737  015266  001252           MOV     #EXIT16,@#BYPASS
3090  014760  012737  014766  001106           MOV     #TEST16,$LPADR  ;SETUP THE LOOP ADDRESS
3091  014766  012706  001100  TEST16: MOV      #STACK,SP       ;SET THE STACK POINTER
3092  014772  004737  027654                   JSR     PC,@#FILBUF     ;FILL THE BUFFER WITH SECTOR ADDRESSES
3093  014776  013737  001510  004176           MOV     @#FC,@#DTADPB+12 ;CYLINDER
3094  015004  113737  001516  004175           MOVB    @#FT,@#DTADPB+11 ;TRACK
3095  015012  105037  004174                   CLRB    @#DTADPB+10     ;SECTOR
3096  015016  013737  001352  004170           MOV     TRCKWC,@#DTADPB+4 ;WORD COUNT
3097  015024  012737  047662  004172           MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
3098  015032  012737  015032  001110           MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3099  015040  012706  001100                   MOV     #STACK,SP       ;LOAD THE STACK POINTER
3100  015044  012737  000161  004166           MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
3101  015052  004037  025474                   JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3102  015056  012737  000151  004166           MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
```

```
3103  015064  012737  015064  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3104  015072  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3105  015076  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3106  015102  012737  015102  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3107  015110  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3108  015114  004037  027712                  JSR     R0,@#CLRBUF     ;CLEAR BUFFER
3109  015120  012737  000171  004166          MOV     #READ,@#DTADPB+2 ;COMMAND = READ
3110  015126  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3111  015132  004037  027760                  JSR     R0,@#CKSCTR     ;CHECK THE SECTOR DATA READ
3112  015136  012700  047662                  MOV     #BUFFER,R0      ;BUFFER ADDRESS
3113  015142  005001                          CLR     R1              ;FIRST SECTOR
3114  015144  012737  015144  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3115  015152  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3116  015156  012737  000161  004166  1$:     MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
3117  015164  012737  177400  004170          MOV     #SCTRWC,@#DTADPB+4 ;WORD COUNT
3118  015172  010037  004172                  MOV     R0,@#DTADPB+6   ;BUFFER ADDRESS
3119  015176  110137  004174                  MOVB    R1,@#DTADPB+10  ;SECTOR
3120  015202  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3121  015206  012737  015206  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3122  015214  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3123  015220  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
3124  015226  013737  001352  004170          MOV     TRCKWC,@#DTADPB+4 ;WORD COUNT
3125  015234  012737  047662  004172          MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
3126  015242  105037  004174                  CLRB    @#DTADPB+10     ;SECTOR
3127  015246  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3128  015252  062700  001000                  ADD     #512.,R0        ;MOVE TO NEXT SECTOR
3129  015256  005201                          INC     R1
3130  015260  023701  001630                  CMP     PRMLMT+22,R1    ;DONE?
3131  015264  103334                          BHIS    1$              ;NO--BRANCH
3132  015266  000004                  EXIT16: SCOPE                   ;LOOP ?
3133
3134                                  ;;**************************************************************
3135                                  ;*TEST 17       TRACK ADDRESSING TEST
3136
3137                                  ;*      THIS TEST WILL WRITE DATA IN THE FORM OF TRACK ADDRESSES
3138                                  ;*      IN CYLINDER "FC" SECTOR "FS" OF EVERY TRACK WITH EACH TRACK
3139                                  ;*      GETTING ITS OWN TRACK ADDRESS.
3140                                  ;*      A WRITE CHECK IS THEN PERFORMED ON EACH TRACK TO ENSURE
3141                                  ;*      THE DATA IS VALID.  THEN TRACK 0 IS REWRITTEN AND TRACK 1
3142                                  ;*      THROUGH TRACK 18 IS WRITE CHECKED.  THEN TRACK 1 IS
3143                                  ;*      REWRITTEN AND TRACK 2 THROUGH TRACK 18 IS WRITE CHECKED.
3144                                  ;*      THIS PROCEDURE IS CONTINUED UP THROUGH REWRITING TRACK 17
3145                                  ;*      AND WRITE CHECKING TRACK 18.
3146
3147                                  ;;**************************************************************
3148  015270                          TST17:
3149  015270  000240                          NOP
3150  015272  033737  001462  001234          BIT     @#BITS+<17*2>,TSTNMS ;DO THIS TEST?
3151  015300  001002                          BNE     64$             ;YES--BRANCH
3152  015302  000137  015710                  JMP     TST20           ;NO--GO TO THE NEXT TEST
3153  015306  012737  000017  001102  64$:     MOV     #17,@#STSTNM    ;SET UP TEST NUMBER AND
3154                                                                  ;CLEAR THE ERROR FLAG (SERFLG)
3155  015314  004737  024560                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
3156  015320  012737  015364  001110          MOV     #TEST17,@#LPERR ;SETUP THE LOOP ON ERROR ADDRESS
3157  015326  013777  001102  163606          MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3158  015334  013737  001506  001204          MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
```

```
3159  015342  113737  001364  001115          MOVB    ERR.CT,$ERMAX   ;MAX ERRORS ALLOWED FOR TEST
3160  015350  012737  015706  001252          MOV     #EXIT17,@#BYPASS
3161  015356  012737  015364  001106          MOV     #TEST17,$LPADR  ;SETUP THE LOOP ADDRESS
3162  015364  012706  001100          TEST17: MOV     #STACK,SP       ;SET THE STACK POINTER
3163  015370  004737  027654                  JSR     PC,@#FILBUF     ;FILL THE BUFFER WITH TRACK ADDRESS
3164  015374  012737  000161  004166          MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
3165  015402  013737  001510  004176          MOV     @#FC,@#DTADPB+12 ;CYLINDER
3166  015410  113737  001524  004174          MOVB    @#FS,@#DTADPB+10 ;SECTOR
3167  015416  012737  177400  004170          MOV     #SCTRWC,@#DTADPB+4 ;WORD COUNT
3168  015424  012737  047662  004172          MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
3169  015432  005000                          CLR     R0              ;TRACK=0
3170  015434  012737  015434  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3171  015442  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3172  015446  110037  004175          1$:     MOVB    R0,@#DTADPB+11  ;TRACK ADDRESS
3173  015452  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3174  015456  062737  001000  004172          ADD     #256.*2.,@#DTADPB+6 ;UPDATE BUFFER ADDRESS
3175  015464  005200                          INC     R0              ;UPDATE TRACK NUMBER
3176  015466  022700  000023                  CMP     #19.,R0         ;OUT OF TRACKS?
3177  015472  003365                          BGT     1$              ;NO--BRANCH
3178  015474  012737  047662  004172          MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
3179  015502  005000                          CLR     R0
3180  015504  012737  015504  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3181  015512  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3182  015516  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK
3183  015524  110037  004175          2$:     MOVB    R0,@#DTADPB+11  ;TRACK ADDRESS
3184  015530  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3185  015534  062737  001000  004172          ADD     #256.*2.,@#DTADPB+6 ;UPDATE BUFFER ADDRESS
3186  015542  005200                          INC     R0              ;UPDATE TRACK NUMBER
3187  015544  022700  000023                  CMP     #19.,R0 ;OUT OF TRACKS?
3188  015550  003365                          BGT     2$              ;NO--BRANCH
3189  015552  005000                          CLR     R0              ;FIRST TRACK ADDRESS
3190  015554  110037  004175          3$:     MOVB    R0,@#DTADPB+11  ;TRACK
3191  015560  010001                          MOV     R0,R1           ;FORM BUFFER ADDRESS
3192  015562  012737  047662  004172          MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
3193  015570  005301                  4$:     DEC     R1
3194  015572  002411                          BLT     5$
3195  015574  062737  001000  004172          ADD     #256.*2.,@#DTADPB+6
3196  015602  000772                          BR      4$
3197  015604  012737  015604  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3198  015612  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3199  015616  012737  000161  004166  5$:     MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
3200  015624  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3201  015630  062737  001000  004172  6$:     ADD     #256.*2.,@#DTADPB+6 ;UPDATE BUFFER ADDRESS
3202  015636  105237  004175                  INCB    @#DTADPB+11     ;MOVE TO NEXT TRACK
3203  015642  012737  015642  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3204  015650  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3205  015654  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
3206  015662  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3207  015666  122737  000022  004175          CMPB    #18.,@#DTADPB+11 ;OUT OF TRACKS?
3208  015674  003355                          BGT     6$              ;NO--BRANCH
3209  015676  005200                          INC     R0              ;NEXT TRACK TO WRITE
3210  015700  022700  000022                  CMP     #18.,R0         ;OUT OF TRACKS?
3211  015704  003323                          BGT     3$              ;NO--BRANCH
3212  015706  000004                  EXIT17: SCOPE
```

```
                        .SBTTL  *** DATA TEST ***

                        ;;*****************************************************************
                        ;*TEST 20        DATA TEST

                        ;*      THIS TEST PERFORMS DATA STORAGE AND RETRIEVAL ON CYLINDERS
                        ;*      "FC" THROUGH "LC" BY THE INCREMENT "IC" USING THE DATA PATTERNS
                        ;*      SPECIFIED. THE FOLLOWING SEQUENCE OCCURS FOR EACH CYLINDER:
                        ;*      1.      SET "NT" TO "FT" THEN REPEAT 2-4 UNTIL "NT" > "LT"
                        ;*      2.      WRITE THEN WRITE CHECK "FS" THROUGH "LS" OF TRACK "NT"
                        ;*      3.      READ THEN SOFTWARE COMPARE "FS" THROUGH "LS" OF TRACK "NT"
                        ;*      4.      INCREMENT "NT" BY "IT"
                        ;*      5.      REPEAT STEPS 1-4 FOR EACH DATA PATTERN
                        ;*      6.      REPEAT STEPS 1-5 FOR "FC" THROUGH "LC" ADVANCING BY "IC"
                        ;*
                        ;*      IF A WRITE CHECK ERROR OCCURS THE ERROR IS REPORTED AND
                        ;*      THE TRACK IN ERROR IS REWRITTEN AND CHECKED. THIS CHECK IS
                        ;*      ACCOMPLISHED BY PERFORMING TWO(2) SUCCESSIVE ERROR FREE
                        ;*      WRITE CHECKS. IF THE CHECK FAILS THE ERROR IS REPORTED AS
                        ;*      FATAL AND NO READ OCCURS.
                        ;*      FS DEFAULTS TO 1 AND LS DEFAULTS TO 0
                        ;*      PAT DEFAULTS TO 177777 (ALL POSSIBLE PATTERNS)
                        ;*      THE POSSIBLE PATTERNS ARE:
                        ;*
                        ;*      PAT 0   PAT 1   PAT 2   PAT 3   PAT 4   PAT 5   PAT 6   PAT 7
                        ;*      ------  ------  ------  ------  ------  ------  ------  ------
                        ;*      165555  000001  177776  000000  000000  052525  007417  026455
                        ;*      133333  000003  177774  000000  010421  052525  007417  026455
                        ;*      165555  000007  177770  000000  021042  052525  007417  026455
                        ;*      133333  000017  177760  177777  031463  125252  170360  151322
                        ;*      165555  000037  177740  177777  042104  125252  170360  151322
                        ;*      133333  000077  177700  177777  052525  125252  170360  151322
                        ;*      165555  000177  177600  000000  063146  052525  007417  026455
                        ;*      133333  000377  177400  000000  073567  052525  007417  026455
                        ;*      165555  000777  177000  177777  104210  125252  170360  151322
                        ;*      133333  001777  176000  177777  114631  125252  170360  151322
                        ;*      165555  003777  174000  000000  125252  052525  007417  026455
                        ;*      133333  007777  170000  177777  135673  125252  170360  151322
                        ;*      165555  017777  160000  000000  146314  052525  007417  026455
                        ;*      133333  037777  140000  177777  156735  125252  170360  151322
                        ;*      165555  077777  100000  000000  167356  052525  007417  026455
                        ;*      133333  177777  000000  177777  177777  125252  170360  151322
                        ;*
                        ;*      PAT 8   PAT 9   PAT 10  PAT 11  PAT 12  PAT 13  PAT 14  PAT 15
                        ;*      ------  ------  ------  ------  ------  ------  ------  ------
                        ;*      165555  000001  177776  172666  077777  153333  000000  177777
                        ;*      133333  000002  177775  155555  137777  066667  177777  000000
                        ;*      165555  000004  177773  172666  157777  153333  177777  000000
                        ;*      133333  000010  177767  155555  167777  066667  177777  000000
                        ;*      165555  000020  177757  172666  173777  153333  177777  000000
                        ;*      133333  000040  177737  155555  175777  066667  177777  000000
                        ;*      165555  000100  177677  172666  176777  153333  177777  000000
                        ;*      133333  000200  177577  155555  177377  066667  177777  000000
                        ;*      165555  000400  177377  172666  177577  153333  177777  000000
                        ;*      133333  001000  176777  155555  177677  066667  177777  000000
                        ;*      165555  002000  175777  172666  177737  153333  177777  000000
```

```
3269                                    ;*      133333  004000  173777  155555  177757  066667  177777  000000
3270                                    ;*      165555  010000  167777  172666  177767  153333  177777  000000
3271                                    ;*      133333  020000  157777  155555  177773  066667  177777  000000
3272                                    ;*      165555  040000  137777  172666  177775  153333  177777  000000
3273                                    ;*      133333  100000  077777  155555  177776  066667  177777  000000
3274                                    ;*
3275
3276                                    ;;**********************************************************************
3277  015710                            TST20:
3278  015710  000240                            NOP
3279  015712  033737  001424  001236            BIT     BITS+<20*2-40>,TSTNMS+2  ;DO THIS TEST ?
3280  015720  001002                            BNE     64$                     ;YES--BRANCH
3281  015722  000137  016430                    JMP     TST21                   ;NO--GO TO THE NEXT TEST
3282  015726  012737  000020  001102  64$:     MOV     #20,@#$ISTNM            ;SET UP TEST NUMBER AND
3283                                                                            ;CLEAR THE ERROR FLAG ($ERFLG)
3284  015734  004737  024560                    JSR     PC,LODPRM               ;LOAD THE PARMETERS FOR THE TEST
3285  015740  012737  016112  001110            MOV     #TEST20,@#$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
3286  015746  013777  001102  163166            MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3287  015754  013737  001506  001204            MOV     @#RPT,$TIMES    ;GET THE ITERATION COUNT
3288  015762  113737  001364  001115            MOVB    ERR.CT,$ERMAX   ;MAX ERRORS ALLOWED FOR TEST
3289  015770  012737  015770  001106            MOV     #.,$LPADR       ;SETUP THE LOOP ADDRESS
3290  015776  005000                            CLR     R0              ;CLEAR SWITCH
3291  016000  005004                            CLR     R4              ;FORM WORD COUNT IN R4
3292  016002  013701  001526                    MOV     @#LS,R1
3293  016006  163701  001524                    SUB     @#FS,R1
3294  016012  002004                            BGE     1$              ;BRANCH IF FS < OR = LS
3295  016014  063701  001630                    ADD     PRMLMT+22,R1    ;ADD MAXIMUM SECTOR ADDRESS TO
3296  016020  005201                            INC     R1              ;MAKE THE DIFFERENCE POSITIVE
3297  016022  005100                            COM     R0              ;SET SWITCH
3298  016024  062704  000400            1$:     ADD     #256.,R4
3299  016030  005301                            DEC     R1
3300  016032  002374                            BGE     1$
3301  016034  005404                            NEG     R4
3302  016036  010405                            MOV     R4,R5           ;COPY NORMAL WORD COUNT INTO SMALL WC
3303  016040  005700                            TST     R0              ;SWITCH SET?
3304  016042  001412                            BEQ     3$              ;NO--BRANCH
3305  016044  005005                            CLR     R5              ;FORM WORD COUNT FOR LS < FS
3306  016046  013701  001630                    MOV     PRMLMT+22,R1
3307  016052  163701  001524                    SUB     @#FS,R1
3308  016056  062705  000400            2$:     ADD     #256.,R5
3309  016062  005301                            DEC     R1
3310  016064  002374                            BGE     2$
3311  016066  005405                            NEG     R5
3312  016070  113737  001524  004174  3$:     MOVB    @#FS,@#DTAOPB+10 ;SECTOR
3313  016076  012737  047662  004172            MOV     #BUFFER,@#DTAOPB+6 ;DATA BUFFER
3314  016104  012737  016426  001252            MOV     #EXIT20,@#BYPASS
3315  016112  012706  001100            TEST20: MOV     #STACK,SP       ;LOAD THE STACK POINTER
3316  016116  005037  001334                    CLR     @#CEFLG         ;CLEAR THE WRITE CHECK ERROR FLAG
3317  016122  013701  001510                    MOV     @#FC,R1 ;PICKUP FIRST CYLINDER.
3318  016126  000407                            BR      2$
3319  016130  005720                    1$:     TST     (R0)+           ;MOVE TO NEXT DATA PATTERN
3320  016132  022700  000040                    CMP     #16.*2.,R0      ;OUT OF PATTERNS?
3321  016136  003004                            BGT     3$              ;NO--BRANCH
3322  016140  004737  027600                    JSR     R0,@#INCCYL     ;MOVE TO NEXT CYLINDER
3323  016144  000530                            BR      EXIT20          ;OUT OF CYLINDERS
3324  016146  005000                    2$:     CLR     R0              ;START WITH PATTERN 0
```

```
3325  016150  036037  001424  001530  3$:     BIT     BITS(R0),@#PAT  ;THIS PATTERN SELECTED?
3326  016156  001764                          BEQ     1$              ;NO--BRANCH
3327  016160  013702  001516                  MOV     @#FT,R2         ;FIRST TRACK
3328  016164  010137  004176                  MOV     R1,@#DTADPB+12  ;CYLINDER
3329  016170  110237  004175          4$:     MOVB    R2,@#DTADPB+11  ;TRACK
3330  016174  010437  004170                  MOV     R4,@#DTADPB+4   ;WORD COUNT
3331  016200  023701  001512                  CMP     LC,R1           ;LAST DISK CYLINDER?
3332  016204  003005                          BGT     5$              ;NO--BRANCH
3333  016206  022702  000022                  CMP     #18.,R2         ;LAST DISK TRACK?
3334  016212  003002                          BGT     5$              ;NO--BRANCH
3335  016214  010537  004170                  MOV     R5,@#DTADPB+4   ;SHORT WORD COUNT
3336  016220  017703  162714          5$:     MOV     @SWR,R3         ;INHIBIT WRITE AND
3337  016224  005103                          COM     R3              ;WRITE CHECK?
3338  016226  032703  000030                  BIT     #SW04!SW03,R3
3339  016232  001436                          BEQ     7$              ;YES--BRANCH
3340  016234  004737  030330                  JSR     PC,@#SETBUF     ;MOVE DATA PATTERN INTO THE BUFFER
3341  016240  032777  000020  162672          BIT     #SW04,@SWR      ;INHIBIT WRITE?
3342  016246  001012                          BNE     6$              ;YES--BRANCH
3343  016250  012737  016250  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3344  016256  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3345  016262  012737  000161  004166          MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
3346  016270  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3347  016274  032777  000010  162636  6$:     BIT     #SW03,@SWR      ;INHIBIT WRITE CHECK?
3348  016302  001012                          BNE     7$              ;YES--BRANCH
3349  016304  012737  016304  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3350  016312  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3351  016316  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
3352  016324  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3353  016330  005737  001334          7$:     TST     @#WCEFLG        ;WRITE CHECK ERROR FLAG SET?
3354  016334  001404                          BEQ     8$              ;NO--BRANCH
3355  016336  032777  000001  162574          BIT     #SW00,@SWR      ;PERFORM READ AFTER FATAL "WCE"?
3356  016344  001424                          BEQ     9$              ;NO--BRANCH
3357  016346  032777  000004  162564  8$:     BIT     #SW02,@SWR      ;INHIBIT READ DATA AND SOFTWARE COMPARE?
3358  016354  001020                          BNE     9$              ;YES--BRANCH
3359  016356  012737  016356  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3360  016364  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3361  016370  012737  000171  004166          MOV     #READ,@#DTADPB+2 ;COMMAND=READ
3362  016376  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3363  016402  032777  000002  162530          BIT     #SW01,@SWR      ;COMPARE THE DATA?
3364  016410  001002                          BNE     9$              ;NO--BRANCH
3365  016412  004737  030420                  JSR     PC,@#DATCMP     ;YES--DO IT
3366  016416  004037  027550          9$:     JSR     R0,@#INCTRK     ;MOVE TO NEXT TRACK
3367  016422  000642                          BR      1$              ;OUT OF TRACKS GO TO NEXT PATTERN
3368  016424  000661                          BR      4$              ;LOOP
3369  016426  000004          EXIT20: SCOPE                           ;SCOPE LOOP
```

```
                                .SBTTL  *** EXERCISE TEST ***

                                ;;*****************************************************************
                                ;*TEST 21        RANDOM ADDRESS AND RANDOM PATTERN TEST

                                ;*      STARTING AT "FC" AND GOING THROUGH "LC" THE DISK PACK
                                ;*      IS WRITTEN WITH A RANDOM PATTERN. THE FIRST TWO WORDS
                                ;*      OF EACH SECTOR WILL BE THE BASE OF THE RANDOM GENERATOR
                                ;*      FOR THAT SECTOR.
                                ;*      THE TEST THEN PERFORMS THE FOLLOWING SEQUENCE "R" TIMES
                                ;*      "R" DEFAULTS TO 20,000.
                                ;*
                                ;*      1)      GENERATE A RANDOM ADDRESS
                                ;*      2)      WRITE A RANDOM PATTERN AT THE ADDRESS
                                ;*              GENERATED IN 1.
                                ;*      3)      GENERATE A RANDOM ADDRESS
                                ;*      4)      READ THE SECTOR AT THE ADDRESS
                                ;*              GENERATED IN 3.
                                ;*      5)      DO A SOFTWARE CHECK OF THE DATA READ IN 4.
                                ;*      6)      DO A WRITE CHECK OF THE DATA WRITTEN IN 2
                                ;*      7)      GENERATE A RANDOM ADDRESS
                                ;*      8)      READ THE SECTOR AT THE ADDRESS
                                ;*              GENERATED IN 7.
                                ;*      9)      DO A SOFTWARE CHECK OF THE DATA READ IN 8
                                ;*      10)     DO A WRITE CHECK OF THE DATA WRITTEN IN 2

                                ;;*****************************************************************
016430                          TST21:
016430  000240                          NOP
016432  033737  001426  001236          BIT     BITS+<21*2-40>,TSTNMS+2  ;DO THIS TEST ?
016440  001002                          BNE     64$                      ;YES--BRANCH
016442  000137  017206                  JMP     TST22                    ;NO--GO TO THE NEXT TEST
016446  012737  000021  001102  64$:    MOV     #21,@#STSTNM             ;SET UP TEST NUMBER AND
                                                                         ;CLEAR THE ERROR FLAG (SERFLG)
016454  004737  024560                  JSR     PC,LOOPRM                ;LOAD THE PARMETERS FOR THE TEST
016460  012737  016700  001110          MOV     #TEST21,@#SLPERR  ;SETUP THE LOOP ON ERROR ADDRESS
016466  013777  001102  162446          MOV     $TSTNM,@DISPLAY   ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
016474  013737  001506  001204          MOV     @#PT,$TIMES       ;GET THE ITERATION COUNT
016502  113737  001364  001115          MOVB    ERR.CT,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
016510  012737  016510  001106          MOV     #.,$LPADR         ;SETUP THE LOOP ADDRESS
016516  012737  017204  001252          MOV     #EXIT21,@#BYPASS
016524  012737  176543  023556          MOV     #176543,@#SHINUM ;PRIME THE RANDOM NUMBER GENERATOR
016532  012737  123456  023560          MOV     #123456,@#SLONUM
016540  013737  001510  004176          MOV     @#FC,@#DTADPB+12 ;CYLINDER
016546  013737  001362  004170          MOV     TRCKWC,@#DTADPB+4 ;WORD COUNT
016554  012737  047662  004172          MOV     #BUFFER,@#DTADPB+6 ;BUFFER ADDRESS
016562  012737  000161  004166          MOV     #WRITE,@#DTADPB+2 ;COMMAND
016570  032737  100000  001220          BIT     #SW15,@#C.SWR     ;WRITE THE DISK PACK BEFORE TESTING?
016576  001027                          BNE     3$                ;NO--BRANCH
016600  004037  020736                  JSR     R0,@#FILRAN       ;FILL DATA BUFFER WITH RANDOM DATA
016604  005037  004174          1$:     CLR     @#DTADPB+10       ;SECTOR AND TRACK
016610  012737  016610  001110          MOV     #.,$LPERR         ;SETUP THE ERROR LOOP ADDRESS
016616  012706  001100                  MOV     #STACK,SP         ;LOAD THE STACK POINTER
016622                          2$:
016622  004037  025474                  JSR     R0,@#DRVCAL       ;START A DATA TRANSFER
016626  105237  004175                  INCB    @#DTADPB+11       ;NEXT TRACK
```

```
3426  016632  122737  000023  004175          CMPB    #19.,@#DTADPB+11 ;TIME FOR NEXT CYLINDER
3427  016640  003370                          BGT     2$              ;NO--BRANCH
3428  016642  005237  004176                  INC     @#DTADPB+12
3429  016646  023737  001512  004176          CMP     @#LC,@#DTADPB+12 ;OUT OF CYLINDERS?
3430  016654  002353                          BGE     1$              ;NO--BRANCH
3431  016656  012737  177400  004170  3$:     MOV     #SCTRWC,@#DTADPB+4 ;WORD COUNT
3432  016664  012737  016700  001106          MOV     #TEST21,@#SLPADR
3433  016672  012737  016700  001110          MOV     #TEST21,@#SLPERR
3434  016700  012706  001100          TEST21: MOV     #STACK,SP       ;SET STACK POINTER
3435  016704  004037  031212                  JSR     R0,@#RANADR     ;GENERATE A RANDOM ADDRESS
3436  016710  013737  004174  001340          MOV     @#DTADPB+10,@#SVADR ;SAVE THE TRACK/SECTOR
3437  016716  013737  004176  001342          MOV     @#DTADPB+12,@#SVADR+2 ;SAVE THE CYLINDER
3438  016724  012737  000161  004166          MOV     #WRITE,@#DTADPB+2 ;COMMAND=WRITE DATA
3439  016732  012701  047662                  MOV     #BUFFER,R1      ;BUFFER ADDRESS
3440  016736  010137  004172                  MOV     R1,@#DTADPB+6
3441  016742  004037  031156                  JSR     R0,@#RANPAT     ;GENERATE RANDOM PATTERN
3442  016746  012737  016746  001110          MOV     #.,SLPERR       ;SETUP THE ERROR LOOP ADDRESS
3443  016754  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3444  016760  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3445  016764  004037  031212                  JSR     R0,@#RANADR
3446  016770  012737  000171  004166          MOV     #READ,@#DTADPB+2 ;COMMAND=READ DATA
3447  016776  012737  050662  004172          MOV     #BUFFER+512.,@#DTADPB+6 ;BUFFER ADDRESS
3448  017004  012737  017004  001110          MOV     #.,SLPERR       ;SETUP THE ERROR LOOP ADDRESS
3449  017012  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3450  017016  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3451  017022  004037  030760                  JSR     R0,@#RANCK      ;CHECK THE DATA
3452  017026  013737  001340  004174          MOV     @#SVADR,@#DTADPB+10 ;GET ADDRESS OF WHERE THE LAST
3453  017034  013737  001342  004176          MOV     @#SVADR+2,@#DTADPB+12 ;WRITE WAS PERFORMED
3454  017042  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
3455  017050  012737  047662  004172          MOV     #BUFFER,@#DTADPB+6 ;DATA BUFFER ADDRESS
3456  017056  012737  017056  001110          MOV     #.,SLPERR       ;SETUP THE ERROR LOOP ADDRESS
3457  017064  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3458  017070  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3459  017074  004037  031212                  JSR     R0,@#RANADR     ;GENERATE A RANDOM ADDRESS
3460  017100  012737  000171  004166          MOV     #READ,@#DTADPB+2 ;COMMAND=READ
3461  017106  012737  050662  004172          MOV     #BUFFER+512.,@#DTADPB+6 ;DATA BUFFER ADDRESS
3462  017114  012737  017114  001110          MOV     #.,SLPERR       ;SETUP THE ERROR LOOP ADDRESS
3463  017122  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3464  017126  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3465  017132  004037  030760                  JSR     R0,@#RANCK      ;CHECK THE DATA
3466  017136  013737  001340  004174          MOV     @#SVADR,@#DTADPB+10 ;GET DISK ADDRESS OF THE
3467  017144  013737  001342  004176          MOV     @#SVADR+2,@#DTADPB+12 ;LAST WRITE
3468  017152  012737  000151  004166          MOV     #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
3469  017160  012737  047662  004172          MOV     #BUFFER,@#DTADPB+6 ;DATA BUFFER ADDRESS
3470  017166  012737  017166  001110          MOV     #.,SLPERR       ;SETUP THE ERROR LOOP ADDRESS
3471  017174  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
3472  017200  004037  025474                  JSR     R0,@#DRVCAL     ;START A DATA TRANSFER
3473  017204  000004                  EXIT21: SCOPE                   ;LOOP ?
3474
3475                                  .SBTTL  *** RP04 ACCESS TIME ADJUSTMENT TEST ***
3476
3477                                  ;;*****************************************************************
3478                                  ;*TEST 22        RP04 ACCESS TIME ADJUSTMENT TEST
3479
3480                                  ;*      THIS TEST PERFORMS SEEKS BETWEEN CYLINDERS 0 & 136 TO ALLOW THE
3481                                  ;*      OPERATOR TO ADJUST THE ACCESS TIME ON AN RP04 USING THE
```

```
                                        ;*      DDU.  THE PROGRAM STALLS APPROXIMATELY 5 SECONDS BETWEEN SEEKS
                                        ;*      SO THAT THE ACCESS TIME INDICATORS ON THE DDU MAY BE OBSERVED.
                                        ;*
                                        ;;**************************************************************
017206                                  TST22:
017206  000240                                  NOP
017210  033737  001430  001236                  BIT     BITS+<22*2-40>,TSTNMS+2  ;DO THIS TEST ?
017216  001002                                  BNE     64$                      ;YES--BRANCH
017220  000137  017356                          JMP     $EOP                     ;NO--GO TO THE END OF THE PROGRAM
017224  012737  000022  001102  64$:     MOV     #22,@#STSTNM             ;SET UP TEST NUMBER AND
                                                                         ;CLEAR THE ERROR FLAG (SERFLG)
017232  004737  024560                          JSR     PC,LOOPRM                ;LOAD THE PARMETERS FOR THE TEST
017236  012737  017274  001110                  MOV     #TEST22,@#SLPERR  ;SETUP THE LOOP ON ERROR ADDRESS
017244  013777  001102  161670                  MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
017252  013737  001506  001204                  MOV     @#RPT,$TIMES     ;GET THE ITERATION COUNT
017260  112737  000144  001115                  MOVB    #100.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
017266  012737  017274  001106                  MOV     #TEST22,$LPADR   ;SETUP THE LOOP ADDRESS
017274  012706  001100          TEST22: MOV     #STACK,SP        ;SETUP THE STACK POINTER
017300  013737  001512  004116                  MOV     LC,DPB.A+12      ;ENDING CYLINDER
017306  112737  000105  004106                  MOVB    #SEEK,@#DPB.A+2  ;SEEK=COMMAND
017314  004037  025002                          JSR     R0,@#CALL.A      ;GO EXECUTE THE COMMAND
017320  004037  026334                          JSR     R0,STALL         ;STALL
017324  001360                                  .WORD   STALL3           ;ADDRESS OF STALL VALUE
017326  013737  001510  004116                  MOV     FC,DPB.A+12      ;STARTING CYLINDER
017334  112737  000105  004106                  MOVB    #SEEK,@#DPB.A+2  ;SEEK=COMMAND
017342  004037  025002                          JSR     R0,@#CALL.A      ;GO EXECUTE THE COMMAND
017346  004037  026334                          JSR     R0,STALL         ;STALL
017352  001360                                  .WORD   STALL3           ;ADDRESS OF STALL VALUE
017354  000004                  EXIT22: SCOPE                    ;LOOP ?
                                        
                                        
                                        
                                        .SBTTL  END OF PASS ROUTINE
                                        
                                        ;;**************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 8. PASSES THRU THE PROGRAM
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO RESTART
                                        
017356                                  $EOP:
017356  104401  017364                          TYPE    ,65$             ;;TYPE ASCIZ STRING
017362  000410                                  BR      64$              ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CR><LF><LF>/END OF PASS/
017404                                  64$:
017404  005737  001232                          TST     @#DRVSEL         ;ANY DRIVES SELECTED?
017410  001434                                  BEQ     1$               ;NO--BRANCH
017412  104401  017420                          TYPE    ,67$             ;;TYPE ASCIZ STRING
017416  000405                                  BR      66$              ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  / ON DRIVE/
017432                                  66$:
017432  013746  001254                          MOV     @#CHKDRV,-(SP)   ;;SAVE @#CHKDRV FOR TYPEOUT
017436  104403                                  TYPOS                    ;;GO TYPE--OCTAL ASCII
017440     002                                  .BYTE   2                ;;TYPE 2 DIGIT(S)
017441     000                                  .BYTE   0                ;;SUPPRESS LEADING ZEROS
017442  104401  017450                          TYPE    ,69$             ;;TYPE ASCIZ STRING
```

END OF PASS ROUTINE

```
                                                        BR      68$                     ;;GET OVER THE ASCIZ
                                                ;;69$:  .ASCIZ  /   ERRORS DETECTED=/
                                                68$:
                                                        MOV     @#SERTTL,-(SP)          ;;SAVE @#SERTTL FOR TYPEOUT
                                                        TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                                1$:     CLR     @#SERTTL                ;ZERO ERROR TOTAL
                                                        CLR     $ISTNM                  ;;ZERO THE TEST NUMBER
                                                        CLR     $TIMES                  ;;ZERO THE NUMBER OF ITERATIONS
                                                        INC     $PASS                   ;;INCREMENT THE PASS NUMBER
                                                        BIC     #100000,$PASS           ;;DON'T ALLOW A NEG. NUMBER
                                                        DEC     (PC)+                   ;;LOOP?
                                                SEOPCT: .WORD   8.
                                                        BGT     $DOAGN                  ;;YES
                                                        MOV     (PC)+,@(PC)+            ;;RESTORE COUNTER
                                                SENOCT: .WORD   8.
                                                        SEOPCT
                                                        TYPE    ,65$                    ;;TYPE ASCIZ STRING
                                                        BR      64$                     ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <CR><LF>/END OF TEST/
                                                64$:
                                                        TYPE    ,$ENULL                 ;TYPE NULL CHARACTER
                                                $GET42: MOV     @#42,R0                 ;;GET MONITOR ADDRESS
                                                        BEQ     $DOAGN                  ;;BRANCH IF NO MONITOR
                                                        RESET                           ;;CLEAR THE WORLD
                                                $ENDAD: JSR     PC,(R0)                 ;;GO TO MONITOR
                                                        NOP                             ;;SAVE ROOM
                                                        NOP                             ;;FOR
                                                        NOP                             ;;ACT11
                                                $DOAGN:
                                                        JMP     @(PC)+                  ;;RETURN
                                                $RTNAD: .WORD   RESTART
                                                $ENULL: .BYTE   -1,-1,0                 ;;NULL CHARACTER STRING
                                                        .EVEN
```

```
                                        .SBTTL  *** SYSMAC SUBROUTINES ***

                                        .SBTTL  ERROR HANDLER ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                        ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                        ;*AND GO TO TYPERR ON ERROR
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW15=1         HALT ON ERROR
                                        ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                        ;*SW10=1         BELL ON ERROR
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

017624                                  $ERROR:
017624  104407                                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
017626  032777  000400  161304                  BIT     #SW08,@SWR      ;SEND ERROR MESSAGE TO TTY?
017634  001411                                  BEQ     7$              ;YES--BRANCH
017636  005737  001230                          TST     @#LPTAVL        ;IS THERE A LINE PRINTER AVAILABLE?
017642  001406                                  BEQ     7$              ;NO--BRANCH
017644  013737  001420  001150                  MOV     @#LPS,@#$TPS    ;YES--SETUP STATUS
017652  013737  001422  001152                  MOV     @#LPB,@#$TPB    ;AND BUFFER REG.'S FOR LINE PRINTER
017660  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
017664  001775                                  BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
017666  013777  001102  161246                  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
017674  032777  002000  161236                  BIT     #BIT10,@SWR     ;;BELL ON ERROR?
017702  001402                                  BEQ     1$              ;;NO - SKIP
017704  104401  001210                          TYPE    ,$BELL          ;;RING BELL
017710  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
017714  011637  001116                          MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
017720  162737  000002  001116                  SUB     #2,$ERRPC
017726  117737  161164  001114                  MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
017734  032777  020000  161176                  BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
017742  001004                                  BNE     20$             ;;SKIP TYPEOUTS
017744  004737  020032                          JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
017750  104401  001215                          TYPE    ,$CRLF
017754                                  20$:
017754  005777  161160          2$:     TST     @SWR            ;;HALT ON ERROR
017760  100002                                  BPL     3$              ;;SKIP IF CONTINUE
017762  000000                                  HALT                    ;;HALT ON ERROR!
017764  104407                                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
017766  032777  001000  161144  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
017774  001402                                  BEQ     4$              ;;BR IF NO
017776  013716  001110                          MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
020002  005737  001206          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
020006  001402                                  BEQ     5$              ;;BR IF NONE
020010  013716  001206                          MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
020014                                  5$:
020014  013737  001414  001150                  MOV     @#TPS,@#$TPS    ;SET STATUS AND BUFFER REG.'S
020022  013737  001416  001152                  MOV     @#TPB,@#$TPB    ;FOR TTY
020030  000002                                  RTI                     ;RETURN FROM ERROR CALL

                                        ;;**********************************************************************
```

```
                                .SBTTL  TYPERR - TYPE ERROR ROUTINE
                                ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
                                ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
                                ;TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
                                ;CONCERNING THE ERROR.
                                ;CALL
                                ;       JSR     PC,@#TYPERR
                                ;       RETURN

020032  113737  001102  001176  TYPERR: MOVB    @#$TSTNM,@#$TMPO ;SAVE THE TEST NUMBER
020040  104412                          SAVREG                  ;SAVE R0 - R5
020042  162700  000004                  SUB     #4,R0           ;FORM TEST PC
020046  010037  001162                  MOV     R0,@#$REG0      ;COPY R0-R5 IN $REG0-$REG5
020052  010137  001164                  MOV     R1,@#$REG1
020056  010237  001166                  MOV     R2,@#$REG2
020062  010337  001170                  MOV     R3,@#$REG3
020066  010437  001172                  MOV     R4,@#$REG4
020072  010537  001174                  MOV     R5,@#$REG5
020076  113700  001114                  MOVB    @#$ITEMB,R0     ;PICKUP ERROR ITEM NUMBER
020102  010001                          MOV     R0,R1           ;AND COPY IT INTO R1
020104  005300                          DEC     R0              ;FORM INDEX FOR ERROR TABLE
020106  106300                          ASLB    R0
020110  106300                          ASLB    R0
020112  106300                          ASLB    R0
020114  103002                          BCC     1$              ;IS ERROR > 37?
020116  062700  000240                  ADD     #ITEM41-$ERRTB,R0 ;YES--FORM OFFSET
020122  062700  004306          1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS
020126  012037  020142                  MOV     (R0)+,2$        ;GET ERROR MESSAGE (EM) POINTER
020132  001447                          BEQ     7$              ;BRANCH IF THERE ISN'T ONE
020134  104401  001215                  TYPE    ,$CRLF          ;"CARRIAGE RETURN - LINE FEED
020140  104401                          TYPE
020142  000000          2$:     .WORD   0                       ;"EM" POINTER GOES HERE
020144  162701  000041                  SUB     #41,R1          ;SPECIAL ERROR ITEM NUMBER?
020150  100440                          BMI     7$              ;NO--BRANCH
020152  013701  001260                  MOV     @#SVSTAT,R1     ;GET STATUS/ERROR INDICATOR
020156  106301                          ASLB    R1              ;STRIP "DONE" BIT (BIT07)
020160  006301                          ASL     R1              ;STRIP "ERROR" BIT (BIT15)
020162  012702  004254                  MOV     #STATBL,R2      ;1ST ADDRESS ON STATUS MESSAGE POINTERS
020166  005003                          CLR     R3              ;CARRIAGE RETURN-LINE FEED SWITCH
020170  104401  020176                  TYPE    ,65$            ;;TYPE ASCIZ STRING
020174  000402                          BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  / (/
020202                  64$:
020202  012237  020224          3$:     MOV     (R2)+,5$        ;MESSAGE POINTER
020206  006301                          ASL     R1              ;TYPE THIS MESSAGE?
020210  103013                          BCC     6$              ;NO--BRANCH
020212  005103                          COM     R3              ;YES--TYPE A "CR" & "LF"?
020214  001002                          BNE     4$              ;NO--BRANCH
020216  104401  001215                  TYPE    ,$CRLF          ;YES
020222  104401                  4$:     TYPE
020224  000000          5$:     .WORD   0                       ;MESSAGE POINTER GOES HERE
020226  005701                          TST     R1              ;MORE TO TYPE?
020230  001403                          BEQ     6$              ;NO--BRANCH
020232  104401  044010                  TYPE    ,MSG.SP         ;YES--SPACES
020236  000761                          BR      3$              ;LOOP
020240  001360          6$:     BNE     3$                      ;BRANCH IF NOT FINISHED
```

```
3683  020242  104401  020250                ;;67$:  TYPE    ,67$            ;;TYPE ASCIZ STRING
3684  020246  000401                              BR      66$             ;;GET OVER THE ASCIZ
3685                                        ;;67$:  .ASCIZ  /)/
3686  020252                                66$:
3687  020252  012037  020266                7$:     MOV     (R0)+,8$        ;PICK UP DATA HEADER (DH) POINTER
3688  020256  001404                              BEQ     9$              ;BRANCH IF NONE
3689  020260  104401  001215                      TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3690  020264  104401                              TYPE
3691  020266  000000                        8$:     .WORD   0               ;"DH" POINTER GOES HERE
3692  020270  012001                        9$:     MOV     (R0)+,R1        ;PICKUP DATA TABLE (DT) POINTER
3693  020272  001450                              BEQ     20$             ;BRANCH IF NONE
3694  020274  005005                              CLR     R5              ;SET INDENT SWITCH
3695  020276  012000                              MOV     (R0)+,R0        ;DATA FORMAT (DF) POINTER
3696  020300  012002                              MOV     (R0)+,R2        ;NUMBER OF DH'S TO TYPE
3697  020302  001441                              BEQ     17$             ;BRANCH IF DH NUMBER IS 0
3698  020304  005105                              COM     R5              ;NO INDENT
3699  020306  104401  001215                      TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3700  020312  112003                        10$:    MOVB    (R0)+,R3        ;NUMBER OF DATA WORDS TO TYPE
3701  020314  112004                              MOVB    (R0)+,R4        ;AND HOW TO TYPE THEM
3702  020316  006004                        11$:    ROR     R4              ;OCTAL OR DECIMAL?
3703  020320  103403                              BCS     12$             ;DECIMAL--BRANCH
3704  020322  013146                              MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3705  020324  104402                              TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3706  020326  000402                              BR      13$
3707  020330                                12$:
3708  020330  013146                              MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3709  020332  104405                              TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
3710  020334  005303                        13$:    DEC     R3              ;MORE NUMBERS TO TYPE?
3711  020336  001403                              BEQ     14$             ;NO--BRANCH
3712  020340  104401  044010                      TYPE    ,MSG.SP         ;YES--TYPE SEPERATORS
3713  020344  000764                              BR      11$             ;LOOP
3714  020346  005302                        14$:    DEC     R2              ;MORE DH'S?
3715  020350  003421                              BLE     20$             ;NO--BRANCH
3716  020352  104401  001215                      TYPE    ,$CRLF          ;YES--START A NEW LINE
3717  020356  005105                              COM     R5              ;INDENT?
3718  020360  001002                              BNE     15$             ;NO--BRANCH
3719  020362  104401  044010                      TYPE    ,MSG.SP         ;YES--TYPE SPACES
3720  020366  012037  020374                15$:    MOV     (R0)+,16$       ;GET NEXT DH
3721  020372  104401                              TYPE                    ;AND TYPE IT
3722  020374  000000                        16$:    .WORD   0               ;DH POINTER GOES HERE
3723  020376  104401  001215                      TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
3724  020402  005705                              TST     R5              ;INDENT?
3725  020404  001342                              BNE     10$             ;NO--BRANCH
3726  020406  104401  044010                17$:    TYPE    ,MSG.SP         ;YES--TYPE SPACES
3727  020412  000737                              BR      10$             ;LOOP
3728  020414  104413                        20$:    RESREG                  ;RESTORE R0 - R5
3729  020416  000207                              RTS     PC              ;RETURN
3730
3731                                        .SBTTL  TYPE ROUTINE
3732
3733                                        ;;**************************************************************
3734                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
3735                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
3736                                        ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
3737                                        ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
3738                                        ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
```

```
3739                                    ;*
3740                                    ;*CALL:
3741                                    ;*1) USING A TRAP INSTRUCTION
3742                                    ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
3743                                    ;*OR
3744                                    ;*      TYPE
3745                                    ;*      MESADR
3746                                    ;*
3747
3748  020420  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
3749  020424  100002                          BPL     1$              ;;BR IF YES
3750  020426  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
3751  020430  000407                          BR      3$              ;;LEAVE
3752  020432  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
3753  020434  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
3754  020440  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
3755  020442  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
3756  020444  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
3757  020446  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
3758  020450  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
3759  020454  000002                          RTI                     ;;RETURN
3760  020456  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
3761  020462  001430                          BEQ     8$
3762  020464  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
3763  020470  001006                          BNE     5$
3764  020472  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
3765  020474  104401                          TYPE                    ;;TYPE 1 CR AND LF
3766  020476  001215                          $CRLF
3767  020500  105037  020634                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
3768  020504  000755                          BR      2$              ;;GET NEXT CHARACTER
3769  020506  004737  020570          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
3770  020512  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
3771  020516  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
3772  020520  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
3773                                                                  ;;AND THE NULL CHAR.
3774  020524  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
3775  020530  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
3776  020532  004737  020570                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
3777  020536  105337  020634                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
3778  020542  000770                          BR      7$              ;;LOOP
3779
3780                                    ;HORIZONTAL TAB PROCESSOR
3781
3782  020544  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3783  020550  004737  020570          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
3784  020554  132737  000007  020634          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3785  020562  001372                          BNE     9$              ;;TAB STOP
3786  020564  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
3787  020566  000724                          BR      2$              ;;GET NEXT CHARACTER
3788  020570  105777  160354          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3789  020574  100375                          BPL     $TYPEC
3790  020576  116677  000002  160346          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3791  020604  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3792  020612  001003                          BNE     1$              ;;BRANCH IF NO
3793  020614  105037  020634                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
3794  020620  000406                          BR      $TYPEX          ;;EXIT
```

```
3795  020622  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
3796  020630  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
3797  020632  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
3798  020634  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
3799  020636  000207                  $TYPEX: RTS     PC
3800
3801
3802                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3803
3804                                  ;;**************************************************************
3805                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3806                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
3807                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3808                                  ;*CALL:
3809                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3810                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
3811                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3812                                  ;*      .BYTE   M               ;;M=1 OR 0
3813                                  ;*                                      ;;1=TYPE LEADING ZEROS
3814                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
3815                                  ;*
3816                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3817                                  ;*$TYPOS OR $TYPOC
3818                                  ;*CALL:
3819                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3820                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
3821                                  ;*
3822                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3823                                  ;*CALL:
3824                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3825                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
3826
3827  020640  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
3828  020644  116637  000001  021063          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
3829  020652  112637  021065                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
3830  020656  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
3831  020662  000406                          BR      $TYPON
3832  020664  112737  000001  021063  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
3833  020672  112737  000006  021065          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
3834  020700  112737  000005  021062  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
3835  020706  010346                          MOV     R3,-(SP)        ;;SAVE R3
3836  020710  010446                          MOV     R4,-(SP)        ;;SAVE R4
3837  020712  010546                          MOV     R5,-(SP)        ;;SAVE R5
3838  020714  113704  021065                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
3839  020720  005404                          NEG     R4
3840  020722  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
3841  020726  110437  021064                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
3842  020732  113704  021063                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
3843  020736  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
3844  020742  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
3845  020744  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
3846  020746  000404                          BR      3$              ;;GO DO MSB
3847  020750  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
3848  020752  006105                          ROL     R5
3849  020754  006105                          ROL     R5
3850  020756  010503                          MOV     R5,R3
```

```
3851  020760  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
3852  020762  105337  021064                  DECB    $OMODE          ;;TYPE THIS DIGIT?
3853  020766  100016                          BPL     7$              ;;BR IF NO
3854  020770  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
3855  020774  001002                          BNE     4$              ;;TEST FOR 0
3856  020776  005704                          TST     R4              ;;SUPPRESS THIS 0?
3857  021000  001403                          BEQ     5$              ;;BR IF YES
3858  021002  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
3859  021004  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
3860  021010  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
3861  021014  110337  021060                  MOVB    R3,8$           ;;SAVE FOR TYPING
3862  021020  104401  021060                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
3863  021024  105337  021062          7$:     DECB    $OCNT           ;;COUNT BY 1
3864  021030  003347                          BGT     2$              ;;BR IF MORE TO DO
3865  021032  002402                          BLT     6$              ;;BR IF DONE
3866  021034  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3867  021036  000744                          BR      2$              ;;GO DO THE LAST DIGIT
3868  021040  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
3869  021042  012604                          MOV     (SP)+,R4        ;;RESTORE R4
3870  021044  012603                          MOV     (SP)+,R3        ;;RESTORE R3
3871  021046  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3872  021054  012616                          MOV     (SP)+,(SP)
3873  021056  000002                          RTI                     ;;RETURN
3874  021060     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3875  021061     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3876  021062     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3877  021063     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
3878  021064  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
3879
3880                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3881
3882                                  ;;**************************************************************
3883                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
3884                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
3885                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
3886                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
3887                                  ;*REPLACED WITH SPACES.
3888                                  ;*CALL:
3889                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
3890                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
3891
3892  021066                          $TYPDS:
3893  021066  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3894  021070  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3895  021072  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3896  021074  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3897  021076  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3898  021100  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
3899  021104  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
3900  021110  100004                          BPL     1$              ;;BR IF INPUT IS POS.
3901  021112  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
3902  021114  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
3903  021122  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
3904  021124  012703  021302                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
3905  021130  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
3906  021134  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
```

```
3907  021136  016001  021272                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
3908  021142  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
3909  021144  002402                          BLT     4$              ;;BR IF DONE
3910  021146  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
3911  021150  000774                          BR      3$
3912  021152  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
3913  021154  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
3914  021156  001002                          BNE     5$              ;;FALL THROUGH IF 0
3915  021160  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
3916  021162  100407                          BMI     7$              ;;BR IF YES
3917  021164  106316                  5$:     ASLB    (SP)            ;;MSD?
3918  021166  103003                          BCC     6$              ;;BR IF NO
3919  021170  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
3920  021176  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
3921  021202  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
3922  021206  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
3923  021210  005720                          TST     (R0)+           ;;JUST INCREMENTING
3924  021212  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
3925  021216  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
3926  021220  003002                          BGT     8$              ;;GO TO EXIT
3927  021222  010502                          MOV     R5,R2           ;;GET THE LSD
3928  021224  000764                          BR      6$              ;;GO CHANGE TO ASCII
3929  021226  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
3930  021230  100003                          BPL     9$              ;;BR IF NO
3931  021232  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
3932  021240  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
3933  021242  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
3934  021244  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
3935  021246  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
3936  021250  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3937  021252  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
3938  021254  104401  021302                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
3939  021260  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
3940  021266  012616                          MOV     (SP)+,(SP)
3941  021270  000002                          RTI                     ;;RETURN TO USER
3942  021272  023420                  $DTBL:  10000.
3943  021274  001750                          1000.
3944  021276  000144                          100.
3945  021300  000012                          10.
3946  021302  000004                  $DBLK:  .BLKW   4
3947
3948                                  .TTL    TTY INPUT ROUTINE
3949
3950                                  ;;*****************************************************************
3951                                  .ENABL  LSB
3952  021312  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
3953  021314  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
3954  021316  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
3955  021320  000002                  $TKQSRT: .BLKB  2               ;;TTY KEYBOARD QUEUE
3956          021322                  $TKQEND=.
3957
3958                                  ;*TK INITIALIZE ROUTINE
3959                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
3960                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
3961                                  ;
3962                                  ;*CALL:
```

```
3963                                    ;*      JSR     PC,$TKINT
3964                                    ;*      RETURN
3965                                    ;
3966  021322  005037  021312            $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
3967  021326  012737  021320  021314            MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
3968  021334  013737  021314  021316            MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
3969  021342  012737  021372  000060            MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
3970  021350  012737  000200  000062            MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
3971  021356  005777  157564                    TST     @$TKB           ;;CLEAR DONE FLAG
3972  021362  012777  000100  157554            MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
3973  021370  000207                            RTS     PC              ;;RETURN TO CALLER
3974
3975                                    ;*TK SERVICE ROUTINE
3976                                    ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
3977                                    ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
3978                                    ;*IT IN THE QUEUE.
3979                                    ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
3980                                    ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START2)
3981                                    ;
3982  021372  117746  157550            $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
3983  021376  042716  177600                    BIC     #↑C177,(SP)     ;;STRIP THE JUNK
3984  021402  021627  000003                    CMP     (SP),#3         ;;IS IT A CONTROL C?
3985  021406  001007                            BNE     1$              ;;BRANCH IF NO
3986  021410  104401  022522                    TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
3987  021414  004737  021322                    JSR     PC,$TKINT       ;;INIT THE KEYBOARD
3988  021420  005726                            TST     (SP)+           ;;CLEAN UP STACK
3989  021422  000137  004660                    JMP     START2          ;;CONTROL C RESTART
3990  021426  021627  000007            1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
3991  021432  001004                            BNE     2$              ;;BRANCH IF NO
3992  021434  022737  000176  001140            CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
3993  021442  001500                            BEQ     6$              ;;GO TO SWR CHANGE
3994
3995  021444                            2$:
3996  021444  022737  000002  021312            CMP     #2,$TKCNT       ;;IS THE QUEUE FULL?
3997  021452  001004                            BNE     3$              ;;BRANCH IF NO
3998  021454  104401  001210                    TYPE    ,$BELL          ;;RING THE TTY BELL
3999  021460  005726                            TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
4000  021462  000451                            BR      5$              ;;EXIT
4001  021464  021627  000023            3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
4002  021470  001021                            BNE     32$             ;;BRANCH IF NO
4003  021472  005077  157446                    CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
4004  021476  005726                            TST     (SP)+           ;;CLEAN CHAR OFF STACK
4005  021500  105777  157440            31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
4006  021504  100375                            BPL     31$             ;;LOOP UNTIL ITS THERE
4007  021506  117746  157434                    MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
4008  021512  042716  177600                    BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII
4009  021516  022627  000021                    CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
4010  021522  001366                            BNE     31$             ;;BRANCH IF NO
4011  021524  012777  000100  157412            MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
4012  021532  000002                            RTI                     ;;RETURN
4013  021534  005237  021312            32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
4014  021540  021627  000140                    CMP     (SP),#140       ;;IS IT UPPER CASE?
4015  021544  002405                            BLT     4$              ;;BRANCH IF YES
4016  021546  021627  000175                    CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
4017  021552  003002                            BGT     4$              ;;BRANCH IF YES
4018  021554  042716  000040                    BIC     #40,(SP)        ;;MAKE IT UPPER CASE
```

```
4014  021560  112677  177530          4$:     MOVB    (SP)+,@STKQIN   ;;AND PUT IT IN QUEUE
4020  021564  005237  021314                  INC     $TKQIN          ;;UPDATE THE POINTER
4021  021570  023727  021314  021322          CMP     $TKQIN,#STKQEND ;;GO OFF THE END?
4022  021576  001003                          BNE     5$              ;;BRANCH IF NO
4023  021600  012737  021320  021314          MOV     #STKQSRT,$TKQIN ;;RESET THE POINTER
4024  021606  000002                  5$:     RTI                     ;;RETURN
4025
4026                                          ;;**********************************************************
4027                                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
4028                                          ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
4029                                          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
4030                                          ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
4031  021610  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
4032  021616  001124                          BNE     1$              ;;EXIT IF NOT
4033  021620  105777  157320                  TSTB    @$TKS           ;;IS A CHAR WAITING?
4034  021624  100121                          BPL     1$              ;;IF NOT, EXIT
4035  021626  117746  157314                  MOVB    @$TKB,-(SP)     ;;YES
4036  021632  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
4037  021636  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
4038  021642  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
4039                                                                  ;;AND EXIT
4040
4041                                          ;;**********************************************************
4042                                          ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
4043                                          ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
4044                                          ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
4045  021644  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
4046  021652  001674                          BEQ     2$              ;;BRANCH IF YES
4047  021654  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
4048  021656  004737  021322                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
4049  021662  005077  157256                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
4050  021666  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
4051
4052  021674  104401  022534                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
4053  021700  104401  022541          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
4054  021704  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
4055  021710  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4056  021712  104401  022552                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
4057  021716  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
4058  021720  005046                          CLR     -(SP)           ;;THE NEW SWR
4059  021722  105777  157216          7$:     TSTB    @$TKS           ;;CHAR THERE?
4060  021726  100375                          BPL     7$              ;;IF NOT TRY AGAIN
4061
4062  021730  117746  157212                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
4063  021734  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
4064
4065  021740  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
4066  021744  001015                          BNE     9$              ;;BRANCH IF NOT
4067  021746  104401  022522                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
4068  021752  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
4069  021756  123727  001135  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
4070  021764  001003                          BNE     8$              ;;BRANCH IF NO
4071  021766  012777  000100  157150          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
4072  021774  000137  004660          8$:     JMP     START2          ;;CONTROL-C RESTART
4073
4074
```

```
4075  022000  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
4076  022004  001005                          BNE     10$             ;;BRANCH IF NOT
4077  022006  104401  022527                  TYPE    $CNTLU          ;;YES, ECHO CONTROL-U (↑U)
4078  022012  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
4079  022016  000737                          BR      19$             ;;LET'S TRY IT AGAIN
4080
4081
4082  022020  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR.
4083  022024  001022                          BNE     16$             ;;BRANCH IF NO
4084  022026  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
4085  022032  001403                          BEQ     11$             ;;BRANCH IF YES
4086  022034  016677  000002  157076          MOV     2(SP),@SWR      ;;SAVE NEW SWR
4087  022042  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
4088  022046  104401  001215          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
4089  022052  123727  001135  000001          CMPB    @INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
4090  022060  001003                          BNE     15$             ;;BRANCH IF NOT
4091  022062  012777  000100  157054          MOV     #100,@STKS      ;;RE-ENABLE TTY KBD INTERRUPTS
4092  022070  000002          15$:    RTI                             ;;RETURN
4093  022072  004737  020570          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
4094  022076  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
4095  022102  002420                          BLT     18$             ;;BRANCH IF YES
4096  022104  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
4097  022110  003015                          BGT     18$             ;;BRANCH IF YES
4098  022112  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
4099  022116  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
4100  022122  001403                          BEQ     17$             ;;BRANCH IF YES
4101  022124  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
4102  022126  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
4103  022130  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
4104  022132  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
4105  022136  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
4106  022142  000667                          BR      7$              ;;GET THE NEXT ONE
4107  022144  104401  001214          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
4108  022150  000720                          BR      20$             ;;SIMULATE CONTROL-U
4109                                  .DSABL  LSB
4110
4111
4112                                  ;;*****************************************************************
4113                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
4114                                  ;*CALL:
4115                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
4116                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
4117                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
4118                                  ;
4119
4120  022152  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
4121  022154  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
4122  022162  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
4123  022166  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
4124  022170  012746  022176                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
4125  022174  000002                          RTI                     ;;POP NEW PC AND PS
4126  022176                          64$:
4127  022176  005737  021312          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
4128  022202  001775                          BEQ     1$
4129  022204  005337  021312                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
4130  022210  117766  177102  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
```

```
 4131  022216  005237  021316                 INC     $TKQOUT                 ;;UPDATE THE POINTER
 4132  022222  023727  021316  021322         CMP     $TKQOUT,#STKQEND        ;;DID IT GO OFF OF THE END?
 4133  022230  001003                         BNE     2$                      ;;BRANCH IF NO
 4134  022232  012737  021320  021316         MOV     #STKQSRT,STKQOUT        ;;RESET THE POINTER
 4135  022240  000002                 2$:     RTI                             ;;RETURN
 4136                                         ;;**************************************************************
 4137                                         ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 4138                                         ;*CALL:
 4139                                         ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 4140                                         ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 4141                                         ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 4142
 4143  022242  010346                 $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
 4144  022244  005046                         CLR     -(SP)                   ;;CLEAR THE RUBOUT KEY
 4145  022246  012703  022476         1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
 4146  022252  022703  022522         2$:     CMP     #$TTYIN+20.,R3          ;;BUFFER FULL?
 4147  022256  101456                         BLOS    4$                      ;;BR IF YES
 4148  022260  104410                         RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
 4149  022262  112613                         MOVB    (SP)+,(R3)              ;;GET CHARACTER
 4150  022264  122713  000177         10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
 4151  022270  001022                         BNE     5$                      ;;BR IF NO
 4152  022272  005716                         TST     (SP)                    ;;IS THIS THE FIRST RUBOUT?
 4153  022274  001007                         BNE     6$                      ;;BR IF NO
 4154  022276  112737  000134  022474         MOVB    #'\,9$                  ;;TYPE A BACK SLASH
 4155  022304  104401  022474                 TYPE    9$
 4156  022310  012716  177777                 MOV     #-1,(SP)                ;;SET THE RUBOUT KEY
 4157  022314  005303                 6$:     DEC     R3                      ;;BACKUP BY ONE
 4158  022316  020327  022476                 CMP     R3,#$TTYIN              ;;STACK EMPTY?
 4159  022322  103434                         BLO     4$                      ;;BR IF YES
 4160  022324  111337  022474                 MOVB    (R3),9$                 ;;SETUP TO TYPEOUT THE DELETED CHAR.
 4161  022330  104401  022474                 TYPE    9$                      ;;GO TYPE
 4162  022334  000746                         BR      2$                      ;;GO READ ANOTHER CHAR.
 4163  022336  005716                 5$:     TST     (SP)                    ;;RUBOUT KEY SET?
 4164  022340  001406                         BEQ     7$                      ;;BR IF NO
 4165  022342  112737  000134  022474         MOVB    #'\,9$                  ;;TYPE A BACK SLASH
 4166  022350  104401  022474                 TYPE    9$
 4167  022354  005016                         CLR     (SP)                    ;;CLEAR THE RUBOUT KEY
 4168  022356  122713  000025         7$:     CMPB    #25,(R3)                ;;IS CHARACTER A CTRL U?
 4169  022362  001003                         BNE     8$                      ;;BR IF NO
 4170  022364  104401  022527                 TYPE    $CNTLU                  ;;TYPE A CONTROL "U"
 4171  022370  000726                         BR      1$                      ;;GO START OVER
 4172  022372  122713  000022         8$:     CMPB    #22,(R3)                ;;IS CHARACTER A "^R"?
 4173  022376  001011                         BNE     3$                      ;;BRANCH IF NO
 4174  022400  105013                         CLRB    (R3)                    ;;CLEAR THE CHARACTER
 4175  022402  104401  001215                 TYPE    ,$CRLF                  ;;TYPE A "CR" & "LF"
 4176  022406  104401  022476                 TYPE    ,$TTYIN                 ;;TYPE THE INPUT STRING
 4177  022412  000717                         BR      2$                      ;;GO PICKUP ANOTHER CHACTER
 4178  022414  104401  001214         4$:     TYPE    $QUES                   ;;TYPE A '?'
 4179  022420  000712                         BR      1$                      ;;CLEAR THE BUFFER AND LOOP
 4180  022422  111337  022474         3$:     MOVB    (R3),9$                 ;;ECHO THE CHARACTER
 4181  022426  104401  022474                 TYPE    9$
 4182  022432  122723  000015                 CMPB    #15,(R3)+               ;;CHECK FOR RETURN
 4183  022436  001305                         BNE     2$                      ;;LOOP IF NOT RETURN
 4184  022440  105063  177777                 CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
 4185  022444  104401  001216                 TYPE    $LF                     ;;TYPE A LINE FEED
 4186  022450  005726                         TST     (SP)+                   ;;CLEAN RUBOUT KEY FROM THE STACK
```

```
 4187  022452  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 4188  022454  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 4189  022456  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 4190  022464  012766  022476  000004          MOV     #STTYIN,4(SP)
 4191  022472  000002                          RTI                     ;;RETURN
 4192  022474     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
 4193  022475     000                          .BYTE   0               ;;TERMINATOR
 4194  022476  000024                  $TTYIN: .BLKB   20.             ;;RESERVE 20. BYTES FOR TTY INPUT
 4195  022522  041536  005015     000  $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
 4196  022527     136  006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
 4197  022534  043536  005015     000  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
 4198  022541     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
 4199  022546  036440  000040
 4200  022552  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
 4201  022560  020075     000
 4202          022564                  .EVEN
 4203
 4204                                  .SBTTL  SCOPE HANDLER ROUTINE
 4205
 4206                                  ;;**********************************************************************
 4207                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS, IT WILL INCREMENT
 4208                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
 4209                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
 4210                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 4211                                  ;*SW14=1        LOOP ON TEST
 4212                                  ;*SW11=1        INHIBIT ITERATIONS
 4213                                  ;*SW09=1        LOOP ON ERROR
 4214                                  ;*CALL
 4215                                  ;*      SCOPE           ;;SCOPE=IOT
 4216
 4217  022564                  $SCOPE:
 4218  022564  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
 4219  022566  032777  040000  156344  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
 4220  022574  001076                          BNE     $OVER           ;;YES IF SW14=1
 4221                                  ;#####START OF CODE FOR THE XOR TESTER#####
 4222  022576  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
 4223                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 4224  022600  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 4225  022604  012737  022624  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
 4226  022612  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
 4227  022616  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 4228  022622  000450                          BR      $SVLAD          ;;GO TO THE NEXT TEST
 4229  022624  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
 4230  022626  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 4231  022632  000413                          BR      7$              ;;LOOP ON THE PRESENT TEST
 4232  022634                  6$:;#####END OF CODE FOR THE XOR TESTER#####
 4233  022634  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
 4234  022640  001421                          BEQ     3$              ;;BR IF NO
 4235  022642  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
 4236  022650  101015                          BHI     3$              ;;BR IF NO
 4237  022652  032777  001000  156260          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
 4238  022660  001404                          BEQ     4$              ;;BR IF NO
 4239  022662  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
 4240  022670  000440                          BR      $OVER
 4241  022672  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 4242  022676  005037  001204                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
```

```
4243  022702  000412                    BR      1$                ;;ESCAPE TO THE NEXT TEST
4244  022704  032777  004000  156226    3$:     BIT     #BIT11,@SAR       ;;INHIBIT ITERATIONS?
4245  022712  001006                    BNE     1$                ;;BR IF YES
4246  022714  005237  001104            INC     $ICNT             ;;INCREMENT ITERATION COUNT
4247  022720  023737  001204  001104    CMP     $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
4248  022726  002021                    BGE     $OVER             ;;BR IF MORE ITERATION REQUIRED
4249  022730  012737  000001  001104    1$:     MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
4250  022736  013737  023006  001204            MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
4251  022744  105237  001102            $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
4252  022750  011637  001106                    MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
4253  022754  011637  001110                    MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
4254  022760  005037  001206                    CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
4255  022764  112737  000001  001115            MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
4256  022772  013777  001102  156142    $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
4257  023000  013716  001106                    MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
4258  023004  000002                            RTI                       ;;FIXES PS
4259  023006  000001                    $MXCNT: 1                         ;;MAX. NUMBER OF ITERATIONS
4260
4261                                    .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
4262
4263                                    ;;**************************************************************
4264                                    ;*SAVE R0-R5
4265                                    ;*CALL:
4266                                    ;*      SAVREG
4267                                    ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
4268                                    ;*
4269                                    ;*TOP---(+16)
4270                                    ;* +2---(+18)
4271                                    ;* +4---R5
4272                                    ;* +6---R4
4273                                    ;* +8---R3
4274                                    ;*+10---R2
4275                                    ;*+12---R1
4276                                    ;*+14---R0
4277
4278  023010                            $SAVREG:
4279  023010  010046                            MOV     R0,-(SP)          ;;PUSH R0 ON STACK
4280  023012  010146                            MOV     R1,-(SP)          ;;PUSH R1 ON STACK
4281  023014  010246                            MOV     R2,-(SP)          ;;PUSH R2 ON STACK
4282  023016  010346                            MOV     R3,-(SP)          ;;PUSH R3 ON STACK
4283  023020  010446                            MOV     R4,-(SP)          ;;PUSH R4 ON STACK
4284  023022  010546                            MOV     R5,-(SP)          ;;PUSH R5 ON STACK
4285  023024  016646  000022                    MOV     22(SP),-(SP)      ;;SAVE PS OF MAIN FLOW
4286  023030  016646  000022                    MOV     22(SP),-(SP)      ;;SAVE PC OF MAIN FLOW
4287  023034  016646  000022                    MOV     22(SP),-(SP)      ;;SAVE PS OF CALL
4288  023040  016646  000022                    MOV     22(SP),-(SP)      ;;SAVE PC OF CALL
4289  023044  000002                            RTI
4290
4291                                    ;*RESTORE R0-R5
4292                                    ;*CALL:
4293                                    ;*      RESREG
4294  023046                            $RESREG:
4295  023046  012666  000022                    MOV     (SP)+,22(SP)      ;;RESTORE PC OF CALL
4296  023052  012666  000022                    MOV     (SP)+,22(SP)      ;;RESTORE PS OF CALL
4297  023056  012666  000022                    MOV     (SP)+,22(SP)      ;;RESTORE PC OF MAIN FLOW
4298  023062  012666  000022                    MOV     (SP)+,22(SP)      ;;RESTORE PS OF MAIN FLOW
```

```
4299  023066  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
4300  023070  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
4301  023072  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
4302  023074  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
4303  023076  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
4304  023100  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
4305  023102  000002                          RTI
4306
4307                                  .SBTTL  TRAP DECODER
4308
4309                                  ;;**********************************************************************
4310                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
4311                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
4312                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
4313                                  ;*GO TO THAT ROUTINE.
4314
4315  023104  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
4316  023106  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
4317  023112  005740                          TST     -(R0)           ;;BACKUP BY 2
4318  023114  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
4319  023116  006300                          ASL     R0              ;;POSITION FOR INDEXING
4320  023120  016000  023140                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
4321  023124  000200                          RTS     R0              ;;GO TO ROUTINE
4322
4323
4324                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
4325
4326  023126  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
4327  023130  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
4328  023136  000002                          RTI                     ;;RESTORE THE PSW
4329
4330                                  .SBTTL  TRAP TABLE
4331
4332                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
4333                                  ;*BY THE "TRAP" INSTRUCTION.
4334
4335                                  ;       ROUTINE
4336                                  ;       -------
4337  023140  023126                  $TRPAD: .WORD   $TRAP2
4338  023142  020420                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
4339  023144  020664                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
4340  023146  020640                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
4341  023150  020700                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
4342  023152  021066                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
4343
4344  023154  021700                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
4345
4346  023156  021610                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
4347  023160  022152                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
4348  023162  022242                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
4349  023164  023010                          $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
4350  023166  023046                          $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
4351
4352                                  .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
4353
4354                                  ;;**********************************************************************
```

SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

```
                                        ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                                        ;*UNSIGNED DECIMAL ASCIZ NUMBER.
                                        ;*CALL
                                        ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
                                        ;*      JSR     PC,@#$S82D      ;;CALL
                                        ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK

023170  016637  000002  023220  $S82D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
023176  012746  023220                  MOV     #1$,-(SP)       ;;SET POINTER
023202  004737  023224                  JSR     PC,@#$D82D      ;;CALL DOUBLE LENGTH CONVERT
023206  062716  000005                  ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
023212  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
023216  000207                          RTS     PC              ;;RETURN
023220  000000  000000          1$:     .WORD   0,0

                                        .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

                                        ;;**************************************************************
                                        ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                                        ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                        ;*POSITIVE.
                                        ;*CALL
                                        ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                        ;*      JSR     PC,@#$D82D
                                        ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                        ;;IS ON THE STACK


023224  104412          $D82D:  SAVREG                  ;;SAVE REGISTERS
023226  016602  000002          MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
023232  012700  023404          MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
023236  010066  000002          MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
023242  012201                  MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
023244  012202                  MOV     (R2)+,R2
023246  012737  000012  023322  MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
023254  012704  023334          MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
023260  012705  023336          MOV     #$TNPWR+2,R5
023264  005003          1$:     CLR     R3              ;;CLEAR PARTIAL
023266  161401          2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
023270  005602                  SBC     R2
023272  161502                  SUB     (R5),R2
023274  002402                  BLT     3$              ;;BR IF TEN POWER TO LARGE
023276  005203                  INC     R3              ;;ADD 1 TO PARTIAL
023300  000772                  BR      2$              ;;LOOP
023302  062401          3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
023304  005502                  ADC     R2
023306  062402                  ADD     (R4)+,R2
023310  022525                  CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
023312  052703  000060          BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
023316  110320                  MOVB    R3,(R0)+        ;;SAVE IT
023320  005327                  DEC     (PC)+           ;;DONE?
023322  000000          4$:     .WORD   0
023324  001357                  BNE     1$              ;;BR IF NO
023326  105020                  CLRB    (R0)+           ;;TERMINATOR
023330  104413                  RESREG                  ;;RESTORE REGISTERS
```

DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

```
                                        RTS     PC              ;;RETURN
                                $TNPWR: 145000                  ;;1.0E09
                                        35632
                                        160400                  ;;1.0E08
                                        2765
                                        113200                  ;;1.0E07
                                        230
                                        041100                  ;;1.0E06
                                        17
                                        103240                  ;;1.0E05
                                        1
                                        23420                   ;;1.0E04
                                        0
                                        1750                    ;;1.0E03
                                        0
                                        144                     ;;1.0E02
                                        0
                                        12                      ;;1.0E01
                                        0
                                        1                       ;;1.0E00
                                        0
                                $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING

                                .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

                                ;;**************************************************************
                                ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
                                ;*LEADING NUMBERS.
                                ;*CALL
                                ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
                                ;*      JSR     PC,@#$SUPRS

 023420  010046                 $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
 023422  016600  000004                 MOV     4(SP),R0        ;;PICKUP THE POINTER
 023426  105710                 1$:     TSTB    (R0)            ;;TERMINATEOR?
 023430  001403                         BEQ     2$              ;;BR IF YES
 023432  122720  000060                 CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
 023436  001773                         BEQ     1$              ;;BR IF YES
 023440  005300                 2$:     DEC     R0              ;;BACKUP BY "1"
 023442  010037  023450                 MOV     R0,3$           ;;SAVE FOR TYPING
 023446  104401                         TYPE                    ;;GO TYPE
 023450  000000                 3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
 023452  012600                         MOV     (SP)+,R0        ;;RESTORE R0
 023454  012616                         MOV     (SP)+,(SP)      ;;RESTORE THE STACK
 023456  000207                         RTS     PC              ;;RETURN

                                .SBTTL  RANDOM NUMBER GENERATOR ROUTINE

                                ;;**************************************************************
                                ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
                                ;*WITH A RANGE OF 0 TO 2(+33)-1.
                                ;*CALL:
                                ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
                                ;*      RETURN                  ;;RETURN HERE THE RANDOM
                                ;*                              ;;NUMBER WILL BE IN
```

```
                                        RANDOM NUMBER GENERATOR ROUTINE

4467                                    ;*                              ;;$HINUM,$LONUM
4468
4469  023460                            $RAND:
4470  023460  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4471  023462  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4472  023464  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4473  023466  013700  023560                    MOV     $LONUM,R0       ;;SET R0 WITH LOW
4474  023472  013701  023556                    MOV     $HINUM,R1       ;;SET R1 WITH HIGH
4475  023476  012702  177771                    MOV     #-7,R2          ;;SET SHIFT COUNT
4476  023502  006300                    1$:     ASL     R0              ;;SHIFT R0 LEFT AND
4477  023504  006101                            ROL     R1              ;;ROTATE CARRY INTO R1 AND
4478  023506  005202                            INC     R2              ;;CHECK FOR DONE
4479  023510  001374                            BNE     1$              ;;CONTINUE SHIFT LOOP
4480  023512  063700  023560                    ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
4481  023516  005501                            ADC     R1              ;;PROPOGATE CARRY
4482  023520  063701  023556                    ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
4483  023524  062700  001057                    ADD     #1057,R0        ;;ADD LOW CONSTANT
4484  023530  005501                            ADC     R1              ;;PROPOGATE CARRY
4485  023532  062701  047401                    ADD     #47401,R1       ;;ADD HIGH CONSTANT
4486  023536  010037  023560                    MOV     R0,$LONUM       ;;SAVE R0
4487  023542  010137  023556                    MOV     R1,$HINUM       ;;SAVE R1
4488  023546  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
4489  023550  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
4490  023552  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
4491  023554  000207                            RTS     PC              ;;RETURN
4492  023556  176543                    $HINUM: .WORD   176543
4493  023560  123456                    $LONUM: .WORD   123456
4494
4495                                    .SBTTL  INTEGER DIVIDE ROUTINE
4496
4497                                    ;;*****************************************************************
4498                                    ;*THIS ROUTINE WILL DIVIDE A 32-BIT TWO'S COMPLEMENT INTEGER
4499                                    ;*DIVIDEND BY A 16-BIT TWO'S COMPLEMENT INTEGER DIVISOR GIVING
4500                                    ;*A 16-BIT TWO'S COMPLEMENT INTEGER QUOTIENT AND A 16-BIT REMAINDER.
4501                                    ;*DIVISION WILL BE PERFORMED SO THAT THE REMAINDER IS OF THE
4502                                    ;*SAVE SIGN AS THE DIVIDEND.
4503                                    ;*CALL:
4504                                    ;*      MOV     LOW DIVIDEND,-(SP) ;;THE HIGH DIVIDEND MUST BE < 1/2
4505                                    ;*      MOV     HIGH DIVIDEND,-(SP);    AS LARGE AS THE DIVISOR
4506                                    ;*      MOV     DIVISOR,-(SP)
4507                                    ;*      JSR     PC,$DIV
4508                                    ;*      RETURN                  ;;QUOTIENT & REMAINDER ARE ON THE STACK
4509                                    ;*      "V"=0   IMPLIES NO ERROR
4510                                    ;*      "V"=1   IMPLIES ERROR OCCURRED
4511                                    ;*              "C"=0   DIVIDE OVERFLOW OCCURRED
4512                                    ;*              "C"=1   ATTEMPTED TO DIVIDE BY ZERO
4513                                    ;*
4514                                    ;*
4515                                    ;*      STACK   NO ERROR        OVERFLOW        DIVIDE BY ZERO
4516                                    ;*      -----   --------        --------        -------------
4517                                    ;*      TOP     REMAINDER       ALL ZEROS       ALL ONES
4518                                    ;*      +2      QUOTIENT        ALL ZEROS       ALL ONES
4519
4520  023562                            $DIV:
4521  023562  104400                            TRAP                    ;;PUSH OLD PSW AND PC ON STACK
4522  023564  042716  000017                    BIC     #17,(SP)        ;;STRIP AWAY CONDITION CODES
```

```
4523  023570  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4524  023572  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4525  023574  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4526  023576  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
4527  023600  005046                          CLR     -(SP)           ;;SAVE A PLACE FOR SIGNS
4528  023602  012746  000021                  MOV     #17.,-(SP)      ;;SETUP THE ITERATION COUNTER
4529  023606  016601  000024                  MOV     24(SP),R1       ;;PICKUP THE DIVIDEND
4530  023612  016600  000022                  MOV     22(SP),R0
4531  023616  100005                          BPL     1$              ;;CHECK THE SIGN
4532  023620  105366  000003                  DECB    3(SP)           ;;KEEP TRACK OF THE SIGN
4533  023624  005400                          NEG     R0              ;;AND NEGATE THE ORIGINAL
4534  023626  005401                          NEG     R1              ;;NUMBER
4535  023630  005600                          SBC     R0
4536  023632  016602  000020          1$:     MOV     20(SP),R2       ;;PICKUP THE DIVISOR
4537  023636  002407                          BLT     2$              ;;CHECK THE SIGN
4538  023640  003011                          BGT     3$              ;;DIVISOR OF 0 IS A NO-NO
4539  023642  052766  000003  000014          BIS     #3,14(SP)       ;;SET "V" & "C"
4540  023650  012700  177777                  MOV     #-1,R0          ;;SET REMAINDER TO ALL ONES
4541  023654  000424                          BR      7$              ;;EXIT
4542  023656  005266  000002          2$:     INC     2(SP)           ;;KEEP TRACK OF DIVISORS SIGN
4543  023662  000401                          BR      4$
4544  023664  005402                  3$:     NEG     R2              ;;NEGATE THE ORIGINAL NUMBER
4545  023666  000241                  4$:     CLC                     ;;CLEAR "C"
4546  023670  000405                          BR      6$              ;;START FORMING QUOTIENT
4547  023672  006100                  5$:     ROL     R0              ;;POSITION MSB'S
4548  023674  010003                          MOV     R0,R3           ;;COPY
4549  023676  060203                          ADD     R2,R3           ;;COMPARE DIVIDEND & DIVISOR
4550  023700  103001                          BCC     6$              ;;BR IF DIVIDEND > DIVISOR
4551  023702  010300                          MOV     R3,R0           ;;REMAINDER AFTER THIS LOOP
4552  023704  0061 1                  6$:     ROL     R1              ;;QUOTIENT BIT ENTERS HERE
4553  023706  0053 6                          DEC     (SP)            ;;DONE?
4554  023710  001370                          BNE     5$              ;;BR IF NO
4555  023712  005701                          TST     R1              ;;OVERFLOW?
4556  023714  100005                          BPL     8$              ;;BR IF NO
4557  023716  052766  000002  000014          BIS     #2,14(SP)       ;;SET "V" IN RETURN STATUS WORD
4558  023724  005000                          CLR     R0              ;;SET REMAINDER TO ALL ZEROS
4559  023726  010001                  7$:     MOV     R0,R1           ;;COPY REMAINDER INTO QUOTIENT
4560  023730  005726                  8$:     TST     (SP)+           ;;CLEAR COUNTER FROM STACK
4561  023732  005716                          TST     (SP)            ;;REMAINDER SIGN CORRECTION NEEDED?
4562  023734  002004                          BGE     9$              ;;BR IF NO
4563  023736  005400                          NEG     R0              ;;NEGATE REMAINDER
4564  023740  105066  000001                  CLRB    1(SP)           ;;CLEAR SIGN
4565  023744  005316                          DEC     (SP)            ;;BUT DON'T FORGET QUOTIENT
4566  023746  005726                  9$:     TST     (SP)+           ;;QUOTIENT SIGN CORRECTION NEEDED?
4567  023750  001401                          BEQ     10$             ;;BR IF NO
4568  023752  005401                          NEG     R1              ;;NEGATE QUOTIENT
4569  023754  010166  000020          10$:    MOV     R1,20(SP)       ;;RETURN QUOTIENT AND
4570  023760  010066  000016                  MOV     R0,16(SP)       ;;REMAINDER TO USER
4571  023764  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
4572  023766  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
4573  023770  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
4574  023772  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
4575  023774  012666  000002                  MOV     (SP)+,2(SP)     ;;SETUP TO RETURN CONDITION CODES
4576  024000  000002                          RTI                     ;;RETURN
4577
4578                          .SBTTL  *** PROGRAM SUBROUTINES ***
```

```
4579
4580                                    ;SET "LPTAVL" TO THE PROPER STATE.
4581                                    ; LPTAVL = 0 IF NO LINE PRINTER AVAILABLE
4582                                    ; LPTAVL = 1 IF LINE PRINTER IS AVAILABLE
4583                                    ;CALL
4584                                    ;       JSR     PC,@#LP.AVL
4585                                    ;       RETURN
4586
4587  024002  005037  001230            LP.AVL: CLR     @#LPTAVL        ;START WITH NO PRINTER AVAIABLE
4588  024006  012737  024032  000004            MOV     #1$,@#ERRVEC    ;SETUP THE TIMEOUT VECTOR
4589  024014  005037  000006                    CLR     @#ERRVEC+2
4590  024020  005777  155374                    TST     @#LPS           ;IS THERE A LINE PRINTER?
4591  024024  005237  001230                    INC     @#LPTAVL        ;YES--SET AVAILABLE SWITCH
4592  024030  000401                            BR      2$
4593  024032  022626                    1$:     CMP     (SP)+,(SP)+     ;NO--POP STACK
4594  024034  012737  000006  000004    2$:     MOV     #ERRVEC+2,@#ERRVEC ;RESTORE TIMEOUT VECTOR
4595  024042  000207                            RTS     PC              ;RETURN
4596
4597                                    ;THIS ROUTINE WILL DETERMINE IF THERE IS A CLOCK ON THE SYSTEM
4598                                    ;AND IF THERE IS IT WILL SETUP THE VECTOR AND START THE CLOCK
4599                                    ;"CLKSTA" WILL INDICATE THE CLOCK TYPE
4600                                    ; 0= NO CLOCK
4601                                    ;+1= KW11-P
4602                                    ;-1= KW11-L
4603                                    ;THIS ROUTINE WILL ALSO SETUP "TICKMS" (TIME
4604                                    ;PER CLOCK TICK IN MILLISECONDS) AND "TICKUS"
4605                                    ;(TIME PER CLOCK TICK IN MICROSECONDS) AS
4606                                    ;PER SW00.
4607                                    ;SW00=0 -- 60HZ
4608                                    ;SW00=1 -- 50HZ
4609                                    ;CALL
4610                                    ;       JSR     PC,@#ST.CLK
4611                                    ;       RETURN
4612
4613  024044  010146                    ST.CLK: MOV     R1,-(SP)        ;SAVE R1
4614  024046  012701  000006                    MOV     #ERRVEC+2,R1    ;SAVE AND SETUP TIMEOUT VECTOR
4615  024052  011146                            MOV     (R1),-(SP)
4616  024054  005011                            CLR     (R1)            ;LEVEL 0
4617  024056  014146                            MOV     -(R1),-(SP)
4618  024060  012711  024110                    MOV     #1$,(R1)        ;GO TO 1$ ON TIMEOUT
4619  024064  005037  001244                    CLR     CLKSTA          ;SET CLOCK STATUS TO NO CLOCK
4620  024070  005777  155304                    TST     @PKCS           ;IS THERE A KW11-P?
4621  024074  012737  000001  001244            MOV     #1,CLKSTA       ;YES--SET STATUS TO KW11-P
4622  024102  004737  024212                    JSR     PC,ST.PCLK      ;START THE KW11-P
4623  024106  000414                            BR      3$              ;GO TO EXIT
4624  024110  022626                    1$:     CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
4625  024114  012711  024136                    MOV     #2$,(R1)        ;IF TIMEOUT GO TO 2$
4626  024116  005777  155270                    TST     @LKS            ;IS THERE A KW11-L?
4627  024122  012737  177777  001244            MOV     #-1,CLKSTA      ;YES-- SET STATUS TO KW11-L
4628  024130  004737  024254                    JSR     PC,ST.LCLK      ;START THE KW11-L
4629  024134  000401                            BR      3$              ;EXIT
4630  024136  022626                    2$:     CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
4631  024140  012621                    3$:     MOV     (SP)+,(R1)+     ;RESTORE THE TIMEOUT VECTOR
4632  024142  012621                            MOV     (SP)+,(R1)+
4633  024144  012601                            MOV     (SP)+,R1        ;RESTORE R1
4634  024146  032737  000100  001220            BIT     #SW06,@#C.SWR   ;50HZ OR 60HZ?
```

```
4635  024154  001407                          BEQ     4$              ;BRANCH IF 60
4636  024156  012737  000020  001246          MOV     #20,@#TICKMS    ;SETUP TIME PER
4637  024164  012737  047040  001250          MOV     #20000.,@#TICKUS ;TICK FOR 50HZ
4638  024172  000406                          BR      5$
4639  024174  012737  000016  001246  4$:     MOV     #16,@#TICKMS    ;SETUP TIME PER
4640  024202  012737  040432  001250          MOV     #16666.,@#TICKUS ;TICK FOR 60HZ
4641  024210  000207                  5$:     RTS     PC              ;RETURN
4642
4643  024212                          ST.PCLK:
4644  024212  032737  000040  001220          BIT     #SWOS,@#C.SWR   ;ALLOW SOFTWARE TIMEOUTS?
4645  024220  001014                          BNE     1$              ;NO--BRANCH
4646  024222  012777  024310  155144          MOV     #SRVCLK,@PKV    ;SETUP THE KW11-P VECTOR
4647  024230  012777  000300  155140          MOV     #300,@PKV+2
4648  024236  012777  000001  155136          MOV     #1,@PKB         ;COUNT ONE TICK
4649  024244  012777  000115  155126          MOV     #115,@PKCS      ;"INT.EN.","COUNT DOWN", "MODE 1 (REPEAT)",
4650                                                                  ;"LINE FREQ", AND "RUN"
4651  024252  000207                  1$:     RTS     PC              ;RETURN
4652
4653  024254                          ST.LCLK:
4654  024254  032737  000040  001220          BIT     #SWOS,@#C.SWR   ;ALLOW SOFTWARE TIMEOUTS?
4655  024262  001011                          BNE     1$              ;NO--BRANCH
4656  024264  012777  024310  155114          MOV     #SRVCLK,@LKV    ;SETUP THE KW11-L VECTOR
4657  024272  012777  000300  155110          MOV     #300,@LKV+2
4658  024300  012777  000100  155104          MOV     #100,@LKS       ;START THE KW11-L
4659  024306  000207                  1$:     RTS     PC              ;RETURN
4660
4661  024310  013746  001246  SRVCLK: MOV     @#TICKMS,-(SP)  ;TIME PER TICK IN MILLISECONDS
4662  024314  004737  040704                  JSR     PC,@#RPTMR      ;COUNT THE ELASPED TIME
4663  024320  000002                          RTI                     ;RETURN AFTER INTERRUPT
4664
4665                                  ;THIS ROUTINE SETS UP DEFAULT PARAMETER VALUES WHEN THE PROGRAM IS
4666                                  ;STARTED OR WHEN THE VALUE OF BIT00 IN 'C.SWR' IS CHANGED.
4667                                  ;CALL
4668                                  ;       JSR     PC,LODFLT
4669                                  ;       RETURN
4670
4671  024322                          LODFLT:
4672  024322  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4673  024324  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4674  024326  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4675  024330  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
4676  024332  012737  176777  001234          MOV     #176777,TSTNMS  ;SELECT TESTS 0-10, 12-17
4677  024340  012737  000001  001236          MOV     #1,TSTNMS+2     ;SET SELECT BIT FOR TEST 20
4678  024346  012700  001664                  MOV     #DFLT,R0        ;DEFAULT PARAMETERS POINTER
4679  024352  012701  002330                  MOV     #PRM0,R1        ;TABLE POINTER
4680  024356  010102                          MOV     R1,R2           ;STOP ADDRESS
4681  024360  012021                  1$:     MOV     (R0)+,(R1)+     ;MOVE DEFAULT PARAMETERS INTO
4682  024362  020002                          CMP     R0,R2           ;RUN TIME TABLES ** DONE?
4683  024364  103775                          BLO     1$              ;NO--BRANCH
4684  024366  012700  003504                  MOV     #PAT8,R0        ;PAT0 DEFAULTS TO PATTERN 8
4685  024372  012701  003104                  MOV     #PAT0,R1
4686  024376  012021                  2$:     MOV     (R0)+,(R1)+
4687  024400  020027  003144                  CMP     R0,#PAT9
4688  024404  103774                          BLO     2$
4689  024406  032737  000001  001220          BIT     #BIT00,C.SWR    ;16 BIT MODE ?
4690  024414  001012                          BNE     3$              ;BR IF 18
```

```
4691  024416  012737  000025  001630          MOV     #21.,PRMLMT+22  ;SET 'FS' LIMIT TO 21.
4692  024424  012737  000025  001632          MOV     #21.,PRMLMT+24  ;SET 'LS' LIMIT TO 21.
4693  024432  012737  165000  001352          MOV     #-(256.*22.),TRCKWC   ;WORD COUNT FOR A 16 BIT TRACK
4694  024440  000411                          BR      4$              ;CONTINUE
4695  024442  012737  000023  001630  3$:     MOV     #19.,PRMLMT+22  ;SET 'FS' LIMIT TO 19.
4696  024450  012737  000023  001632          MOV     #19.,PRMLMT+24  ;SET 'LS' LIMIT TO 19.
4697  024456  012737  166000  001352          MOV     #-(256.*20.),TRCKWC   ;WORD COUNT FOR COUNT FOR AN 18 BIT TRACK
4698  024464  012701  001536          4$:     MOV     #PRMPT,R1       ;ADDRESS OF PARAMETER POINTER TABLE
4699  024470  005711                  5$:     TST     (R1)            ;END OF THE TABLE ?
4700  024472  001425                          BEQ     8$              ;BR IF END
4701  024474  032731  002000                  BIT     #BIT10,@(R1)+   ;'LS' SELECTED ?
4702  024500  001773                          BEQ     5$              ;BR IF NOT
4703  024502  016102  177776                  MOV     -2(R1),R2       ;PARAMETER TABLE ADDRESS
4704  024506  011246                          MOV     (R2),-(SP)      ;PARAMETER ALLOCATION BITS
4705  024510  012703  000013                  MOV     #11.,R3         ;NUMBER OF PARAMETERS (MAXIMUM) BEFORE 'LS'
4706  024514  006216                  6$:     ASR     (SP)            ;COUNT THE PARAMETER
4707  024516  103002                          BCC     7$              ;BR IF NOT USED
4708  024520  062702  000002                  ADD     #2,R2           ;INCREMENT THE PARAMETER TABLE ADDRESS
4709  024524  005303                  7$:     DEC     R3              ;COUNT THE PARAMETER
4710  024526  001372                          BNE     6$              ;BR IF NOT THERE YET
4711  024530  005726                          TST     (SP)+           ;CORRECT THE STACK POINTER
4712  024532  021237  001_30                  CMP     (R2),PRMLMT+22  ;IS 'LS' TOO LARGE FOR THE MODE SELECTED ?
4713  024536  101754                          BLOS    5$              ;BR IF NOT
4714  024540  013712  001630                  MOV     PRMLMT+22,(R2)  ;RESET VALUE FOR MODE USED
4715  024544  000751                          BR      5$              ;CONTINUE
4716  024546                          8$:
4717  024546  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
4718  024550  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
4719  024552  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
4720  024554  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
4721  024556  000207                          RTS     PC              ;RETURN
4722
4723                                  ;THIS ROUTINE FILLS THE PARAMETER TABLE THE CURRENT TEST.
4724                                  ;CALL
4725                                  ;       MOV     #TESTNUM,$TSTNM  ;LOAD THE TEST NUMBER
4726                                  ;       JSR     PC,LOOPRM
4727                                  ;       RETURN
4728
4729  024560                          LOOPRM:
4730  024560  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4731  024562  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4732  024564  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
4733  024566  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
4734  024570  005004                          CLR     R4              ;CLEAR R4
4735  024572  113704  001102                  MOVB    $TSTNM,R4       ;GET THE TEST NUMBER
4736  024576  006304                          ASL     R4              ;SETUP TO ADDRESS WORDS
4737  024600  016401  001536                  MOV     PRMPT(R4),R1    ;GET THE TEST'S PARAMETER TABLE ADDRESS
4738  024604  012702  001504                  MOV     #PRM,R2         ;PARAMETER EXECUTION TABLE
4739  024610  005003                          CLR     R3              ;R3 IS USED AS A COUNTER
4740  024612  013704  001254                  MOV     CHKDRV,R4       ;DRIVE'S ADDRESS
4741  024616  012122                          MOV     (R1)+,(R2)+     ;PARAMETER SPECIFIER
4742  024620  006237  001504          1$:     ASR     PRM             ;THIS PARAMETER USED IN THE TEST ?
4743  024624  103002                          BCC     2$              ;BR IF NOT
4744  024626  012122                          MOV     (R1)+,(R2)+     ;LOAD THE VALUE
4745  024630  000401                          BR      3$              ;CONTINUE
4746  024632  005022                  2$:     CLR     (R2)+           ;CLEAR THE UNUSED PARAMETER LOCATION
```

```
4747  024634  005203                  3$:     INC     R3              ;COUNT THE POSITION IN THE OUTPUT TABLE
4748  024636  020327  000014                  CMP     R3,#12.         ;FINISHED ?
4749  024642  001430                          BEQ     6$              ;BR IF YES
4750  024644  020327  000003                  CMP     R3,#3           ;DOING THE CYLINDER ADDRESSES ?
4751  024650  001363                          BNE     1$              ;BR IF NOT
4752  024652  132764  000004  034346          BITB    #BIT02,DRVTYP(R4)  ;RP06 ?
4753  024660  001016                          BNE     5$              ;BR IF IT IS
4754  024662  062703  000002                  ADD     #2,R3           ;COUNT THE BYPASSED PARAMETERS (FC' & LC')
4755  024666  006237  001504                  ASR     PRM             ;SHIFT THE COUNTER
4756  024672  103002                          BCC     4$              ;BR IF FC' IS NOT USED
4757  024674  062701  000002                  ADD     #2,R1           ;MOVE THE INPUT POINTER
4758  024700  006237  001504          4$:     ASR     PRM             ;COUNT THE PARAMETER
4759  024704  103345                          BCC     1$              ;BR IF LC' NOT USED
4760  024706  062701  000002                  ADD     #2,R1           ;MOVE THE INPUT PINTER
4761  024712  000742                          BR      1$              ;KEEP GOING
4762  024714  000741                          BR      1$              ;KEEP GOING
4763  024716  162702  000004          5$:     SUB     #4,R2           ;BACKUP THE OUTPUT POINTER
4764  024722  000736                          BR      1$              ;KEEP GOING
4765  024724                          6$:
4766  024724  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
4767  024726  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
4768  024730  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
4769  024732  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
4770  024734  000207                          RTS     PC              ;RETURN
4771
4772                                  ;THIS ROUTINE LOADS A READ HEADER AND DATA COMMAND OR A SEEK COMMAND
4773                                  ;INTO DPB.B+2 AND DPB.C+2, DEPENDING ON THE STATE OF "CONTROL SWITCH"
4774                                  ;BIT07.
4775                                  ;CALL
4776                                  ;       JSR     PC,@#LDCMD
4777                                  ;       RETURN
4778
4779  024736  032737  000200  001220  LDCMD:  BIT     #SW07,@#C.SWR   ;DO EXPLICIT SEEKS?
4780  024744  001007                          BNE     1$              ;YES--BRANCH
4781  024746  012737  000173  004126          MOV     #READHD,@#DPB.B+2 ;NO--SET UP FOR READ HEADER AND
4782  024754  012737  000173  004146          MOV     #READHD,@#DPB.C+2 ;DATA COMMAND
4783  024762  000406                          BR      2$
4784  024764  012737  000105  004126  1$:     MOV     #SEEK,@#DPB.B+2 ;SETUP FOR SEEK COMMAND
4785  024772  012737  000105  004146          MOV     #SEEK,@#DPB.C+2
4786  025000  000207                  2$:     RTS     PC
4787
4788                                  ;THIS ROUTINE WILL CALL THE RP04/5/6 DRIVER AND THEN WAIT ON THE FUNCTION
4789                                  ;TO COMPLETE. IF AN ERROR OCCURS IT IS REPORTED.
4790                                  ;CALL
4791                                  ;       FILL "DPB" WITH COMMAND INFORMATION
4792                                  ;       JSR     R0,@#CALL.A
4793                                  ;       RETURN
4794
4795  025002  005037  001206          CALL.A: CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
4796  025006  004037  025252                  JSR     R0,@#RP04       ;CALL RP04 DRIVER
4797  025012  004104                          DPB.A
4798  025014  000772                          BR      CALL.A
4799  025016  005737  004122          1$:     TST     @#DPB.A+16      ;DONE?
4800  025022  001775                          BEQ     1$              ;NO--LOOP
4801  025024  100032                          BPL     3$              ;BRANCH IF NO ERROR
4802  025026  012737  025102  001206          MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
```

```
4803  025034  013737  004116  001270          MOV     @#DPB.A+12,@#CYL.DS ;CYLINDER
4804  025042  113737  004115  001274          MOVB    @#DPB.A+11,@#TRK.DS ;TRACK
4805  025050  113737  004114  001272          MOVB    @#DPB.A+10,@#SEC.DS ;SECTOR
4806  025056  012746  004122                  MOV     #DPB.A+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
4807  025062  004737  026216                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
4808  025066  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
4809  025070  104041                          ERROR   41              ;
4810  025072  104042                          ERROR   42              ;PARITY ERROR
4811  025074  104043                          ERROR   43              ;UNSAFE ERROR
4812  025076  104044                          ERROR   44              ;NON-I/O ERROR
4813  025100  104045                          ERROR   45              ;I/O ERROR
4814  025102  013746  004122          2$:     MOV     DPB.A+16,-(SP)  ;STATUS WORD
4815  025106  004737  026156                  JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
4816  025112  000200          3$:     RTS     R0              ;RETURN
4817
4818                                  ;THIS ROUTINE IS THE SAME AS "CALL.A" EXCEPT FOR THE DPB USED AND IF
4819                                  ;THE COMMAND IS A READ HEADER AND DATA THE HEADER (CYLINDER, TRACK,
4820                                  ;AND SECTOR) READ IS CHECKED FOR VALIDITY.
4821                                  ;CALL
4822                                  ;       FILL DPB
4823                                  ;       JSR     R0,@#CALL.B
4824                                  ;       RETURN
4825
4826  025114  005037  001206  CALL.B: CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
4827  025120  004037  025252          JSR     R0,@#RP04       ;CALL RP04 DRIVER
4828  025124  004124                  DPB.B
4829  025126  000772                  BR      CALL.B
4830  025130  005737  004142          1$:     TST     DPB.B+16        ;DONE?
4831  025134  001775                  BEQ     1$              ;NO--BRANCH
4832  025136  100042                  BPL     4$              ;BRANCH IF NO ERROR
4833  025140  012737  025232  001206          MOV     #3$,$ESCAPE     ;ESCAPE TO 3$ ON ERROR
4834  025146  013737  004136  001270          MOV     @#DPB.B+12,@#CYL.DS ;CYLINDER
4835  025154  113737  004135  001274          MOVB    @#DPB.B+11,@#TRK.DS ;TRACK
4836  025162  113737  004134  001272          MOVB    @#DPB.B+10,@#SEC.DS ;SECTOR
4837  025170  012746  004142                  MOV     #DPB.B+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
4838  025174  004737  026216                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
4839  025200  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
4840  025202  104041                          ERROR   41              ;
4841  025204  104042                          ERROR   42              ;PARITY ERROR
4842  025206  104043                          ERROR   43              ;UNSAFE ERROR
4843  025210  104044                          ERROR   44              ;NON-I/O ERROR
4844  025212  005737  004220                  TST     RP.REG+RPER1    ;DRIVE ERROR ?
4845  025216  001404                  BEQ     2$              ;BR IF NOT
4846  025220  032737  177677  004220          BIT     #^C100,RP.REG+RPER1  ;SEE IF ONLY 'HCE' SET
4847  025226  001406                  BEQ     4$              ;BR IF IT IS
4848  025230  104045          2$:     ERROR   45              ;I/O ERROR
4849  025232  013746  004142          3$:     MOV     DPB.B+16,-(SP)  ;STATUS WORD
4850  025236  004737  026156                  JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
4851  025242  000410                  BR      5$              ;CHECK FOR STALL
4852  025244  123727  004126  000173  4$:     CMPB    @#DPB.B+2,#REWOHO ;DOING IMPLIED SEEKS?
4853  025252  001004                  BNE     5$              ;NO--BRANCH
4854  025254  004037  026476                  JSR     R0,@#VERIFY     ;YES--GO CHECK THE DATA
4855  025260  004134                  DPB.B+10
4856  025262  000407                  BR      6$              ;ERROR DURING VERIFY
4857  025264  032737  040000  001220  5$:     BIT     #SW14,@#C.SWR   ;STALL?
4858  025272  001403                  BEQ     6$              ;NO--BRANCH
```

```
 4859  025274  004037  026334            JSR     R0,@#STALL      ;YES--CALL STALL ROUTINE
 4860  025300  001354                    .WORD   STALL1          ;STALL TIME POINTER
 4861  025302  000200            6$:     RTS     R0              ;RETURN
 4862
 4863                                    ;THIS ROUTINE IS THE SAME AS "CALL.B" EXCEPT FOR THE DPB USED.
 4864                                    ;CALL
 4865                                    ;       FILL DPB
 4866                                    ;       JSR     R0,@#CALL.C
 4867                                    ;       RETURN
 4868
 4869  025304  005037  001206    CALL.C: CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
 4870  025310  004037  035252            JSR     R0,@#RP04       ;CALL RP04 DRIVER
 4871  025314  004144                    DPB.C
 4872  025316  000772                    BR      CALL.C
 4873  025320  005737  004162    1$:     TST     @#DPB.C+16      ;DONE?
 4874  025324  001775                    BEQ     1$              ;NO--LOOP
 4875  025326  100042                    BPL     4$              ;YES--BRANCH IF NO ERROR
 4876  025330  012737  025422  001206    MOV     #3$,$ESCAPE     ;ESCAPE TO 3$ ON ERROR
 4877  025336  013737  004156  001270    MOV     @#DPB.C+12,@#CYL.DS ;CYLINDER
 4878  025344  113737  004155  001274    MOVB    @#DPB.C+11,@#TRK.DS ;TRACK
 4879  025352  113737  004154  001272    MOVB    @#DPB.C+10,@#SEC.DS ;SECTOR
 4880  025360  012746  004162            MOV     #DPB.C+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
 4881  025364  004737  026216            JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
 4882  025370  062607                    ADD     (SP)+,PC        ;REPORT PROPER ERROR
 4883  025372  104041                    ERROR   41              ;
 4884  025374  104042                    ERROR   42              ;PARITY ERROR
 4885  025376  104043                    ERROR   43              ;UNSAFE ERROR
 4886  025400  104044                    ERROR   44              ;NON-I/O ERROR
 4887  025402  005737  004220            TST     RP.REG+RPER1    ;DRIVE ERROR ?
 4888  025406  001404                    BEQ     2$              ;BR IF NOT
 4889  025410  032737  177677  004220    BIT     #^C100,RP.REG+RPER1  ;SEE IF ONLY 'HCE' SET
 4890  025416  001406                    BEQ     4$              ;BR IF IT IS
 4891  025420  104045            2$:     ERROR   45              ;I/O ERROR
 4892  025422  013746  004162    3$:     MOV     DPB.C+16,-(SP)  ;STATUS WORD
 4893  025426  004737  026156            JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
 4894  025432  000410                    BR      5$
 4895  025434  123727  004146  000173 4$:CMPB    @#DPB.C+2,#READHD ;DOING IMPLIED SEEK?
 4896  025442  001004                    BNE     5$              ;NO--EXIT
 4897  025444  004037  026476            JSR     R0,@#VERIFY     ;YES--CHECK THE DATA
 4898  025450  004154                    DPB.C+10
 4899  025452  000407                    BR      6$              ;ERROR DURING VERIFY
 4900  025454  032737  040000  001220 5$:BIT     #SW14,@#C.SWR   ;STALL?
 4901  025462  001403                    BEQ     6$              ;NO--BRANCH
 4902  025464  004037  026334            JSR     R0,@#STALL      ;YES--CALL STALL ROUTINE
 4903  025470  001354                    .WORD   STALL1          ;STALL TIME POINTER
 4904  025472  000200            6$:     RTS     R0
 4905
 4906
 4907                                    ;THIS ROUTINE IS THE SAME AS "CALL.A" EXCEPT FOR THE DPB USED AND
 4908                                    ;ON AN ERROR LOCATION "ERR.CT" IS EXAMINED. IF ERR.CT IS EQUAL TO
 4909                                    ;$ERFLG EXIT IS TO THE NEXT TEST.
 4910                                    ;CALL
 4911                                    ;       FILL DPB
 4912                                    ;       JSR     R0,@#DRVCAL
 4913                                    ;       RETURN
 4914
```

```
 4915  025474  005037  001206          DRVCAL: CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
 4916  025500  005037  001334                  CLR     @#WCEFLG        ;CLEAR WRITE CHECK ERROR FLAG
 4917  025504  004037  035252                  JSR     R0,@#RP04       ;CALL RP04 DRIVER
 4918  025510  004164                          DTADPB
 4919  025512  000770                          BR      DRVCAL
 4920  025514  005737  004202          DRVCL1: TST     @#DTADPB+16     ;DONE
 4921  025520  001775                          BEQ     DRVCL1          ;NO--LOOP
 4922  025522  100402                          BMI     1$              ;BR IF ERRORS
 4923  025524  000137  026136                  JMP     10$             ;NO ERRORS
 4924  025530                          1$:
 4925  025530  012737  025604  001206          MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
 4926  025536  013737  004176  001270          MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER
 4927  025544  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
 4928  025552  113737  004174  001272          MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
 4929  025560  012746  004202                  MOV     #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
 4930  025564  004737  026216                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
 4931  025570  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
 4932  025572  104041                          ERROR   41              ;
 4933  025574  104042                          ERROR   42              ;PARITY ERROR
 4934  025576  104043                          ERROR   43              ;UNSAFE ERROR
 4935  025600  104044                          ERROR   44              ;NON-I/O ERROR
 4936  025602  104045                          ERROR   45              ;I/O ERROR
 4937  025604  122737  000020  001102  2$:     CMPB    #20,@#$TSTNM    ;TEST 20?
 4938  025612  001137                          BNE     8$              ;NO--BRANCH
 4939  025614  013746  004202                  MOV     DTADPB+16,-(SP) ;STATUS WORD
 4940  025620  004737  026156                  JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
 4941  025624  122737  000151  004166          CMPB    #WRCKD,@#DTADPB+2 ;DOING A WRITE CHECK?
 4942  025632  001133                          BNE     12$             ;NO--BRANCH
 4943  025634  032737  040000  004214          BIT     #BIT14,@#RP.REG+10 ;IS "WCE"=1?
 4944  025642  001527                          BEQ     12$             ;NO--BRANCH
 4945  025644  032777  000020  153266          BIT     #$HO4,@$HR      ;INHIBIT WRITES?
 4946  025652  001123                          BNE     12$             ;YES--BRANCH
 4947  025654  112737  000161  004166          MOVB    #WRITE,@#DTADPB+2 ;SETUP FOR A WRITE
 4950  025662  005037  001206                  CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
 4951  025666  004037  035252                  JSR     R0,@#RP04       ;DO THE WRITE
 4952  025672  004164                          DTADPB
 4953  025674  000240                          NOP
 4954  025676  005737  004202          3$:     TST     @#DTADPB+16     ;DONE?
 4955  025702  001775                          BEQ     3$              ;NO--LOOP
 4956  025704  100126                          BPL     4$              ;YES--BRANCH IF NO ERROR
 4957  025706  012737  026112  001206          MOV     #8$,$ESCAPE     ;;ESCAPE TO 8$ ON ERROR
 4960  025714  013737  004176  001270          MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER
 4961  025722  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
 4962  025730  113737  004174  001272          MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
 4963  025736  012746  004202                  MOV     #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
 4964  025742  004737  026216                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
 4965  025746  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
 4966  025750  104041                          ERROR   41              ;
 4967  025752  104042                          ERROR   42              ;PARITY ERROR
 4970  025754  104043                          ERROR   43              ;UNSAFE ERROR
 4971  025756  104044                          ERROR   44              ;NON-I/O ERROR
 4972  025760  104045                          ERROR   45              ;I/O ERROR
 4973  025762  112737  000151  004166  4$:     MOVB    #WRCKD,@#DTADPB+2 ;COMMAND=WRITE CHECK DATA
 4974  025770  004037  035252                  JSR     R0,@#RP04       ;DO THE WRITE CHECK
 4975  025774  004164                          DTADPB
 4976  025776  000240                          NOP
```

```
4971  026000  005737  004202          5$:     TST     @#DTADPB+16     ;DONE?
4972  026004  001775                  BEQ     5$              ;NO--LOOP
4973  026006  100410                  BMI     7$              ;YES--BRANCH IF ERROR
4974  026010  004037  035252          JSR     R0,@#RP04       ;DO A 2ND WRITE CHECK
4975  026014  004164                  DTADPB
4976  026016  000240                  NOP
4977  026020  005737  004202          6$:     TST     @#DTADPB+16     ;DONE?
4978  026024  001775                  BEQ     6$              ;NO--LOOP
4979  026026  100043                  BPL     10$             ;YES--BRANCH IF NO ERROR
4980  026030  012737  000001  001334  7$:     MOV     #1,@#WCEFLG     ;SET THE WRITE CHECK ERROR FLAG
4981  026036  012737  026112  001206          MOV     #8$,$ESCAPE     ;;ESCAPE TO 8$ ON ERROR
4982  026044  013737  004176  001270          MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER
4983  026052  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
4984  026060  113737  004174  001272          MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
4985  026066  012746  004202                  MOV     #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
4986  026072  004737  026216                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
4987  026076  062607                  ADD     (SP)+,PC        ;REPORT PROPER ERROR
4988  026100  104041                  ERROR   41              ;
4989  026102  104042                  ERROR   42              ;PARITY ERROR
4990  026104  104043                  ERROR   43              ;UNSAFE ERROR
4991  026106  104044                  ERROR   44              ;NON-I/O ERROR
4992  026110  104046                  ERROR   46              ;FATAL WRITE CHECK
4993  026112  013746  004202          8$:     MOV     DTADPB+16,-(SP) ;STATUS WORD
4994  026116  004737  026156                  JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
4995  026122  123737  001364  001103  12$:    CMPB    @#ERR.CT,@#SERFLG ;GO TO NEXT TEST?
4996  026130  101002                  BHI     10$             ;NO--BRANCH
4997  026132  013700  001252          9$:     MOV     @#BYPASS,R0     ;YES--GET EXIT ADDRESS
4998  026136  032737  040000  001220  10$:    BIT     #SW14,@#C.SWR   ;STALL?
4999  026144  001403                  BEQ     11$             ;NO--BRANCH
5000  026146  004037  026334                  JSR     R0,@#STALL      ;YES--CALL STALL ROUTINE
5001  026152  001356                  .WORD   STALL2          ;STALL TIME POINTER
5002  026154  000200                  11$:    RTS     R0

5003                                  ;THIS SUBROUTINE CHECK FOR LOOP, ABORT, OR CONTINUE SWITCHES AFTER
5004                                  ;ERRORS 41, 42, 43, 44, 45, AND 46.
5005                                  ;CALL
5006                                  ;       MOV     DTA+16,-(SP)    ;STATUS WORD FROM DPB IN USE
5007                                  ;       JSR     PC,LOP.CK
5008                                  ;       RETURN
5009
5010
5011  026156  032777  001000  152754  LOP.CK: BIT     #SW9,@SWR       ;LOOP ON ERROR
5012  026164  001402                  BEQ     1$              ;BR IF NOT
5013  026166  000177  152716                  JMP     @$LPERR         ;START AT THE LOOP ADDRESS
5014  026172  005037  001206          1$:     CLR     $ESCAPE         ;CLEAR ERROR ESCAPE FLAG
5015  026176  032766  072006  000002          BIT     #BIT14!BIT13!BIT12!BIT10!BIT02!BIT01,2(SP)  ;CHECK ERROR TYPE
5016  026204  001402                  BEQ     2$              ;BR IF DRIVE NOT OFFLINE, UNLOADED, OR
5017                                                          ;PERSISTENT UNSAFE OR FATAL MASSBUS PARITY
5018  026206  000137  017356                  JMP     $EOP            ;TERMINATE DRIVE
5019  026212  012616                  2$:     MOV     (SP)+,(SP)      ;ADJUST RETURN ADDRESS
5020  026214  000207                  RTS     PC
5021
5022                                  ;THIS ROUTINE FORMS AN INDEX THAT WILL BE USED TO DISPATCH
5023                                  ;TO THE PROPER ERROR CALL.  THE INDEX IS FORMED BY EXAMINING
5024                                  ;THE STATUS/ERROR INDICATOR OF THE APPLICABLE DPB.
5025                                  ;INDEX           STATUS/ERROR
5026                                  ;-----           ------------
```

```
                                *** PROGRAM SUBROUTINES ***

                                ;     0 BIT14!BIT13!BIT08!BIT01
                                ;     2 BIT11!BIT10!BIT02
                                ;     4 BIT12!BIT04
                                ;     6 BIT05!BIT03!<BIT09 & COMMAND=NON-I/O>
                                ;    10 BIT06!<BIT09 & COMMAND=I/O>
                                ;CALL
                                ;       JSR     @#OPB+16,-(SP)  ;ADDRESS OF STATUS/ERROR INDICATOR
                                ;       JSR     PC,@#ERINDX     ;FORM INDEX
                                ;       RETURN                  ;INDEX IS ON THE STACK

026216  010046          ERINDX: MOV     R0,-(SP)        ;SAVE R0
026220  010146                  MOV     R1,-(SP)        ;SAVE R1
026222  016600  000006          MOV     6(SP),R0        ;GET STATUS/ERROR INDICATOR POINTER
026226  011037  001260          MOV     (R0),@#SVSTAT   ;SAVE THE STATUS/ERROR INDICATOR
026232  005001                  CLR     R1              ;START INDEX AT ZERO
026234  032710                  BIT     (PC)+,(R0)      ;FORM INDEX OF 0?
026236  020402                  .WORD   BIT13!BIT08!BIT01
026240  001027                  BNE     5$              ;YES--BRANCH
026242  032710                  BIT     (PC)+,(R0)      ;FORM PARITY ERROR OR PORT REQUEST INDEX (2)?
026244  006004                  .WORD   BIT11!BIT10!BIT02
026246  001023                  BNE     4$              ;YES--BRANCH
026250  032710                  BIT     (PC)+,(R0)      ;FORM UNSAFE INDEX (4)?
026252  050020                  .WORD   BIT14!BIT12!BIT04
026254  001017                  BNE     3$              ;YES--BRANCH
026256  032710                  BIT     (PC)+,(R0)      ;FORM NON-I/O ERROR INDEX (6)?
026260  000050                  .WORD   BIT05!BIT03
026262  001013                  BNE     2$              ;YES--BRANCH
026264  032710                  BIT     (PC)+,(R0)      ;FORM I/O ERROR INDEX (10)?
026266  000100                  .WORD   BIT06
026270  001007                  BNE     1$              ;YES--BRANCH
026272  032710                  BIT     (PC)+,(R0)      ;SOFTWARE TIMEOUT?
026274  001000                  .WORD   BIT09
026276  001410                  BEQ     5$              ;NO--FORM INDEX OF 0
026300  122760  000150  177762  CMPB    #150,-16(R0)    ;YES--I/O?
026306  003001                  BGT     2$              ;NO--BRANCH
026310  005201          1$:     INC     R1              ;INDEX=10---ERROR=45 OR 46
026312  005201          2$:     INC     R1              ;INDEX=6---ERROR=44
026314  005201          3$:     INC     R1              ;INDEX=4---ERROR=43
026316  005201          4$:     INC     R1              ;INDEX=2---ERROR=42
026320  006301          5$:     ASL     R1              ;INDEX=0---ERROR=41
026322  010166  000006          MOV     R1,6(SP)        ;RETURN INDEX TO USER
026326  012601                  MOV     (SP)+,R1        ;RESTORE R1
026330  012600                  MOV     (SP)+,R0        ;RESTORE R0
026332  000207                  RTS     PC              ;RETURN FROM CALL

                                ;THIS ROUTINE WILL PROVIDE A STALL IN MILLISECONDS FOR A SPECIFIC
                                ;AMOUNT OF TIME IF BIT13 OF C.SWR = 0 OR A RANDOM AMOUNT OF TIME
                                ;IF BIT 13 OF C.SWR = 1.
                                ;STALL1 CONTAINS SPECIFIED TIME FOR TESTS 0 - 7, AND STALL2
                                ;CONTAINS THE TIME FOR TESTS 16-21.
                                ;CALL
                                ;       JSR     R0,@#STALL
                                ;       TIME POINTER            ;WHERE TO FIND THE STALL TIME

026334  013046          STALL:  MOV     @(R0)+,-(SP)    ;PICKUP STALL TIME
026336  032737  020000  001220  BIT     #SW13,@#C.SWR   ;USE A RANDOM TIME?
```

```
                                        *** PROGRAM SUBROUTINES ***

5083  026344  001406                            BEQ     1$              ;NO--BRANCH
5084  026346  004737  023560                    JSR     PC,@#$RAND      ;YES--FORM RANDOM NUMBER
5085  026352  013716  023560                    MOV     @#$LONUM,(SP)   ;AND USE IT FOR THE STALL TIME
5086  026356  042716  177700                    BIC     #^C77,(SP)      ;BUT NEVER > 64 MILLISECONDS
5087  026362  005046                    1$:     CLR     -(SP)           ;CLEAR TEMP. LOCATION
5088  026364  162766  000001  000002    2$:     SUB     #1,2(SP)        ;MORE STALL REQUIRED?
5089  026372  103407                            BLO     4$              ;NO--BRANCH
5090  026374  012716  000144                    MOV     #100.,(SP)      ;STALL FOR ABOUT 1 MILLISECOND
5091  026400  005700                    3$:     TST     R0              ;NOP TO KILL TIME
5092  026402  005366  000000                    DEC     0(SP)           ;COUNT
5093  026406  001374                            BNE     3$              ;LOOP IF MORE COUNTS NEEDED
5094  026410  000765                            BR      2$
5095  026412  022626                    4$:     CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
5096  026414  000200                            RTS     R0              ;EXIT
5097
5098
5099                                    ;ROUTINE TO PROVIDE A 2 MS STALL AFTER A SEEK OPERATION IN THE SEEK TIMING
5100                                    ;TESTS.  THIS STALL IS REQUIRED TO COMPENSATE FOR THE 'ACCESS READY' DELAY
5101                                    ;IN THE RP04.  THIS STALL TIME IS NOT INCLUDED IN THE CALCULATED SEEK TIMES.
5102                                    ;CALL
5103                                    ;       JSR     PC,@#TWOMS
5104                                    ;       RETURN
5105
5106  026416  013746  177776            TWOMS:  MOV     @#PS,-(SP)      ;SAVE THE PRESENT PROCESSOR STATUS
5107  026422  012737  000240  177776            MOV     #(5*32.),@#PS   ;SET THE PROCESSOR PRIORITY TO 5
5108  026430  017746  152740                    MOV     @PKV,-(SP)      ;SAVE THE OLD CLOCK VECTOR ADDRESS
5109  026434  012777  026460  152732            MOV     #1$,@PKV        ;SETUP NEW VECTOR ADDRESS
5110  026442  012777  000310  152732            MOV     #200.,@PKB      ;LOAD THE CLOCK BUFFER
5111  026450  012777  000101  152722            MOV     #101,@PKCS      ;START THE CLOCK
5112  026456  000001                            WAIT                    ;WAIT FOR 2 MS
5113  026460  062706  000004            1$:     ADD     #4,SP           ;INCREMENT STACK FOR RETURN
5114  026464  012677  152704                    MOV     (SP)+,@PKV      ;RESTORE OLD CLOCK VECTOR
5115  026470  012637  177776                    MOV     (SP)+,@#PS      ;RESTORE THE OLD PROCESSOR STATUS
5116  026474  000207                            RTS     PC              ;RETURN
5117
5118                                    ;ROUTINE TO SOFTWARE COMPARE HEADER ON IMPLIED SEEKS
5119                                    ;CALL
5120                                    ;       JSR     R0,@#VERIFY
5121                                    ;       ADR POINTER             ;ADDRESS OF DPB+10 (SECTOR NUMBER)
5122                                    ;       RETURN
5123
5124  026476  010146                    VERIFY: MOV     R1,-(SP)        ;SAVE R1
5125  026500  012001                            MOV     (R0)+,R1        ;GET ADDRESS OF DPB+10
5126  026502  042737  010000  047662            BIC     #FMT22,@#BUFFER ;STRIP FORMAT BIT FROM CYLINDER NUMBER
5127  026510  023761  047662  000002            CMP     @#BUFFER,2(R1)  ;CYLINDER NUMBER OK?
5128  026516  001003                            BNE     1$              ;NO--BRANCH
5129  026520  023711  047664                    CMP     @#BUFFER+2,(R1) ;YES--HOW ABOUT TRACK/SECTOR?
5130  026524  001441                            BEQ     3$              ;BRANCH IF GOOD
5131  026526  013737  047662  001262    1$:     MOV     @#BUFFER,@#CYL.RD ;SAVE THE EXPECTED AND THE
5132  026534  113737  047665  001264            MOVB    @#BUFFER+3,@#TRK.RD ;RECIEVED CYLINDER, TRACK,
5133  026542  113737  047664  001266            MOVB    @#BUFFER+2,@#SEC.RD ;AND SECTOR
5134  026550  112137  001272                    MOVB    (R1)+,@#SEC.DS
5135  026554  112137  001274                    MOVB    (R1)+,@#TRK.DS
5136  026560  011137  001270                    MOV     (R1),@#CYL.DS
5137  026564  012737  026576  001206            MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
5138  026572  005740                            TST     -(R0)           ;MAKE IT TEST PC+4
```

```
5139  026574  104012                          ERROR   12              ;REPORT THE ERROR
5140  026576  012737  000107  004106  2$:     MOV     #RECAL,DPB.A+2  ;LOAD RECALIBRATE ORDER CODE
5141  026604  004037  025002                  JSR     R0,@#CALL.A     ;GO EXECUTE THE COMMAND
5142  026610  005037  001206                  CLR     $ESCAPE         ;CLEAR ERROR ESCAPE FLAG
5143  026614  032777  001000  152316          BIT     #SW9,@SWR       ;LOOP ON ERROR ?
5144  026622  001404                          BEQ     4$              ;BR IF NOT
5145  026624  000177  152260                  JMP     @$LPERR         ;RETURN TO ERROR LOOP ADDRESS
5146  026630  062700  000002          3$:     ADD     #2,R0           ;INCREMENT RETURN ADDRESS
5147  026634  012601                  4$:     MOV     (SP)+,R1        ;RESTORE R1
5148  026636  000200                          RTS     R0              ;EXIT
5149
5150                                  ;THIS ROUTINE WILL PERFORM A "MASSBUS" INIT. FOLLOWED BY
5151                                  ;A "RECALIBRATE" ON THE DRIVE UNDER TEST.
5152                                  ;NOTE: THIS ROUTINE DESTROYS R1 AND R4
5153                                  ;CALL
5154                                  ;       JSR     R0,SRCH00       ;DO A MASSBUS INIT. AND RECAL
5155                                  ;       RETURN1                 ;RETURN HERE IF NO ERROR
5156                                  ;       RETURN2                 ;RETURN HERE ON ERROR
5157
5158  026640  005001                  SRCH00: CLR     R1              ;INCASE OF ERROR (TYPTIM)
5159  026642  005037  177776                  CLR     @#PS
5160  026646  012777  037346  005612          MOV     #ISR,@RPVEC     ;SETUP INTERRUPT VECTOR
5161  026654  013704  034464                  MOV     @#RPADR,R4      ;PICKUP ADDRESS OF RPCS1
5162  026660  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;MASSBUS INIT.
5163  026666  005037  004174                  CLR     @#DTADPB+10     ;TRACK=0; SECTOR=0
5164  026672  005037  004176                  CLR     @#DTADPB+12     ;CYLINDER =0
5165  026676  012737  000107  004166          MOV     #RECAL,@#DTADPB+2 ;COMMAND = RECALIBRATE
5166  026704  005037  001206                  CLR     @#$ESCAPE       ;NO ESCAPE ADDRESS
5167  026710  004037  035252                  JSR     R0,@#RP04       ;CALL THE DRIVER
5168  026714  004164                          DTADPB                  ;DPB POINTER
5169  026716  000440                          BR      4$              ;QUEUE IS FULL
5170  026720  005737  004202          1$:     TST     DTADPB+16       ;WAIT ON DONE
5171  026724  001775                          BEQ     1$
5172  026726  100030                          BPL     3$              ;TAKE NORMAL EXIT IF NO ERROR
5173  026730  012737  027004  001206          MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
5174  026736  013737  004176  001270          MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER
5175  026744  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
5176  026752  113737  004174  001272          MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
5177  026760  012746  004202                  MOV     #DTADPB+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
5178  026764  004737  026216                  JSR     PC,@#ERINDX     ;FORM DISPATCH INDEX
5179  026770  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
5180  026772  104041                          ERROR   41              ;
5181  026774  104042                          ERROR   42              ;PARITY ERROR
5182  026776  104043                          ERROR   43              ;UNSAFE ERROR
5183  027000  104044                          ERROR   44              ;NON-I/O ERROR
5184  027002  104045                          ERROR   45              ;I/O ERROR
5185  027004  005720                  2$:     TST     (R0)+           ;ADJUST FOR ERROR EXIT
5186  027006  000404                          BR      4$              ;GO TO THE EXIT
5187  027010  005064  000006          3$:     CLR     RPDA(R4)        ;TRACK AND SECTOR = 0
5188  027014  005064  000034                  CLR     RPCA(R4)        ;CYLINDER = 0
5189  027020  000200                  4$:     RTS     R0              ;RETURN
5190
5191                                  ;THIS IS AN RTI WHICH IS USED BY THE TIMING TESTS & THE SERVO SETTLE DOWN TEST
5192
5193  027022  000002                  DORTI:  RTI                     ;RETURN FROM INTERRUPT
5194
```

```
                                        *** PROGRAM SUBROUTINES ***

                                        ;THIS ROUTINE WILL INITIALIZE THE TIMERS USED BY THE "TIMING ROUTI ES
                                        ;CALL
                                        ;       JSR     PC,@#STRTMR
                                        ;

005195                                  STRTMR: SAVREG                          ;SAVE R0-R5
005196  027024  104412                          MOV     #TIM.UP,R0              ;START AT TIM.UP (MINIMUM)
005197  027026  012700  001276                  MOV     #TIM.PT,R1              ;STOP AT TIM.PT
005198  027032  012701  001332          1$:     CLR     (R0)+                   ;CLEAR
005199  027036  005020                          CMP     R0,R1                   ;DONE?
005200  027040  020001                          BLO     1$                      ;NO--BRANCH
005201  027042  103775                          MOV     #BUFFER,(R0)            ;SETUP POINTER
005202  027044  012710  047662                  MOV     #^CBIT15,@#TIM.UP       ;SET MINIMUM TIME TO MAXIMUM
005203  027050  012737  077777  001276          MOV     #^CBIT15,@#TIM.DN       ;POSITIVE NUMBER
005204  027056  012737  077777  001314          RESREG                          ;RESTORE R0-R5
005205  027064  104413                          RTS     PC                      ;RETURN
005206  027066  000207

                                        ;THIS ROUTINE WILL ADD THE ELAPSED TIME TO THE AVERAGE COUNTER AND
                                        ;MAINTAIN THE MINIMUM AND MAXIMUM TIMES.
                                        ;NOTE: THIS ROUTINE DESTROYS R2
                                        ;CALL
                                        ;       MOV     #TP,R3                  ;PARAMETER POINTER
                                        ;       MOV     FLAG,R5                 ;FLAG=0=COUNT UP
                                        ;                                       ;FLAG=-1=COUNT DOWN
                                        ;       JSR     PC,@#COUNT
                                        ;       RETURN

005222  027070  012702  001276          COUNT:  MOV     #TIM.UP,R2              ;PICKUP THE "UP" POINTER
005223  027074  005705                          TST     R5                      ;USE IT?
005224  027076  001402                          BEQ     1$                      ;YES--BRANCH
005225  027100  012702  001314                  MOV     #TIM.DN,R2              ;NO--PICKUP "DOWN" POINTER
005226  027104  027722  152274          1$:     CMP     @PKC,(R2)+              ;LESS THAN PREVIOUS LOW?
005227  027110  002003                          BGE     2$                      ;NO--BRANCH
005228  027112  017762  152266  177776          MOV     @PKC,-2(R2)             ;YES--SAVE IT
005229  027120  027763  152260  000004  2$:     CMP     @PKC,4(R3)              ;LESS THAN THE LOW LIMIT?
005230  027126  002001                          BGE     3$                      ;NO--BRANCH
005231  027130  005212                          INC     (R2)                    ;YES--COUNT IT
005232  027132  005722                  3$:     TST     (R2)+                   ;ADVANCE THE POINTER
005233  027134  027722  152244                  CMP     @PKC,(R2)+              ;GREATER THAN PREVIOUS HIGH?
005234  027140  003403                          BLE     4$                      ;NO--BRANCH
005235  027142  017762  152236  177776          MOV     @PKC,-2(R2)             ;YES--SAVE IT
005236  027150  027763  152230  000006  4$:     CMP     @PKC,6(R3)              ;GREATER THAN THE HIGH LIMIT?
005237  027156  003401                          BLE     5$                      ;NO--BRANCH
005238  027160  005212                          INC     (R2)                    ;YES--COUNT IT
005239  027162  005722                  5$:     TST     (R2)+                   ;ADVANCE THE POINTER
005240  027164  067722  152214                  ADD     @PKC,(R2)+              ;ADD THIS COUNT TO THE TOTAL
005241  027170  005522                          ADC     (R2)+
005242  027172  005212                          INC     (R2)                    ;COUNT THIS READING
005243  027174  022737  056152  001332          CMP     #BUFFER+<4*814.>,@#TIM.PT ;SAVE THIS COUNT?
005244  027202  101406                          BLOS    6$                      ;NO--BRANCH
005245  027204  017777  152174  152120          MOV     @PKC,@TIM.PT            ;YES--WELL SAVE IT THEN
005246  027212  062737  000002  001332          ADD     #2,@#TIM.PT             ;ADVANCE THE POINTER
005247  027220  000207                  6$:     RTS     PC                      ;RETURN
005248
005249                                  ;THIS ROUTINE IS USED TO TYPE THE MINIMUM,
005250                                  ;MAXIMUM, AND AVERAGE TIMES FOR THE TIM'NG TESTS
```

```
                                        ;IT WILL ALSO CHECK THE TIMES TO ENSURE
                                        ;THEY ARE WITHIN TOLERANCE AND IF NOT FLAG THE BAD TIMES.
                                        ;NOTE: THIS ROUTINE DESTROYS R2-R5
                                        ;CALL
                                        ;       JSR     R0,@#TYPTIM     ;GO REPORT THE TIMES
                                        ;       TABLE                   ;POINT TO THE PROPER TABLE
                                        ;       RETURN
                                        ;
                                        ;TABLE:MSGADR1                  ;ADDRESS OF ASCIZ MESSAGE NUMBER 1
                                        ;       MSGADR2                 ;ADDRESS OF ASCIZ MESSAGE NUMBER 2
                                        ;       MIN.ALLOWED             ;MINIMUM TIME ALLOWED
                                        ;       MAX.ALLOWED             ;MAXIMUM TIME ALLOWED

027222  012002                  TYPTIM: MOV     (R0)+,R2        ;PICKUP THE TABLE POINTER
027224  032777  000100  151706          BIT     #SW06,@SWR      ;INHIBIT TIME REPORTS?
027232  001145                          BNE     7$              ;YES--BRANCH
027234  012237  027254                  MOV     (R2)+,2$        ;ADDRESS OF MESSAGE NUMBER 1
027240  012205                          MOV     (R2)+,R5        ;ADDRESS OF MESSAGE NUMBER 2
027242  012203                          MOV     (R2)+,R3        ;PICKUP THE LOW LIMIT
027244  011202                          MOV     (R2),R2         ;AND THE HIGH LIMIT
027246  012704  001276                  MOV     #TIM.UP,R4      ;PARAMETER POINTER
027252  104401                  1$:     TYPE                    ;TYPE THE MESSAGE
027254  000000                  2$:     .WORD   0               ;ASCIZ MESSAGE POINTER GOES HERE
027256  005764  000014                  TST     14(R4)          ;DID ANY COUNTS OCCUR?
027262  001527                          BEQ     6$              ;NO--BRANCH
027264  104401  043650                  TYPE    ,MSGMIN         ;"MIN="
027270  012446                          MOV     (R4)+,-(SP)             ;PUT (R4)+ ON THE STACK
027272  004737  023170                  JSR     PC,@#$SB2D      ;CHANGE (R4)+ TO DECIMAL ASCIZ
027276  004737  023420                  JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
027302  104401  043675                  TYPE    ,MSGOUS         ;"0 US"
027306  005724                          TST     (R4)+           ;ANY SEEKS BELOW THE LOW LIMIT
027310  001421                          BEQ     3$              ;NO--BRANCH
027312  104401  044010                  TYPE    ,MSG.SP         ;" "
027316  016446  177776                  MOV     -2(R4),-(SP)            ;PUT -2(R4) ON THE STACK
027322  004737  023170                  JSR     PC,@#$SB2D      ;CHANGE -2(R4) TO DECIMAL ASCIZ
027326  004737  023420                  JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
027332  104401  043702                  TYPE    ,MBELOW         ;"BELOW THE MINIMUM OF"
027336  010346                          MOV     R3,-(SP)                ;PUT R3 ON THE STACK
027340  004737  023170                  JSR     PC,@#$SB2D      ;CHANGE R3 TO DECIMAL ASCIZ
027344  004737  023420                  JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
027350  104401  043675                  TYPE    ,MSGOUS
027354  104401  043657          3$:     TYPE    ,MSGMAX         ;"MAX="
027360  012446                          MOV     (R4)+,-(SP)             ;PUT (R4)+ ON THE STACK
027362  004737  023170                  JSR     PC,@#$SB2D      ;CHANGE (R4)+ TO DECIMAL ASCIZ
027366  004737  023420                  JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
027372  104401  043675                  TYPE    ,MSGOUS
027376  005724                          TST     (R4)+           ;ANY SEEKS ABOVE THE HIGH LIMIT
027400  000421                          BR      4$              ;NO--BRANCH
027402  104401  044010                  TYPE    ,MSG.SP         ;YES--REPORT HOW MANY
027406  016446  177776                  MOV     -2(R4),-(SP)            ;PUT -2(R4) ON THE STACK
027412  004737  023170                  JSR     PC,@#$SB2D      ;CHANGE -2(R4) TO DECIMAL ASCIZ
027416  004737  023420                  JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
027422  104401  043731                  TYPE    ,MABOVE         ;"ABOVE THE MAXIMUM OF"
027426  010246                          MOV     R2,-(SP)                ;PUT R2 ON THE STACK
027430  004737  023170                  JSR     PC,@#$SB2D      ;CHANGE R2 TO DECIMAL ASCIZ
027434  004737  023420                  JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
```

```
                                        *** PROGRAM SUBROUTINES ***

5307  027440  104401  043675                    TYPE    ,MSGOUS
5308  027444  104401  043666    4$:             TYPE    ,MSGAVG         ;"AVG="
5309  027450  012446                            MOV     (R4)+,-(SP)     ;FORM THE AVERAGE
5310  027452  012446                            MOV     (R4)+,-(SP)
5311  027454  012446                            MOV     (R4)+,-(SP)
5312  027456  004737  023562                    JSR     PC,@#$DIV
5313  027462  006126                            ROL     (SP)+           ;IS THE REMAINDER OVER HALF?
5314  027464  100001                            BPL     5$              ;NO--BRANCH
5315  027466  005216                            INC     (SP)            ;YES--ROUND UP
5316  027470                                5$:
5317  027470  004737  023170                    JSR     PC,@#$S82D      ;CHANGE  TO DECIMAL ASCIZ
5318  027474  004737  023420                    JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
5319  027500  104401  043675                    TYPE    ,MSGOUS
5320  027504  104401  044010                    TYPE    ,MSG.SP
5321  027510  016446  177776                    MOV     -2(R4),-(SP)            ;PUT -2(R4) ON THE STACK
5322  027514  004737  023170                    JSR     PC,@#$S82D      ;CHANGE -2(R4) TO DECIMAL ASCIZ
5323  027520  004737  023420                    JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
5324  027524  104401  043760                    TYPE    ,MSGNUM         ;"SEEKS TIMED"
5325  027530  010537  027254                    MOV     R5,2$           ;NEXT MESSAGE POINTER
5326  027534  001404                            BEQ     7$              ;IF NONE EXIT
5327  027536  005005                            CLR     R5              ;NO MORE THAN 2
5328  027540  000644                            BR      1$
5329  027542  104401  043775    6$:             TYPE    ,MSGNON
5330  027546  000200                7$:         RTS     R0              ;EXIT
5331
5332                                    ;THIS SUBROUTINE WILL INCREMENT THE TRACK
5333                                    ;NUMBER (R2) BY THE AMOUNT SPECIFIED BY 'IT'.
5334                                    ;CALL
5335                                    ;       JSR     R0,@#INCTRK
5336                                    ;       RETURN1                 ;TRACK NUMBER GREATER THAN LT15
5337                                    ;       RETURN2                 ;TRACK NUMBER INCREMENTED
5338
5339  027550  020237  001520          INCTRK: CMP     R2,@#LT         ;LAST TRACK COMPLETED?
5340  027554  001410                            BEQ     2$              ;YES--EXIT
5341  027556  063702  001522                    ADD     @#IT,R2         ;NO--UPDATE TRACK
5342  027562  020237  001520                    CMP     R2,@#LT         ;TRACK TO BIG?
5343  027566  003402                            BLE     1$              ;NO--EXIT
5344  027570  013702  001520                    MOV     @#LT,R2         ;YES--SET TRACK TO LAST TRACK
5345  027574  005720                1$:         TST     (R0)+           ;ADJUST FOR RETURN 2
5346  027576  000200                2$:         RTS     R0              ;RETURN
5347
5348
5349                                    ;THIS SUBROUTINE WILL INCREMENT THE CYLINDER
5350                                    ;NUMBER (R1) BY THE AMOUNT SPECIFIED BY 'IC'.
5351                                    ;CALL
5352                                    ;       JSR     R0,@#INCCYL
5353                                    ;       RETURN1                 ;CYLINDER NUMBER GREATER THAN LC15
5354                                    ;       RETURN2                 ;CYLINDER NUMBER INCREMENTED
5355
5356  027600  020137  001512          INCCYL: CMP     R1,@#LC         ;LAST CYLINDER COMPLETED?
5357  027604  001410                            BEQ     2$              ;YES--EXIT
5358  027606  063701  001514                    ADD     @#IC,R1         ;NO--UPDATE CYLINDER
5359  027612  020137  001512                    CMP     R1,@#LC         ;CYLINDER TO BIG?
5360  027616  003402                            BLE     1$              ;NO--EXIT
5361  027620  013701  001512                    MOV     @#LC,R1         ;YES--SET CYLINDER TO LAST CYLINDER
5362  027624  005720                1$:         TST     (R0)+           ;ADJUST FOR RETURN 2
```

```
5363  027626  000200                    2$:     RTS     R0              ;RETURN
5364
5365                                    ;THIS ROUTINE DECREMENTS THE SECTOR ADDRESS.
5366                                    ;CALL
5367                                    ;       CLR     -(SP)           ;CLEAR THE STACK
5368                                    ;       JSR     PC,DECSEC       ;SUBROUTINE ENTRY
5369                                    ;       RETURN
5370
5371  027630  113766  004212  000002  DECSEC: MOVB    RP.REG+RPDA,2(SP)  ;PUT THE SECTOR ADDRESS ON THE STACK
5372  027636  005366  000002                  DEC     2(SP)           ;DECREMENT THE ADDRESS
5373  027642  100003                          BPL     1$              ;BR IF NOT CORRECTION NEEDED
5374  027644  013766  001634  000002        MOV     PRMLMT+22.,2(SP)  ;OVERFLOW OCCURED, FORCE TO MAXIMUM ADDRESS
5375  027652  000207                    1$:     RTS     PC              ;RETURN
5376
5377                                    ;THIS SUBROUTINE IS USED TO FILL THE DATA BUFFER
5378                                    ;WITH ADDRESSES FROM 0 TO 21 WITH EACH ADDRESS
5379                                    ;BEING STORED IN 256 CONSECUTIVE LOCATIONS
5380                                    ;CALL
5381                                    ;       JSR     PC,@#FILBUF
5382                                    ;       RETURN
5383
5384  027654  104412                    FILBUF: SAVREG                  ;SAVE R0 - R5
5385  027656  005000                          CLR     R0              ;FIRST DISK ADDRESS
5386  027660  012701  047662                  MOV     #BUFFER,R1      ;START FILLING HERE
5387  027664  012702  000400            1$:     MOV     #256.,R2        ;DO 256 WORDS
5388  027670  010021                    2$:     MOV     R0,(R1)+        ;STORE
5389  027672  005302                          DEC     R2              ;MORE?
5390  027674  003375                          BGT     2$              ;YES--BRANCH
5391  027676  005200                          INC     R0              ;NO--UPDATE DISK ADDRESS
5392  027700  023700  001630                  CMP     PRMLMT+22,R0    ;DONE?
5393  027704  103367                          BHIS    1$              ;NO--BRANCH
5394  027706  104413                          RESREG                  ;RESTORE R0 - R5
5395  027710  000207                          RTS     PC              ;RETURN
5396
5397                                    ;THIS ROUTINE WILL CLEAR THE BUFFER BY
5398                                    ;SETTING EACH WORD TO "177400".
5399                                    ;CALL
5400                                    ;       JSR     R0,@#CLRBUF
5401                                    ;       RETURN
5402
5403  027712  104412                    CLRBUF: SAVREG                  ;SAVE R0 - R5
5404  027714  012701  177400                  MOV     #177400,R1      ;WORD TO FILL BUFFER WITH
5405  027720  012702  047662                  MOV     #BUFFER,R2      ;FIRST ADDRESS OF BUFFER
5406  027724  012703  075662                  MOV     #BUFFER+<512.*22.>,R3 ;LAST ADDRESS+2 OF BUFFER
5407  027730  010122                    1$:     MOV     R1,(R2)+        ;FILL WORDS 1, 9....249,...5625
5408  027732  010122                          MOV     R1,(R2)+        ;FILL WORDS 2,10,...250,...5626
5409  027734  010122                          MOV     R1,(R2)+        ;FILL WORDS 3,11,...251,...5627
5410  027736  010122                          MOV     R1,(R2)+        ;FILL WORDS 4,12,...252,...5628
5411  027740  010122                          MOV     R1,(R2)+        ;FILL WORDS 5,13,...253,...5629
5412  027742  010122                          MOV     R1,(R2)+        ;FILL WORDS 6,14,...254,...5630
5413  027744  010122                          MOV     R1,(R2)+        ;FILL WORDS 7,15,...255,...5631
5414  027746  010122                          MOV     R1,(R2)+        ;FILL WORDS 8,16,...256,...5632
5415  027750  020203                          CMP     R2,R3           ;DONE?
5416  027752  103766                          BLO     1$              ;NO--BRANCH
5417  027754  104413                          RESREG                  ;RESTORE R0 - R5
5418  027756  000200                          RTS     R0              ;RETURN FROM CALL
```

```
                                        ;THIS ROUTINE IS USED TO CHECK THE DATA BUFFER
                                        ;FOR ADDRESSES 0 THROUGH 21 WITH EACH ADDRESS
                                        ;BEING STORED IN 256 CONSECUTIVE LOCATIONS
                                        ;CALL
                                        ;       JSR     R0,@#CKSCTR
                                        ;       RETURN
027760  104412                 CKSCTR: SAVREG                          ;SAVE R0 - P5
027762  162706  000004                  SUB     #4,SP                  ;RESERVE TEMP. STORAGE AREA
027766  005001                          CLR     R1                     ;FIRST SECTOR
027770  012716  047662                  MOV     #BUFFER,(SP)           ;FIRST ADDRESS OF DATA BUFFER
027774  005066  000002                  CLR     2(SP)                  ;NO ERRORS
030000  012702  000020          1$:     MOV     #16.,R2 ;LOOP COUNT (16*16=256)
030004  011603                          MOV     (SP),R3                ;GET 1ST ADDRESS OF THIS SECTORS DATA
030006                          2$:
030006  020123                          CMP     R1,(R3)+               ;WORD 1
030010  001063                          BNE     7$                     ;BRANCH IF BAD
030012  020123                          CMP     R1,(R3)+               ;WORD 2
030014  001061                          BNE     7$                     ;BRANCH IF BAD
030016  020123                          CMP     R1,(R3)+               ;WORD 3
030020  001057                          BNE     7$                     ;BRANCH IF BAD
030022  020123                          CMP     R1,(R3)+               ;WORD 4
030024  001055                          BNE     7$                     ;BRANCH IF BAD
030026  020123                          CMP     R1,(R3)+               ;WORD 5
030030  001053                          BNE     7$                     ;BRANCH IF BAD
030032  020123                          CMP     R1,(R3)+               ;WORD 6
030034  001051                          BNE     7$                     ;BRANCH IF BAD
030036  020123                          CMP     R1,(R3)+               ;WORD 7
030040  001047                          BNE     7$                     ;BRANCH IF BAD
030042  020123                          CMP     R1,(R3)+               ;WORD 8
030044  001045                          BNE     7$                     ;BRANCH IF BAD
030046  020123                          CMP     R1,(R3)+               ;WORD 9
030050  001043                          BNE     7$                     ;BRANCH IF BAD
030052  020123                          CMP     R1,(R3)+               ;WORD 10
030054  001041                          BNE     7$                     ;BRANCH IF BAD
030056  020123                          CMP     R1,(R3)+               ;WORD 11
030060  001037                          BNE     7$                     ;BRANCH IF BAD
030062  020123                          CMP     R1,(R3)+               ;WORD 12
030064  001035                          BNE     7$                     ;BRANCH IF BAD
030066  020123                          CMP     R1,(R3)+               ;WORD 13
030070  001033                          BNE     7$                     ;BRANCH IF BAD
030072  020123                          CMP     R1,(R3)+               ;WORD 14
030074  001031                          BNE     7$                     ;BRANCH IF BAD
030076  020123                          CMP     R1,(R3)+               ;WORD 15
030100  001027                          BNE     7$                     ;BRANCH IF BAD
030102  020123                          CMP     R1,(R3)+               ;WORD 16
030104  001025                          BNE     7$                     ;BRANCH IF BAD
030106  005302                          DEC     R2                     ;FINISHED WITH THIS SECTORS DATA?
030110  001336                          BNE     2$                     ;NO--BRANCH
030112  062716  001000          3$:     ADD     #512.,(SP)             ;YES--FIRST ADDRESS OF NEXT SECTOR
030116  005201                          INC     R1                     ;MOVE TO NEXT SECTOR
030120  023701  001630                  CMP     PRMLMT+22,R1           ;DONE?
030124  103425                          BLOS    1$                     ;NO--BRANCH
030126  005766  000002          4$:     TST     2 SP)                  ;ERROR OCCUR?
030132  001406                          BEQ     6$                     ;NO--BRANCH
```

```
5475  030134  123737  001364  001103          CMPB    @#ERR.CT,@#SERFLG ;MAX. ERROR OCCURRED?
5476  030142  101002                          BHI     6$                ;NO--BRANCH
5477  030144  013700  001252          5$:     MOV     @#BYPASS,R0       ;TAKE ERROR EXIT
5478  030150  062706  000004          6$:     ADD     #4,SP             ;FREE TEMP. AREA
5479  030154  104413                          RESREG                    ;RESTORE R0 - R5
5480  030156  000200                          RTS     R0                ;RETURN FROM CALL
5481  030160  010304                  7$:     MOV     R3,R4             ;FORM WORD NUMBER AND
5482  030162  161404                          SUB     (SP),R4           ;ADDRESS TO CONTINUE FROM
5483  030164  010405                          MOV     R4,R5
5484  030166  006204                          ASR     R4                ;WORD NUMBER
5485  030170  042705  177740                  BIC     #^C37,R5
5486  030174  001002                          BNE     8$                ;BRANCH IF NOT A MULTIPLE OF 16
5487  030176  012705  000040                  MOV     #40,R5            ;SET TO WORD 16
5488  030202  006305                  8$:     ASL     R5
5489  030204  062705  030006                  ADD     #2$,R5            ;ADDRESS
5490  030210  016337  177776  001126          MOV     -2(R3),@#BDUAT    ;SAVE BAD DATA
5491  030216  005766  000002                  TST     2(SP)             ;FIRST ERROR?
5492  030222  001015                          BNE     10$               ;NO--BRANCH
5493  030224  013737  004176  001270          MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER NUMBER
5494  030232  113737  004175  001274          MOVB    @#DTADPB+11,@#TRK.DS ;TRACK NUMBER
5495  030240  012737  030250  001206          MOV     #9$,$ESCAPE       ;ESCAPE TO 9$ ON ERROR
5496  030246  104021                          ERROR   21                ;REPORT THE ERROR
5497  030250  105166  000002          9$:     COMB    2(SP)             ;SET ERROR SWITCH
5498  030254  000404                          BR      11$
5499  030256                          10$:
5500  030256  012737  030266  001206          MOV     #11$,$ESCAPE      ;ESCAPE TO 11$ ON ERROR
5501  030264  104022                          ERROR   22                ;REPORT THE ERROR
5502  030266  032777  001000  150644  11$:    BIT     #SW09,@SWR        ;LOOP ON ERROR?
5503  030274  001323                          BNE     5$                ;YES
5504  030276  032777  000002  150634          BIT     #SW01,@SWR        ;STOP DATA COMPARE?
5505  030304  001310                          BNE     4$                ;YES--BRANCH
5506  030306  123737  001364  001103          CMPB    @#ERR.CT,@#SERFLG ;MAX. ERRORS?
5507  030314  101713                          BLOS    5$                ;YES--BRANCH
5508  030316  032777  000040  150614          BIT     #SW05,@SWR        ;REPORT ONLY 1ST ERROR PER SECTOR?
5509  030324  001272                          BNE     3$                ;YES--BRANCH
5510  030326  000115                          JMP     (R5)
5511
5512                                  ;THIS ROUTINE WILL MOVE THE 16 WORDS OF THE
5513                                  ;DESIRED PATTERN INTO THE DATA BUFFER.
5514                                  ;CALL
5515                                  ;       MOV     #NX,R0            ;PATTERN NUMBER INDEX TO R0
5516                                  ;       JSR     PC,@#SETBUF
5517
5518  030330  104412                  SETBUF: SAVREG                    ;SAVE R0 - R5
5519  030332  012701  047642                  MOV     #BUFFER,R1        ;FIRST ADDRESS
5520  030336  013702  004170                  MOV     @#DTADPB+4,R2     ;WORD COUNT
5521  030342  016003  003044          1$:     MOV     PAT.PT(R0),R3     ;PICKUP PATTERN POINTER
5522  030346  012321                          MOV     (R3)+,(R1)+       ;MOVE WORD 1 INTO DATA BUFFER
5523  030350  012321                          MOV     (R3)+,(R1)+       ;MOVE WORD 2 INTO DATA BUFFER
5524  030352  012321                          MOV     (R3)+,(R1)+       ;MOVE WORD 3 INTO DATA BUFFER
5525  030354  012321                          MOV     (R3)+,(R1)+       ;MOVE WORD 4 INTO DATA BUFFER
5526  030356  012321                          MOV     (R3)+,(R1)+       ;MOVE WORD 5 INTO DATA BUFFER
5527  030360  012321                          MOV     (R3)+,(R1)+       ;MOVE WORD 6 INTO DATA BUFFER
5528  030362  012321                          MOV     (R3)+,(R1)+       ;MOVE WORD 7 INTO DATA BUFFER
5529  030364  012321                          MOV     (R3)+,(R1)+       ;MOVE WORD 8 INTO DATA BUFFER
5530  030366  012321                          MOV     (R3)+,(R1)+       ;MOVE WORD 9 INTO DATA BUFFER
```

```
5531  030370  012321                MOV     (R3)+,(R1)+     ;MOVE WORD 10 INTO DATA BUFFER
5532  030372  012321                MOV     (R3)+,(R1)+     ;MOVE WORD 11 INTO DATA BUFFER
5533  030374  012321                MOV     (R3)+,(R1)+     ;MOVE WORD 12 INTO DATA BUFFER
5534  030376  012321                MOV     (R3)+,(R1)+     ;MOVE WORD 13 INTO DATA BUFFER
5535  030400  012321                MOV     (R3)+,(R1)+     ;MOVE WORD 14 INTO DATA BUFFER
5536  030402  012321                MOV     (R3)+,(R1)+     ;MOVE WORD 15 INTO DATA BUFFER
5537  030404  012321                MOV     (R3)+,(R1)+     ;MOVE WORD 16 INTO DATA BUFFER
5538  030406  062702  000020        ADD     #16.,R2 ;DONE?
5539  030412  001353                BNE     1$              ;NO--BRANCH
5540  030414  104413                RESREG                  ;RESTORE R0 - R5
5541  030416  000207                RTS     PC              ;RETURN
5542
5543                                ;THIS ROUTINE COMPARES A 16 WORD DATA PATTERN
5544                                ;AGAINST THE DATA BUFFER
5545                                ;CALL
5546                                :       MOV     #NX,R0          ;PATTERN NUMBER INDEX TO R0
5547                                :       JSR     PC,@#DATCMP
5548                                :       RETURN
5549
5550  030420  104412                DATCMP: SAVREG                  ;SAVE R0 - R5
5551  030422  012701  047662                MOV     #BUFFER,R1      ;FIRST ADDRESS OF BUFFER
5552  030426  013702  004170                MOV     @#DTADPB+4,R2   ;WORD COUNT
5553  030432  005046                        CLR     -(SP)           ;NO ERROR
5554  030434  016003  003044        1$:     MOV     PAT.PT(R0),R3   ;PATTERN POINTER
5555  030440                        2$:
5556  030440  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 1
5557  030442  001044                        BNE     4$              ;BRANCH IF DIFFERENT
5558  030444  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 2
5559  030446  001042                        BNE     4$              ;BRANCH IF DIFFERENT
5560  030450  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 3
5561  030452  001040                        BNE     4$              ;BRANCH IF DIFFERENT
5562  030454  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 4
5563  030456  001036                        BNE     4$              ;BRANCH IF DIFFERENT
5564  030460  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 5
5565  030462  001034                        BNE     4$              ;BRANCH IF DIFFERENT
5566  030464  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 6
5567  030466  001032                        BNE     4$              ;BRANCH IF DIFFERENT
5568  030470  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 7
5569  030472  001030                        BNE     4$              ;BRANCH IF DIFFERENT
5570  030474  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 8
5571  030476  001026                        BNE     4$              ;BRANCH IF DIFFERENT
5572  030500  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 9
5573  030502  001024                        BNE     4$              ;BRANCH IF DIFFERENT
5574  030504  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 10
5575  030506  001022                        BNE     4$              ;BRANCH IF DIFFERENT
5576  030510  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 11
5577  030512  001020                        BNE     4$              ;BRANCH IF DIFFERENT
5578  030514  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 12
5579  030516  001016                        BNE     4$              ;BRANCH IF DIFFERENT
5580  030520  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 13
5581  030522  001014                        BNE     4$              ;BRANCH IF DIFFERENT
5582  030524  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 14
5583  030526  001012                        BNE     4$              ;BRANCH IF DIFFERENT
5584  030530  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 15
5585  030532  001010                        BNE     4$              ;BRANCH IF DIFFERENT
5586  030534  162321                        SUB     (R3)+,(R1)+     ;CHECK WORD 16
```

```
5587  030536  001006                        BNE     4$              ;BRANCH IF DIFFERENT
5588  030540  062702  000020                ADD     #16.,R2         ;DONE ?
5589  030544  001333                        BNE     1$              ;NO--BRANCH
5590  030546  005726                3$:     TST     (SP)+           ;YES -- CLEAN UP STACK
5591  030550  104413                        RESREG                  ;RESTORE R0 - R5
5592  030552  000207                        RTS     PC
5593  030554  010104                4$:     MOV     R1,R4           ;FORM THE WORD NUMBER
5594  030556  162704  047662                SUB     #BUFFER,R4
5595  030562  006204                        ASR     R4              ;WORD NUMBER
5596  030564  010305                        MOV     R3,R5           ;FORM ADDRESS TO CONTINUE FROM
5597  030566  166005  003044                SUB     PAT.PT(R0),R5
5598  030572  006305                        ASL     R5
5599  030574  062705  030440                ADD     #2$,R5          ;ADDRESS
5600  030600  064341                        ADD     -(R3),-(R1)     ;RECONSTRUCT THE BAD WORD
5601  030602  010137  001122                MOV     R1,@#$BDAOR     ;SAVE THE ERROR INFORMATION
5602  030606  010337  001120                MOV     R3,@#$GDAOR     ;
5603  030612  012137  001126                MOV     (R1)+,@#$BDDAT  ;
5604  030616  012337  001124                MOV     (R3)+,@#$GDDAT  ;
5605  030622  005716                        TST     (SP)            ;1ST DATA COMPARE ERROR?
5606  030624  001023                        BNE     6$              ;NO--BRANCH
5607  030626  013737  004176  001270        MOV     @#DTADPB+12,@#CYL.DS ;CYLINDER
5608  030634  113737  004175  001274        MOVB    @#DTADPB+11,@#TRK.DS ;TRACK
5609  030642  113737  004174  001272        MOVB    @#DTADPB+10,@#SEC.DS ;SECTOR
5610  030650  016600  000016                MOV     16(SP),R0       ;GET TEST PC+4
5611  030654  012737  030664  001206        MOV     #5$,$ESCAPE     ;;ESCAPE TO 5$ ON ERROR
5612  030662  104013                        ERROR   13              ;REPORT THE ERROR
5613  030664  016600  000014          5$:   MOV     14(SP),R0       ;PATTERN NUMBER INDEX
5614  030670  105116                        COMB    (SP)            ;SET THE ERROR SWITCH
5615  030672  000404                        BR      7$
5616  030674                          6$:
5617  030674  012737  030704  001206        MOV     #7$,$ESCAPE     ;;ESCAPE TO 7$ ON ERROR
5618  030702  104014                        ERROR   14              ;REPORT THE ERROR
5619  030704  032777  000002  150226  7$:   BIT     #S.HD1,@SWR     ;STOP DATA COMPARE?
5620  030712  001315                        BNE     3$              ;YES--EXIT
5621  030714  123737  001364  001103        CMPB    @#ERR.CT,@#SERFLG ;MAX. ERRORS?
5622  030722  101004                        BHI     8$              ;NO--BRANCH
5623  030724  013766  001252  000016        MOV     @#BYPASS,16(SP) ;YES--ERROR EXIT
5624  030732  000705                        BR      3$
5625  030734  000115                8$:     JMP     (R5)            ;NO--CONTINUE AT NEXT WORD
5626
5627                                  ;THIS ROUTINE WILL FILL THE DATA BUFFER (256*22 WORDS) WITH
5628                                  ;A RANDOM PATTERN.  THE FIRST TWO WORDS OF EVERY 256 WILL
5629                                  ;BE THE BASE OF THE RANDOM NUMBER GENERATOR FOR THE
5630                                  ;NEXT 254 WORDS.
5631                                  ;NOTE: THIS ROUTINE DESTROYS R1 AND R2
5632                                  ;CALL
5633                                  ;       JSR     R0,@#FILRAN
5634                                  ;       RETURN
5635
5636  030736  012701  047662          FILRAN: MOV     #BUFFER,R1
5637  030742  013702  001630                  MOV     PRMLMT+22,R2    ;MAXIMUM NUMBER OF SECTORS
5638  030746  004037  031156          1$:     JSR     R0,@#RANPAT
5639  030752  005302                        DEC     R2
5640  030754  100374                        BPL     1$
5641  030756  000200                        RTS     R0
5642
```

```
                                        *** PROGRAM SUBROUTINES ***

                                        ;THIS ROUTINE USES THE FIRST TWO WORDS OF THE
                                        ;READ BUFFER TO GENERATED A RANDOM PATTERN.  THEN
                                        ;THE READ BUFFER IS COMPARED TO THE PATTERN GENERATED.
                                        ;NOTE: THIS ROUTINE DESTROYS R1-R4
                                        ;CALL
                                        ;       JSR     R0,@#RANCK
                                        ;       RETURN
                                        ;
030750  013746  023556          RANCK:  MOV     @#$HINUM,-(SP)  ;SAVE THE PRESENT RANDOM NUMBER
030764  013746  023560                  MOV     @#$LONUM,-(SP)
030770  012702  050662                  MOV     #BUFFER+512.,R2 ;READ BUFFER ADDRESS
030774  012701  051662                  MOV     #BUFFER+1024.,R1 ;RANDOM PATTERN ADDRESS
031000  010103                          MOV     R1,R3           ;COPY IT INTO R3 FOR LATER USE
031002  011237  023560                  MOV     (R2),@#$LONUM   ;PRIME THE RANDOM NUMBER GENERATOR
031006  016237  000002  023556          MOV     2(R2),@#$HINUM
031014  004037  031156                  JSR     R0,@#RAMPAT     ;GENERATE A RANDOM PATTERN
031020  012637  023560                  MOV     (SP)+,@#$LONUM  ;RESTORE PRESENT RANDOM NUMBER
031024  012637  023556                  MOV     (SP)+,@#$HINUM
031030  005046                          CLR     -(SP)           ;NO ERRORS
031032  162322                  1$:     SUB     (R3)+,(R2)+     ;ARE THESE TWO WORDS DIFFERENT?
031034  001441                          BEQ     4$              ;NO--BRANCH
031036  012737  031110  001206          MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
031044  064342                          ADD     -(R3),-(R2)     ;RECREATE THE BAD WORD
031046  010237  001122                  MOV     R2,@#$BADDR     ;ADDRESS OF BAD DATA
031052  010337  001120                  MOV     R3,@#$GDADR     ;ADDRESS OF GOOD DATA
031056  012237  001126                  MOV     (R2)+,@#$BDDAT  ;BAD DATA
031062  012337  001124                  MOV     (R3)+,@#$GDDAT  ;GOOD DATA
031066  010204                          MOV     R2,R4           ;FORM WORD NUMBER (1 TO 256)
031070  162704  050662                  SUB     #BUFFER+512.,R4
031074  006204                          ASR     R4
031076  005716                          TST     (SP)            ;FIRST ERROR
031100  001002                          BNE     2$              ;NO--BRANCH
031102  105116                          COMB    (SP)            ;YES--SET ERROR SWITCH
031104  104015                          ERROR   15              ;REPORT THE ERROR
031106  104016                  2$:     ERROR   16              ;REPORT THE ERROR
031110  032777  001000  150022  3$:     BIT     #SWD9,@SWR      ;LOOP ON ERROR?
031116  001012                          BNE     5$              ;YES--BRANCH
031120  123737  001364  001103          CMPB    @#ERR.CT,@#SERFLG ;MAX. ERRORS OCCURRED?
031126  101406                          BLOS    5$              ;YES--BRANCH
031130  032777  000002  150002          BIT     #SWD1,@SWR      ;STOP COMPARING?
031136  001002                          BNE     5$              ;YES--BRANCH
031140  020103                  4$:     CMP     R1,R3           ;ALL DATA BEEN COMPARED?
031142  101333                          BHI     1$              ;NO--BRANCH
031144  005726                  5$:     TST     (SP)+           ;ERROR OCCUR?
031146  001402                          BEQ     6$              ;NO--BRANCH
031150  013700  001252                  MOV     @#BYPASS,R0     ;TAKE ERROR EXIT
031154  000200                  6$:     RTS     R0              ;EXIT

                                        ;THIS ROUTINE FILLS A 256 WORD BUFFER WITH A RANDOM
                                        ;PATTERN OF WHICH THE FIRST TWO WORDS ARE THE BASE
                                        ;OF THE PATTERN.
                                        ;CALL
                                        ;       MOV     #ADR,R1         ;ADDRESS OF THE BUFFER
                                        ;       JSR     R0,@#RAMPAT
                                        ;       RETURN
```

```
                                *** PROGRAM SUBROUTINES ***

5699  031156  010246            RANPAT: MOV     R2,-(SP)        ;SAVE R2
5700  031160  012702  000200            MOV     #256./2.,R2     ;GENERATE 256 WORDS
5701  031164  000403                    BR      2$
5702  031166  004737  023460    1$:     JSR     PC,@#$RAND      ;GENERATE A RANDOM NUMBER
5703  031172  013721  023560    2$:     MOV     @#$LONUM,(R1)+  ;PUT LOW WORD IN BUFFER
5704  031176  013721  023556            MOV     @#$HINUM,(R1)+  ;PUT HIGH WORD IN BUFFER
5705  031202  005302                    DEC     R2              ;DONE?
5706  031204  003370                    BGT     1$              ;NO--BRANCH
5707  031206  012602                    MOV     (SP)+,R2        ;RESTORE R2
5708  031210  000200                    RTS     R0              ;EXIT
5709
5710                                ;THIS ROUTINE GENERATES RANDOM CYLINDER, TRACK, AND SECTOR
5711                                ;ADDRESSES AND SAVES THEN IN THE DPB (DTADPB+10 AND DTADPB+12).
5712                                ;NOTE: THIS ROUTINE DESTROYS R1-R3
5713                                ;CALL
5714                                ;       JSR     R0,@#RANADR
5715                                ;       RETURN
5716
5717  031212  004737  023460    RANADR: JSR     PC,@#$RAND      ;GENERATE A RANDOM NUMBER
5718  031216  113701  023560            MOVB    @#$LONUM,R1     ;FORM SECTOR IN R1
5719  031222  113702  023561            MOVB    @#$LONUM+1,R2   ;FORM TRACK IN R2
5720  031226  013703  023556            MOV     @#$HINUM,R3     ;FORM CYLINDER IN R3
5721  031232  105701                    TSTB    R1              ;ENSURE THE SECTOR IS BETWEEN 0 AND 21
5722  031234  002403                    BLT     2$
5723  031236  123701  001630    1$:     CMPB    PRMLMT+22,R1    ;CHECK MAXIMUM SECTOR ADDRESS
5724  031242  103003                    BHIS    3$
5725  031244  000241            2$:     CLC
5726  031246  106001                    RORB    R1
5727  031250  000772                    BR      1$
5728  031252  105702            3$:     TSTB    R2              ;ENSURE THE TRACK IS BETWEEN 0 AND 18
5729  031254  002403                    BLT     5$
5730  031256  122702  000023    4$:     CMPB    #19.,R2
5731  031262  003003                    BGT     6$
5732  031264  000241            5$:     CLC
5733  031266  106002                    RORB    R2
5734  031270  000772                    BR      4$
5735  031272  023703  001510    6$:     CMP     @#FC,R3         ;ENSURE THE CYLINDER IS BETWEEN FC AND LC
5736  031276  003413                    BLE     7$
5737  031300  000241                    CLC
5738  031302  006003                    ROR     R3
5739  031304  005503                    ADC     R3
5740  031306  001371                    BNE     6$
5741  031310  010103                    MOV     R1,R3
5742  031312  000303                    SWAB    R3
5743  031314  060203                    ADD     R2,R3
5744  031316  005203                    INC     R3
5745  031320  003364                    BGT     6$
5746  031322  005403                    NEG     R3
5747  031324  000762                    BR      6$
5748  031326  023703  001512    7$:     CMP     @#LC,R3
5749  031332  002003                    BGE     8$
5750  031334  000241                    CLC
5751  031336  006003                    ROR     R3
5752  031340  000772                    BR      7$
5753  031342  023703  001510    8$:     CMP     @#FC,R3
5754  031346  003403                    BLE     9$
```

```
5755  031350  005203                          INC     R3
5756  031352  000303                          SWAB    R3
5757  031354  000764                          BR      7$
5758  031356  110137  004174          9$:     MOVB    R1,@#DTADPB+10  ;SAVE SECTOR ADDRESS
5759  031362  110237  004175                  MOVB    R2,@#DTADPB+11  ;SAVE TRACK ADDRESS
5760  031366  010337  004176                  MOV     R3,@#DTADPB+12  ;SAVE CYLINDER ADDRESS
5761  031372  000200                          RTS     R0              ;RETURN
5762
5763                                  ;THIS ROUTINE IS USED TO INPUT THE "CONTROL SWITCHES".
5764                                  ;IF SWR(07)=1 THE PRESENT SETTING WILL BE TYPED AND THE NEW
5765                                  ;SETTING IS READ AND STORED.
5766                                  ;NOTE: THIS ROUTINE DESTROYS R3 AND R4
5767                                  ;CALL
5768                                  ;       JSR     PC,@#GETSWR
5769                                  ;       RETURN                  ;(C.SWR)=DESIRED CONTROL SWITCHES
5770
5771  031374  032777  000200  147536  GETSWR: BIT     #SW07,@SWR      ;READ CONTROL SWITCHES?
5772  031402  001430                          BEQ     2$              ;NO--BRANCH
5773  031404  104401  031412                  TYPE    ,65$            ;;TYPE ASCIZ STRING
5774  031410  000410                          BR      64$             ;;GET OVER THE ASCIZ
5775                                  ;;65$:  .ASCIZ  <CR><LF>/SET SWR(07)=0/
5776  031432                          64$:
5777  031432  012703  043020                  1$:     MOV     #MSG.CS,R3      ;"CONTROL SWITCHES="
5778  031436  013704  001220                          MOV     @#C.SWR,R4      ;PRESENT CONTROL SWITCH SETTINGS
5779  031442  004037  031466                          JSR     R0,@#GETNUM     ;GET THE NEW SWITCH SETTINGS
5780  031446  000771                          BR      1$              ; COMMA
5781  031450  000240                          NOP                     ;PERIOD
5782  031452  013737  001220  001222                  MOV     C.SWR,SAVCSW    ;SAVE PREVIOUS VALUE
5783  031460  010437  001220                          MOV     R4,@#C.SWR      ; DOUBLE PERIOD-SAVE NEW SWITCH SETTING
5784  031464  000207                  2$:     RTS     PC              ;RETURN FROM CALL
5785
5786                                  ;THIS ROUTINE WILL TYPE AN ASCIZ MESSAGE AND THEN
5787                                  ;INPUT AN ASCIZ STRING AND CHANGE THE STRING TO OCTAL
5788                                  ;IF REQUIRED.
5789                                  ;NOTE: THIS ROUTINE DESTROYS R1
5790                                  ;CALL
5791                                  ;       MOV     #ADR,R3         ;ADDRESS OF ASCIZ MESSAGE
5792                                  ;       MOV     #NUM,R4         ;OCTAL NUMBER
5793                                  ;       JSR     R0,@#GETNUM
5794                                  ;       RETURN1                 ;INPUT TERMINATED WITH A COMMA
5795                                  ;       RETURN2                 ;WITH A PERIOD
5796                                  ;       RETURN3                 ;WITH A DOUBLE PERIOD
5797                                  ;                               ;R4=INPUT NUMBER AND
5798                                  ;                               ;R2=R4*32 FOR ALL
5799                                  ;                               ;THREE RETURNS
5800
5801  031466  010337  031474          GETNUM: MOV     R3,2$           ;SAVE MESSAGE POINTER
5802  031472  104401                  1$:     TYPE                    ;TYPE THE MESSAGE
5803  031474  000000                  2$:     .WORD   0               ;MESSAGE POINTER GOES HERE
5804  031476  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
5805  031500  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5806  031502  104401  043044                  TYPE    ,SLASH          ;' / '
5807  031506  104411                          RDLIN                   ;READ AN ASCIZ STRING
5808  031510  012601                          MOV     (SP)+,R1        ;ADDRESS OF FIRST CHARACTER
5809  031512  004037  033736                  JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
5810  031516  031472                          1$                      ;ILLEGAL CHARACTER
```

```
5811  031520  031472                        1$                      ;CARRIAGE RETURN
5812  031522  031534                        3$                      ;"/"
5813  031524  031560                        7$                      ;"."
5814  031526  031566                        8$                      ;"'"
5815  031530  031532                        11$                     ;DIGIT 0-9
5816  031532  005301              11$:      DEC     R1              ;DECREMENT THE INPUT POINTER
5817  031534                      3$:
5818  031534  004037  034176              JSR     R0,@#CK.NUM     ;CHECK THE NUMBER
5819  031540  031472                        1$                      ;ILLEGAL INPUT
5820  031542  031554                        6$                      ;TERMINATED WITH A "," OR "CR"
5821  031544  031552                        5$                      ;TERMINATED WITH A "'"
5822  031546  031550                        4$                      ;TERMINATED WITH A ".."
5823  031550  005720              4$:       TST     (R0)+           ;DOUBLE PERIOD
5824  031552  005720              5$:       TST     (R0)+           ;SINGLE PERIOD
5825  031554  010204              6$:       MOV     R2,R4           ;COMMA--SAVE INPUT NUMBER
5826  031556  000414                        BR      10$             ;GO TO EXIT
5827  031560  105711              7$:       TSTB    (R1)            ;TERMINATOR AFTER A COMMA?
5828  031562  001343                        BNE     1$              ;NO--LOOP
5829  031564  000411                        BR      10$             ;YES--EXIT
5830  031566  105711              8$:       TSTB    (R1)            ;TERMINATOR AFTER A PERIOD?
5831  031570  001406                        BEQ     9$              ;YES--EXIT
5832  031572  122721  000056              CMPB    #'.,(R1)+       ;NO--DOUBLE PERIOD?
5833  031576  001335                        BNE     1$              ;NO--LOOP
5834  031600  105711                        TSTB    (R1)            ;YES--TERMINATOR?
5835  031602  001333                        BNE     1$              ;NO--LOOP
5836  031604  005720                        TST     (R0)+           ;DOUBLE PERIOD
5837  031606  005720              9$:       TST     (R0)+           ;PERIOD
5838  031610  010402              10$:      MOV     R4,R2           ;COMMA--POSITION THE
5839  031612  000302                        SWAB    R2              ;NUMBER IN CASE IT
5840  031614  006202                        ASR     R2              ;IS THE PRIORITY LEVEL
5841  031616  006202                        ASR     R2
5842  031620  006202                        ASR     R2
5843  031622  000200                        RTS     R0              ;EXIT
5844
5845                          ;THIS ROUTINE IS USED TO CHANGE OR MODIFY
5846                          ;THE TEST PARAMETERS.  IT GIVES THE OPERATOR
5847                          ;THE CAPABILITY OF SPECIFYING WHICH DRIVES TO TEST, WHICH
5848                          ;TESTS TO RUN AND HOW MANY TIMES TO
5849                          ;REPEAT EACH TEST
5850
5851  031624  104412              GT.PRM: SAVREG                  ;SAVE R0 - R5
5852  031626  005037  001232      GT.PR1: CLR     DRVSEL          ;NO DRIVE SELECTED
5853  031632  104401  031640              TYPE    ,65$            ;;TYPE ASCIZ STRING
5854  031636  000406                      BR      64$             ;;GET OVER THE ASCIZ
5855                          ;;65$:  .ASCIZ  <CR><LF>/DRIVE(S)=/
5856  031654                  64$:
5857  031654  104411                      RDLIN                   ;READ TTY
5858  031656  012601                      MOV     (SP)+,R1        ;ADDRESS OF ASCIZ STRING
5859  031660  004037  033736              JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
5860  031664  031626                      GT.PR1                  ;ILLEGAL CHARACTER
5861  031666  031626                      GT.PR1                  ;CARRIAGE RETURN
5862  031670  031626                      GT.PR1                  ;"/"
5863  031672  031626                      GT.PR1                  ;"."
5864  031674  031626                      GT.PR1                  ;"'"
5865  031676  031700                      1$                      ;DIGIT 0-9
5866  031700  005301              1$:     DEC     R1
```

```
                                        2$:
031702                                          MOV     #7,R2   ;UPPER LIMIT OF INPUT
031702  012702  000007                          JSR     R0,@#CK.DIG     ;CHECK THE DIGIT(S)
031706  004037  034012                          GT.PR1                  ;ILLEGAL INPUT
031712  031626                                  GT.PR1                  ;INPUT TO LARGE
031714  031626                                  3$                      ;TERMINATED WITH A "," OR "CR"
031716  031724                                  4$                      ;TERMINATED WITH A ":"
031720  031750                                  4$                      ;TERMINATED WITH A "/"
031722  031750                          3$:     BISB    ATABIT(R2),DRVSEL ;SET THE DRIVE SELECTED BIT
031724  156237  034452  001232                  TSTB    -(R1)           ;WAS THE LINE TERMINATED?
031732  105741                                  BNE     2$              ;NO-GET THE NEXT DRIVE
031734  001362                                  CLR     @#TSTNMS        ;DESELECT ALL TESTS
031736  005037  001234                          CLR     TSTNMS+2
031742  005037  001236                          BR      GTTST1          ;YES--SELECT TEST
031746  000405                          4$:     BISB    ATABIT(R2),DRVSEL ;SET THE SELECTED DRIVE BITS
031750  156237  034452  001232          GT.PR2: RESREG                  ;RESTORE R0 - R5
031756  104413                                  RTS     PC              ;EXIT
031760  000207
                                        GTTST1:
031762                                          TYPE    ,65$            ;;TYPE ASCIZ STRING
031762  104401  031770                          BR      64$             ;;GET OVER THE ASCIZ
031766  000403                          ;;65$:  .ASCIZ  /TEST=/
                                        64$:
031776                                          RDLIN                   ;READ AN ASCIZ STRING
031776  104411                                  MOV     (SP)+,R1        ;POINTER TO R1
032000  012601                                  CMPB    #'.,(R1)        ;DOUBLE PERIOD?
032002  122711  000056                          BNE     1$              ;NO--BRANCH
032006  001007                                  CMPB    #'.,1(R1)
032010  122761  000056  000001                  BNE     1$
032016  001003                                  TSTB    2(R1)           ;"CR"?
032020  105761  000002                          BEQ     GT.PR2          ;YES--EXIT
032024  001754                          1$:     CLR     TSTNMS          ;NO TEST SELECTED
032026  005037  001234                          CLR     TSTNMS+2
032032  005037  001236                          CLR     OPNFLG          ;NO TESTS TO BE OPENED
032036  005037  001240                          CLR     OPNFLG+2
032042  005037  001242                  GTTST2: CMPB    (R1),#'S        ;ALL SEEK TESTS?
032046  121127  000123                          BNE     1$              ;NO--BRANCH
032052  001004                                  BIS     #777,TSTNMS     ;YES--SELECT TESTS 0-10
032054  052737  000777  001234                  BR      GTTST3
032062  000552                          1$:     CMPB    (R1),#'T        ;ALL TIMING TESTS?
032064  121127  000124                          BNE     2$              ;NO--BRANCH
032070  001004                                  BIS     #36000,TSTNMS   ;YES--SELECT TESTS 12-15
032072  052737  036000  001234                  BR      GTTST3
032100  000543                          2$:     CMPB    (R1),#'A        ;ALL ADDRESSING TESTS?
032102  121127  000101                          BNE     3$              ;NO--BRANCH
032106  001004                                  BIS     #140000,TSTNMS  ;YES--SELECT TESTS 16 & 17
032110  052737  140000  001234                  BR      GTTST3
032116  000534                          3$:     CMPB    (R1),#'D        ;DATA TEST?
032120  121127  000104                          BNE     4$              ;NO--BRANCH
032124  001004                                  BIS     #1,TSTNMS+2     ;YES--SELECT TEST 20
032126  052737  000001  001236                  BR      GTTST3
032134  000525                          4$:     CMPB    (R1),#'E        ;EXERCISER TEST?
032136  121127  000105                          BNE     5$              ;NO--BRANCH
032142  001004                                  BIS     #2,TSTNMS+2     ;YES--SELECT TEST 21
032144  052737  000002  001236                  BR      GTTST3
032152  000516                          5$:     JSR     R0,@#CK.OCT     ;OCTAL DIGIT?
032154  004037  033662
```

```
5923  032160  000514                          BR      GTTST4          ;NO--BRANCH
5924  032162  010205                          MOV     R2,R5           ;YES--SAVE IT
5925  032164  005201                          INC     R1              ;MOVE TO NEXT CHARACTER
5926  032166  004037  033662                  JSR     R0,2$CK.OCT     ;OCTAL DIGIT
5927  032172  000405                          BR      6$              ;NO--BRANCH
5928  032174  005201                          INC     R1              ;MOVE TO NEXT CHARACTER
5929  032176  006305                          ASL     R5              ;SCALE HIGH DIGIT
5930  032200  006305                          ASL     R5
5931  032202  006305                          ASL     R5
5932  032204  060502                          ADD     R5,R2           ;COMBINE HIGH & LOW DIGITS
5933  032206  020227  000022          6$:     CMP     R2,#STN-1       ;VALID TEST NUMBER?
5934  032212  003263                          BGT     GTTST1          ;NO--BRANCH
5935  032214  010237  032406                  MOV     R2,9$           ;SAVE THE TEST NUMBER
5936  032220  010204                          MOV     R2,R4           ;CONVERT TEST NUMBER INTO AN INDEX
5937  032222  042704  000017                  BIC     #17,R4          ;CLEAR UNWANTED BITS
5938  032226  006204                          ASR     R4              ;SHIFT THE BITS
5939  032230  006204                          ASR     R4              ;SHIFT THE BITS
5940  032232  006204                          ASR     R4              ;SHIFT THE BITS
5941  032234  006302                          ASL     R2
5942  032236  056264  001424  001234          BIS     BITS(R2),TSTNMS(R4)  ;SELECT TEST
5943  032244  121127  000055                  CMPB    (R1),#'-        ;TEST STRING?
5944  032250  001060                          BNE     GTTST4          ;NO--BRANCH
5945  032252  005201                          INC     R1              ;YES--MOVE TO NEXT CHARACTER
5946  032254  004037  033662                  JSR     R0,2$CK.OCT     ;OCTAL DIGIT?
5947  032260  000640                          BR      GTTST1          ;NO--BRANCH
5948  032262  010205                          MOV     R2,R5           ;YES--SAVE IT
5949  032264  005201                          INC     R1              ;MOVE TO NEXT CHARACTER
5950  032266  004037  033662                  JSR     R0,2$CK.OCT     ;OCTAL DIGIT?
5951  032272  000405                          BR      7$              ;NO--BRANCH
5952  032274  005201                          INC     R1              ;YES--MOVE TO NEXT CHARACTER
5953  032276  006305                          ASL     R5              ;SCALE HIGH DIGIT
5954  032300  006305                          ASL     R5
5955  032302  006305                          ASL     R5
5956  032304  060502                          ADD     R5,R2           ;COMBINE HIGH & LOW DIGIT
5957  032306  020227  000022          7$:     CMP     R2,#STN-1       ;VALID TEST NUMBER?
5958  032312  003223                          BGT     GTTST1          ;NO--BRANCH
5959  032314  023702  032406                  CMP     9$,R2           ;IS THE FIRST NUMBER OF THE
5960                                                                  ;STRING SMALLER THAN THE LAST?
5961  032320  002220                          BGE     GTTST1          ;NO--BRANCH
5962  032322  010246                          MOV     R2,-(SP)        ;SAVE ENDING TEST NUMBER
5963  032324  013702  032406                  MOV     9$,R2           ;GET STARTING TEST NUMBER
5964  032330  012637  032406                  MOV     (SP)+,9$        ;STORE ENDING TEST NUMBER
5965  032334  006337  032406                  ASL     9$              ;SHIFT ENDING TEST NUMBER
5966  032340  006302                          ASL     R2              ;SHIFT TEST NUMBER
5967  032342  010204                  8$:     MOV     R2,R4           ;COPY TEST NUMBER INTO R4
5968  032344  042704  000037                  BIC     #37,R4          ;CLEAR LOWER BITS
5969  032350  006204                          ASR     R4              ;SHIFT THE TEST NUMBER
5970  032352  006204                          ASR     R4              ;SHIFT THE TEST NUMBER
5971  032354  006204                          ASR     R4              ;SHIFT THE TEST NUMBER
5972  032356  006204                          ASR     R4              ;SHIFT THE TEST NUMBER
5973  032360  056264  001424  001234          BIS     BITS(R2),TSTNMS(R4)  ;SELECT THE TEST
5974  032366  062702  000002                  ADD     #2,R2           ;INCREMENT THE TEST NUMBER
5975  032372  020237  032406                  CMP     R2,9$           ;SEE IF FINISHED
5976  032376  101761                          BLOS    8$              ;BR IF NOT
5977  032400  162702  000002                  SUB     #2,R2           ;CORRECT TEST NUMBER
5978  032404  000402                          BR      GTTST4          ;CONTINUE
```

```
5979  032406  000000                  9$:     .WORD   0               ;STORE TEST NUMBER HERE
5980  032410  005201                  GTTST3: INC     R1              ;MOVE TO NEXT CHARACTER
5981  032412  121127  000056          GTTST4: CMPB    (R1),#'.        ;"PERIOD"?
5982  032416  001441                          BEQ     GTTST5          ;YES--BRANCH
5983  032420  005737  001234                  TST     TSTNMS          ;ANY TEST SELECTED THIS CYCLE?
5984  032424  001005                          BNE     1$              ;BR IF YES
5985  032426  005737  001236                  TST     TSTNMS+2        ;ANY TEST SELECTED THIS CYCLE ?
5986  032432  001002                          BNE     1$              ;BR IF YES
5987  032434  000137  031762                  JMP     GTTST1          ;NO
5988  032440  121127  000057          1$:     CMPB    (R1),#'/        ;"OPEN"?
5989  032444  001004                          BNE     2$              ;NO--BRANCH
5990  032446  056264  001424  001240          BIS     BITS(R2), OPNFLG(R4)  ;YES--SET BITS FOR TEST TO OPEN
5991  032454  000405                          BR      3$
5992  032456  121127  000054          2$:     CMPB    (R1),#',        ;"COMMA"?
5993  032462  001402                          BEQ     3$              ;BR IF YES
5994  032464  000137  031762                  JMP     GTTST1          ;NO
5995  032470  005201                  3$:     INC     R1              ;MOVE TO NEXT CHARACTER
5996  032472  105711                          TSTB    (R1)            ;"CR"?
5997  032474  001402                          BEQ     4$              ;BR IF 'CR'
5998  032476  000137  032046                  JMP     GTTST2          ;NO--GO GET NEXT CHARACTER
5999  032502  005737  001240          4$:     TST     OPNFLG          ;ANY TESTS TO OPEN ?
6000  032506  001042                          BNE     OPNTST          ;BR IF YES
6001  032510  005737  001242                  TST     OPNFLG+2        ;ANY TESTS TO OPEN ?
6002  032514  001037                          BNE     OPNTST          ;BR IF YES
6003  032516  000137  031762                  JMP     GTTST1          ;NO--START AGAIN
6004  032522  005201                  GTTST5: INC     R1              ;MOVE TO NEXT CHARACTER
6005  032524  121127  000056                  CMPB    (R1),#'.        ;"PERIOD"?
6006  032530  001414                          BEQ     GTTST6          ;YES--BRANCH
6007  032532  105711                          TSTB    (R1)            ;"CR"?
6008  032534  001402                          BEQ     1$              ;YES--BRANCH
6009  032536  000137  031762                  JMP     GTTST1
6010  032542  005737  001240          1$:     TST     OPNFLG          ;ANY TESTS TO OPEN ?
6011  032546  001022                          BNE     OPNTST          ;BR IF YES
6012  032550  005737  001242                  TST     OPNFLG+2        ;ANY TESTS TO OPEN ?
6013  032554  001017                          BNE     OPNTST          ;BR IF YES
6014  032556  000137  031756                  JMP     GT.PR2          ;NO--GO START TESTING
6015  032562  005201                  GTTST6: INC     R1              ;MOVE TO NEXT CHARACTER
6016  032564  105711                          TSTB    (R1)            ;"CR"?
6017  032566  001402                          BEQ     1$              ;YES--BRANCH
6018  032570  000137  031762                  JMP     GTTST1          ;NO--GO ASK FOR TEST
6019  032574  005737  001240          1$:     TST     OPNFLG          ;ANY TESTS TO OPEN ?
6020  032600  001005                          BNE     OPNTST          ;BR IF YES
6021  032602  005737  001242                  TST     OPNFLG+2        ;ANY TESTS TO OPEN ?
6022  032606  001002                          BNE     OPNTST          ;BR IF YES
6023  032610  000137  031756                  JMP     GT.PR2          ;NO--GO START TESTING
6024
6025                                  ;OPEN THE SELECTED TEST FOR CHANGES
6026
6027  032614  104412                  OPNTST: SAVREG                  ;SAVE R0 - R5
6028  032616  005027                          CLR     (PC)+           ;START WITH TEST 0
6029  032620  000000                  OPN.CT: .WORD   0               ;COUNT STORED HERE
6030  032622  000411                          BR      OPN.2           ;SKIP THE INCREMENT
6031  032624  005237  032620          OPN.1:  INC     OPN.CT          ;MOVE TO THE NEXT TEST
6032  032630  022737  000022  032620          CMP     #STN-1,OPN.CT   ;TEST NUMBER TOO BIG?
6033  032636  002003                          BGE     OPN.2           ;NO--OPEN THE NEXT TEST
6034  032640  104413                          RESREG                  ;RESTORE R0 - R5
```

```
6035  032642  000137  031762                    JMP     GTTST1          ;YES--GO ASK FOR MORE TESTS
6036  032646  013705  032620          OPN.2:    MOV     OPN.CT,R5       ;SETUP TO USE THE
6037  032652  006305                            ASL     R5              ;TEST NUMBER AS AN INDEX
6038  032654  013703  032620                    MOV     OPN.CT,R3       ;GET INDEX
6039  032660  042703  000017                    BIC     #17,R3          ;CLEAR LOWER TEST BITS
6040  032664  006203                            ASR     R3              ;SHIFT TEST NUMBER
6041  032666  006203                            ASR     R3              ;SHIFT TEST NUMBER
6042  032670  006203                            ASR     R3              ;SHIFT TEST NUMBER
6043  032672  036563  001424  001240            BIT     BITS(R5),OPNFLG(R3)  ;OPEN THIS TEST?
6044  032700  001751                            BEQ     OPN.1           ;NO--MOVE TO NEXT TEST
6045  032702  104401  032710                    TYPE    ,65$            ;;TYPE ASCIZ STRING
6046  032706  000404                            BR      64$             ;;GET OVER THE ASCIZ
6047                                  ;;65$:    .ASCIZ  / TEST /
6048  032720                        64$:
6049  032720  013746  032620                    MOV     OPN.CT,-(SP)    ;;SAVE OPN.CT FOR TYPEOUT
6050                                                                    ;;TEST NUMBER
6051  032724  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
6052  032726     002                            .BYTE   2               ;;TYPE 2 DIGIT(S)
6053  032727     000                            .BYTE   0               ;;SUPPRESS LEADING ZEROS
6054  032730  104401  001215                    TYPE    ,$CRLF          ;TYPE "CR" & "LF"
6055  032734  016500  001536                    MOV     PRMPT(R5),R0    ;PICKUP PARAMETER POINTER
6056  032740  011046                            MOV     (R0),-(SP)      ;SAVE THE VARIABLE INDICATOR
6057  032742  012702  001504                    MOV     #PRM,R2         ;FIRST ADDRESS OF TABLE
6058  032746  000405                            BR      2$
6059  032750  006216                  1$:       ASR     (SP)            ;CHECK FOR A VARIABLE
6060  032752  103403                            BCS     2$              ;GO MOVE THIS ONE
6061  032754  001404                            BEQ     OPNPRM          ;DONE
6062  032756  005722                            TST     (R2)+           ;BUMP THE POINTER
6063  032760  000773                            BR      1$
6064  032762  012022                  2$:       MOV     (R0)+,(R2)+     ;MOVE THIS VARIABLE INTO THE
6065  032764  000771                            BR      1$              ;COMMON AREA
6066  032766  013716  001504          OPNPRM:   MOV     @#PRM,(SP)      ;GET THE VARIABLE INDICATOR
6067  032772  005004                            CLR     R4              ;ZERO THE INDEX
6068  032774  006216                  1$:       ASR     (SP)            ;CHECK FOR A VARIABLE
6069  032776  103403                            BCS     3$              ;GO GET IT
6070  033000  001772                            BEQ     OPNPRM          ;OUT OF VARIABLES
6071  033002  005724                  2$:       TST     (R4)+           ;UPDATE THE INDEX
6072  033004  000773                            BR      1$
6073  033006  005764  001606          3$:       TST     PRMLMT(R4)      ;IS THE MAX. MAGNITUDE NEG?
6074  033012  100466                            BMI     OPNPAT          ;YES--THEN IT IS THE PATTERN
6075  033014  104401  044010                    TYPE    ,MSG.SP         ;TYPE SPACES
6076  033020  016437  001636  033030            MOV     PRMMSG(R4),4$   ;TYPE THE NAME OF THIS VARIABLE
6077  033026  104401                            TYPE
6078  033030  000000                  4$:       .WORD   0
6079  033032  104401  043016                    TYPE    ,MSG.EQ         ;TYPE "="
6080  033036  016446  001506                    MOV     RPT(R4),-(SP)           ;PUT RPT(R4) ON THE STACK
6081  033042  004737  023170                    JSR     PC,@#$S82D      ;CHANGE RPT(R4) TO DECIMAL ASCIZ
6082  033046  004737  023420                    JSR     PC,@#$SUPRS     ;TYPE WITHOUT LEADING ZEROS
6083  033052  104401  043044                    TYPE    ,SLASH          ;' / '
6084  033056  104411                            RDLIN
6085  033060  012601                            MOV     (SP)+,R1        ;READ AN ASCIZ STRING
6086  033062  004037  033736                    JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
6087  033066  033306                            3$                      ;ILLEGAL CHARACTER
6088  033070  033002                            2$                      ;CARRIAGE RETURN
6089  033072  033140                            8$                      ;"/"
6090  033074  033102                            5$                      ;","
```

```
6091  033076  033110                        6$                      ;" "
6092  033100  033136                        7$                      ;DIGIT 0-9
6093  033102  105711              5$:       TSTB    (R1)            ;"CR"?
6094  033104  001340                        BNE     3$              ;NO--STAY ON THIS VARIABLE
6095  033106  000735                        BR      2$              ;YES--MOVE TO NEXT VARIABLE
6096  033110  105711              6$:       TSTB    (R1)            ;IS THERE A "CR" AFTER THE PERIOD?
6097  033112  001002                        BNE     64$             ;NO
6098  033114  000137  033530                JMP     OPN.N2          ;YES--GO CLOSE THIS TEST
6099  033120  122721  000056      64$:      CMPB    #'.,(R1)+       ;DOUBLE PERIOD?
6100  033124  001330                        BNE     3$              ;NO--GO ASK FOR THIS VARIABLE
6101  033126  105711                        TSTB    (R1)            ;YES--IS A "CR" AFTER THE DOUBLE PERIOD?
6102  033130  001326                        BNE     3$              ;NO--ASK FOR THIS VARIABLE AGAIN
6103  033132  000137  033546              JMP     OPN.X2          ;YES--CLOSE ALL TEST
6104  033136  005301              7$:       DEC     R1              ;BACK THE POINTER UP BY ONE
6105  033140                      8$:
6106  033140  016402  001606              MOV     PRMLMT(R4),R2   ;UPPER LIMIT OF INPUT
6107  033144  004037  034012              JSR     R0,@#CK.DIG     ;CHECK THE DIGIT(S)
6108  033150  033006                        3$                      ;ILLEGAL INPUT
6109  033152  033006                        3$                      ;INPUT TO LARGE
6110  033154  033162                        9$                      ;TERMINATED WITH A "," OR "CR"
6111  033156  033524                        OPN.N1                  ;TERMINATED WITH A ":"
6112  033160  033542                        OPN.X1                  ;TERMINATED WITH A ".."
6113  033162  010264  001506      9$:       MOV     R2,RPT(R4)      ;SAVE THIS VARIABLE
6114  033166  000705                        BR      2$              ;MOVE TO NEXT VARIABLE
6115  033170  104401  044010      OPNPAT:   TYPE    ,MSG.SP         ;TYPE SPACES
6116  033174  104401  043012              TYPE    ,MSG.PAT        ;TYPE "PAT"
6117  033200  104401  043016              TYPE    ,MSG.EQ         ;TYPE "="
6118  033204  016446  001506              MOV     RPT(R4),-(SP)   ;;SAVE RPT(R4) FOR TYPEOUT
6119  033210  104402                      TYPOC                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6120  033212  104401  044011              TYPE    ,MSG.SP+1       ;TYPE ONE SPACE
6121  033216  104411                      RDLIN                   ;READ ASCIZ STRING
6122  033220  012601                      MOV     (SP)+,R1        ;PICKUP POINTER
6123  033222  004037  033736              JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
6124  033226  033170                      OPNPAT                  ;ILLEGAL CHARACTER
6125  033230  032766                      OPNPRM                  ;CARRIAGE RETURN
6126  033232  033264                      3$                      ;"/"
6127  033234  032766                      OPNPRM                  ;"."
6128  033236  033242                      1$                      ;"'"
6129  033240  033262                      2$                      ;DIGIT 0-9
6130  033242  105711              1$:       TSTB    (R1)            ;"CR" AFTER THE PERIOD?
6131  033244  001531                        BEQ     OPN.N2          ;YES--GO CLOSE THIS TEST
6132  033246  122721  000056              CMPB    #'.,(R1)+       ;NO--PERIOD?
6133  033252  001346                        BNE     OPNPAT          ;NO--LOOP
6134  033254  105711                        TSTB    (R1)            ;"CR" AFTER A DOUBLE PERIOD?
6135  033256  001533                        BEQ     OPN.X2          ;YES--GO START TESTING
6136  033260  000743                        BR      OPNPAT          ;NO--LOOP
6137  033262  005301              2$:       DEC     R1              ;BACKUP THE ASCII POINTER
6138  033264                      3$:
6139  033264  004037  034176              JSR     R0,@#CK.NUM     ;CHECK THE NUMBER
6140  033270  033170                      OPNPAT                  ;ILLEGAL INPUT
6141  033272  033300                      4$                      ;TERMINATED WITH A "," OR "CR"
6142  033274  033524                      OPN.N1                  ;TERMINATED WITH A ":"
6143  033276  033542                      OPN.X1                  ;TERMINATED WITH A ".."
6144  033300  010264  001506      4$:       MOV     R2,RPT(R4)      ;SAVE THE INPUT NUMBER
6145  033304  006002                        ROR     R2              ;OPEN PATTERN 0?
6146  033306  103227                        BCC     OPNPRM          ;NO--START AT BEGINNING OF PARAMETER TABLE
```

```
6147  033312  104412                  SAVREG                          ;SAVE R0 - R5
6148  033312  005000          OPNWDS: CLR     R0                      ;START WITH WORD 0
6149  033314  012704  003104          MOV     #PAT0,R4
6150  033320                  1$:
6151  033320  104401  033326          TYPE    ,65$                    ;;TYPE ASCIZ STRING
6152  033324  000403                  BR      64$                     ;;GET OVER THE ASCIZ
6153                          ;;65$:  .ASCIZ  /   WD/
6154  033334                  64$:
6155  033334  010046                  MOV     R0,-(SP)                        ;PUT R0 ON THE STACK
6156  033336  004737  023170          JSR     PC,@#$SB2D              ;CHANGE R0 TO DECIMAL ASCIZ
6157  033342  004737  023420          JSR     PC,@#$SUPRS             ;TYPE WITHOUT LEADING ZEROS
6158  033346  104401  043016          TYPE    ,MSG.EQ                 ;TYPE "="
6159  033352  011446                  MOV     (R4),-(SP)              ;;SAVE (R4) FOR TYPEOUT
6160  033354  104402                  TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
6161  033356  104411                  RDLIN                           ;READ ASCIZ STRING
6162  033360  012601                  MOV     (SP)+,R1                ;PICKUP THE POINTER
6163  033362  004037  033736          JSR     R0,@#CK.CHR             ;CHECK ONE CHARACTER
6164  033366  033320                  1$                              ;ILLEGAL CHARACTER
6165  033370  033422                  4$                              ;CARRIAGE RETURN
6166  033372  033404                  2$                              ;"/"
6167  033374  033422                  4$                              ;","
6168  033376  033436                  5$                              ;"."
6169  033400  033402                  10$                             ;DIGIT 0-9
6170  033402  005301          10$:    DEC     R1                      ;BACKUP THE ASCII POINTER
6171  033404                  2$:
6172  033404  004037  034176          JSR     R0,@#CK.NUM             ;CHECK THE NUMBER
6173  033410  033320                  1$                              ;ILLEGAL INPUT
6174  033412  033420                  3$                              ;TERMINATED WITH A "," OR "CR"
6175  033414  033456                  6$                              ;TERMINATED WITH A "/"
6176  033416  033470                  8$                              ;TERMINATED WITH A ".."
6177  033420  010214          3$:     MOV     R2,(R4)                 ;SAVE THE INPUT
6178  033422  005724          4$:     TST     (R4)+                   ;MOVE TO NEXT WORD
6179  033424  005200                  INC     R0                      ;INCREMENT THE COUNT
6180  033426  022700  000020          CMP     #16.,R0                 ;COUNT TO LARGE?
6181  033432  003332                  BGT     1$                      ;NO--BRANCH
6182  033434  000726                  BR      OPNWDS                  ;YES--BRANCH
6183  033436  105711          5$:     TSTB    (R1)                    ;"CR" AFTER THE PERIOD?
6184  033440  001407                  BEQ     7$                      ;YES--GO CLOSE THIS TEST
6185  033442  122721  000056          CMPB    #'.,(R1)+               ;NO--PERIOD?
6186  033446  001324                  BNE     1$                      ;NO--BRANCH ILLEGAL INPUT STRING
6187  033450  105711                  TSTB    (R1)                    ;"CR" AFTER THE "PERIOD-PERIOD"?
6188  033452  001407                  BEQ     9$                      ;YES--GO START TESTING
6189  033454  000721                  BR      1$                      ;NO--LOOP
6190  033456  010224          6$:     MOV     R2,(R4)+                ;SAVE THE INPUT
6191  033460  004737  033502  7$:     JSR     PC,@#CLSWDS             ;CLOSE THE DATA PATTERN
6192  033464  104413                  RESREG                          ;RESTORE R0 - R5
6193  033466  000420                  BR      OPN.N2                  ;MOVE TO NEXT TEST
6194  033470  010224          8$:     MOV     R2,(R4)+                ;SAVE THE INPUT
6195  033472  004737  033502  9$:     JSR     PC,@#CLSWDS             ;CLOSE THE DATA PATTERN
6196  033476  104413                  RESREG                          ;RESTORE R0 - R5
6197  033500  000422                  BR      OPN.X2                  ;START TESTING
6198  033502  012701  003104  CLSWDS: MOV     #PAT0,R1                ;FIRST ADDRESS OF DATA PATTERN
6199  033506  005200          1$:     INC     R0                      ;COUNT THE LAST WORD THAT WAS STORED
6200  033510  022700  000017          CMP     #15.,R0                 ;END OF TABLE
6201  033514  002402                  BLT     2$                      ;YES--EXIT
6202  033516  012124                  MOV     (R1)+,(R4)+             ;COPY
```

```
 6203  033520  000772                    BR      1$              ;LOOP
 6204  033522  000207                2$: RTS     PC              ;RETURN
 6205  033524  010264  001506      OPN.N1: MOV   R2,RPT(R4)       ;SAVE THIS VARIABLE
 6206  033530  005726              OPN.N2: TST   (SP)+           ;CLEAN OFF THE STACK
 6207  033532  004737  033602            JSR   PC,CLOSE          ;CLOSE THIS TEST
 6208  033536  000137  033524            JMP   OPN               ;GO OPEN THE NEXT TEST
 6209  033542  010264  001506      OPN.X1: MOV   R2,RPT(R4)       ;SAVE THIS VARIABLE
 6210  033546  005726              OPN.X2: TST   (SP)+           ;CLEAN OFF THE STACK
 6211  033550  004737  033602        1$:   JSR   PC,CLOSE        ;CLOSE THIS TEST
 6212  033554  005726                2$:   TST   (R5)+           ;UPDATE THE INDEX
 6213  033556  020527  000034            CMP   R5,#16*2          ;INDEX TO BIG?
 6214  033562  002403                      BLT   3$              ;NO--BRANCH
 6215  033564  104413                      RESREG                ;RESTORE R0 - R5
 6216  033566  000137  031756            JMP   GT.PR2            ;GO TO EXIT
 6217  033572  036503  001424        3$:   BIT   BITS(R5),F3     ;IS THIS TEST OPEN FOR CHANGE?
 6218  033576  001364                      BNE   1$              ;YES--GO CLOSE IT
 6219  033600  000765                      BR    2$              ;NO--MOVE TO NEXT TEST
 6220  033602  104412              CLOSE:  SAVREG                ;SAVE R0 - R5
 6221  033604  012700  001504            MOV   #PRM,R0           ;"FROM" ADDRESS
 6222  033610  016101  001536            MOV   PRMPT(R5),R1      ;"TO" ADDRESS
 6223  033614  012002                      MOV   (R0)+,R2        ;"FROM" INDICATOR
 6224  033616  012103                      MOV   (R1)+,R3        ;"TO" INDICATOR
 6225  033620  012704  000001            MOV   #1,R4             ;TEST BIT START A "RPT"
 6226  033624  030402              1$:   BIT   R4,R2             ;PARAMETER TO BE MOVED?
 6227  033626  001403                      BEQ   2$              ;NO--BRANCH
 6228  033630  030403                      BIT   R4,R3           ;A PLACE TO PUT IT?
 6229  033632  001404                      BEQ   3$              ;NO--BRANCH
 6230  033634  011011                      MOV   (R0),(R1)       ;YES--MOVE "FROM" TO "TO"
 6231  033636  030403              2$:   BIT   R4,R3             ;"TO" PARAMETER?
 6232  033640  001401                      BEQ   3$              ;NO--BRANCH
 6233  033642  005721                      TST   (R1)+           ;YES--UPDATE THE POINTER
 6234  033644  005720              3$:   TST   (R0)+             ;UPDATE FROM POINTER
 6235  033646  006304                      ASL   R4              ;POSITION THE TEST BIT
 6236  033650  032704  002000            BIT   #BIT10,R4         ;DONE?
 6237  033654  001763                      BEQ   1$              ;NO--BRANCH
 6238  033656  104413                      RESREG                ;RESTORE R0 - R5
 6239  033660  000207                      RTS   PC              ;RETURN
                                ;THIS ROUTINE IS USED TO CHECK IF AN
                                ;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
                                ;CALL
                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                ;       JSR     R0,@#CK.OCT     ;CHECK THE CHARACTER
                                ;       RETURN1                 ;CHARACTER IS NOT BETWEEN 0-7
                                ;       RETURN2                 ;CHARACTER IS IN R2 AS A
                                ;                               ;OCTAL DIGIT

 6256  033662  121127  000060  CK.OCT: CMPB  (R1),#'0          ;LESS THAN ZERO?
 6257  033666  103407                    BLO   1$              ;YES -- BRANCH
 6258  033670  121127  000067          CMPB  (R1),#'7          ;GREATER THAN SEVEN?
 6259  033674  101004                    BHI   1$              ;YES -- BRANCH
 6260  033676  111102                    MOVB  (R1),R2         ;GET THE CHARACTER
 6261  033700  042702  177770          BIC   #^C7,R2           ;STRIP AWAY THE ASCII
 6262  033704  005720                    TST   (R0)+           ;ADJUST FOR RETURN
 6263  033706  000200              1$:   RTS   R0                ;RETURN
 6264
 6265                           ;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
```

```
                                        ;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
                                        ;CALL
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                        ;       JSR     R0,@#CK.DEC     ;CHECK THE CHARACTER
                                        ;       RETURN                  ;NOT BETWEEN 0 AND 9
                                        ;       RETURN.                 ;BETWEEN 0 AND 9
                                        ;                               ;R2 = DIGIT

033710  121127  000060        CK.DEC: CMPB    (R1),#'0        ;LESS THAN ZERO?
033714  103407                        BLO     1$              ;YES -- BRANCH
033716  121127  000071                CMPB    (R1),#'9                ;GREATER THAN NINE?
033722  101004                        BHI     1$              ;YES -- BRANCH
033724  111102                        MOVB    (R1),R2         ;GET THE CHARACTER
033726  042702  000060                BIC     #'0,R2          ;STRIP AWAY THE ASCII
033732  005720                        TST     (R0)+           ;ADJUST FOR RETURN
033734  000200                1$:     RTS     R0              ;RETURN

                                        ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
                                        ;DETERMINE WHAT IT IS.
                                        ;CALL
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                        ;       JSR     R0,@#CK.CHR     ;CHECK CHARACTER
                                        ;       RETURN  ADR1            ;UNKNOWN CHARACTER
                                        ;       RETURN  ADR2            ;CARRIAGE RETURN * (R1)=ADR+1
                                        ;       RETURN  ADR3            ;SLASH * (R1)=ADR+1
                                        ;       RETURN  ADR4            ;COMMA * (R1)=ADR+1
                                        ;       RETURN  ADR5            ;PERIOD * (R1)=ADR+1
                                        ;       RETURN  ADR6            ;DIGIT BETWEEN 0 AND 9.
                                        ;                               ;R2 = DIGIT * (R1)=ADR+1

033736  105711                CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
033740  001420                        BEQ     4$              ;YES -- BRANCH
033742  121127  000057                CMPB    (R1),#'/        ;"SLASH"?
033746  001414                        BEQ     3$              ;YES -- BRANCH
033750  121127  000054                CMPB    (R1),#',        ;"COMMA"?
033754  001410                        BEQ     2$              ;YES -- BRANCH
033756  121127  000056                CMPB    (R1),#'.        ;"PERIOD"?
033762  001404                        BEQ     1$              ;YES -- BRANCH
033764  004037  033710                JSR     R0,@#CK.DEC     ;"DIGIT"?
033770  000406                        BR      5$              ;NO -- BRANCH
033772  005720                        TST     (R0)+           ;DIGIT BETWEEN 0-9
033774  005720                1$:     TST     (R0)+           ;PERIOD
033776  005720                2$:     TST     (R0)+           ;COMMA
034000  005720                3$:     TST     (R0)+           ;SLASH
034002  005720                4$:     TST     (R0)+           ;CARRIAGE RETURN
034004  005201                        INC     R1              ;MOVE POINTER TO NEXT CHARACTER
034006  011000                5$:     MOV     (R0),R0         ;UNKNOWN CHARACTER
034010  000200                        RTS     R0              ;RETURN

                                        ;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
                                        ;CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.
                                        ;CALL
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
                                        ;       MOV     #NUM,R2         ;MAX. MAGNITUDE OF INPUT NUMBER
                                        ;       JSR     R0,@#CK.DIG     ;CHECK DIGITS
                                        ;       RETURN  ADR1            ;ILLEGAL CHARACTER -- R2=?
```

```
                                        ;       RETURN  ADR2            ;INPUT NUMBER TO LARGE -- R2=?
                                        ;       RETURN  ADR3            ;"COMMA" -- R2 = NUMBER
                                        ;       RETURN  ADR4            ;"PERIOD" -- R2 = NUMBER
                                        ;       RETURN  ADR5            ;"PERIOD-PERIOD" -- R2 = NUMBER

034012  010446                        CK.DIG: MOV     R4,-(SP)        ;SAVE R4
034014  010346                                MOV     R3,-(SP)        ;SAVE R3
034016  010246                                MOV     R2,-(SP)        ;SAVE THE MAX. SIZE ON THE STACK
034020  005002                                CLR     R2              ;START WITH 0
034022  005003                                CLR     R3
034024  005004                                CLR     R4
034026  004037  033736                        JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
034032  034162                                BS                      ;ILLEGAL CHARACTER
034034  034162                                BS                      ;CARRIAGE RETURN
034036  034162                                BS                      ;"/"
034040  034162                                BS                      ;"-"
034042  034162                                BS                      ;","
034044  034046                                1$                      ;DIGIT 0-9
034046  006303                        1$:     ASL     R3              ;2
034050  010346                                MOV     R3,-(SP)        ;SAVE *2
034052  006303                                ASL     R3              ;4
034054  006303                                ASL     R3              ;8
034056  062603                                ADD     (SP)+,R3        ;(*8)+(*2)=*10.
034060  060203                                ADD     R2,R3           ;UPDATE THE INPUT NUMBER
034062  004037  033736                        JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
034066  034162                                BS                      ;ILLEGAL CHARACTER
034070  034162                                BS                      ;CARRIAGE RETURN
034072  034162                                BS                      ;"/"
034074  034110                                3$                      ;"-"
034076  034106                                2$                      ;","
034100  034046                                1$                      ;DIGIT 0-9
034102  005301                        9$:     DEC     R1              ;BACKUP THE CHARACTER POINTER
034104  000401                                BR      3$              ;CONTINE
034106  005724                        2$:     TST     (R4)+           ;"PERIOD"
034110  005724                        3$:     TST     (R4)+           ;"COMMA" OR "CR"
034112  004037  033736                        JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
034116  034162                                BS                      ;ILLEGAL CHARACTER
034120  034162                                BS                      ;CARRIAGE RETURN
034122  034162                                BS                      ;"/"
034124  034162                                BS                      ;"-"
034126  034132                                4$                      ;","
034130  034142                                5$                      ;DIGIT 0-9
034132  005724                        4$:     TST     (R4)+           ;"PERIOD-PERIOD"
034134  105711                                TSTB    (R1)            ;"CR"?
034136  001405                                BEQ     6$              ;YES--BRANCH
034140  000410                                BR      8$
034142  126127  177776  000054        5$:     CMPB    -2(R1),#',      ;WAS CHARACTER BEFORE THE DIGIT A COMMA?
034150  001004                                BNE     8$              ;NO--EXIT
034152  020316                        6$:     CMP     R3,(SP)         ;INPUT TO LARGE?
034154  101001                                BHI     7$              ;YES -- BRANCH
034156  060400                                ADD     R4,R0           ;ADJUST RETURN ADDRESS
034160  005720                        7$:     TST     (R0)+
034162  010302                        8$:     MOV     R3,R2           ;NUMBER TO R2
034164  005726                                TST     (SP)+           ;CLEAN MAX. SIZE OFF OF STACK
034166  012603                                MOV     (SP)+,R3        ;RESTORE R3
034170  012604                                MOV     (SP)+,R4        ;RESTORE R4
```

```
                        *** PROGRAM SUBROUTINES ***

6371  034172  011000                MOV     (R0),R0         ;GET RETURN ADDRESS
6372  034174  000200                RTS     R0              ;RETURN
6373
6374                          ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
6375                          ;AND FORMS AN OCTAL NUMBER IN R2
6376                          ;CALL:
6377                          ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
6378                          ;       JSR     R0,@#CK.NUM     ;GO FORM THE NUMBER
6379                          ;       RETURN  ADR1            ;ILLEGAL CHARACTER IN THE INPUT STRING
6380                          ;       RETURN  ADR2            ;"COMMA" OR "CR"--R2=NUMBER
6381                          ;       RETURN  ADR3            ;"PERIOD"--R2=NUMBER
6382                          ;       RETURN  ADR4            ;"PERIOD-PERIOD"--R2=NUMBER
6383
6384  034176  010346          CK.NUM: MOV     R3,-(SP)        ;SAVE R3
6385  034200  005003                  CLR     R3              ;START NUMBER AT ZERO
6386  034202  004037  033662          JSR     R0,@#CK.OCT     ;OCTAL DIGIT?
6387  034206  000440                  BR      6$              ;NO--BRANCH
6388  034210  005201          1$:     INC     R1              ;MOVE TO NEXT CHARACTER
6389  034212  006303                  ASL     R3              ;FOR THE OCTAL NUMBER IN R3
6390  034214  103435                  BCS     6$              ;DON'T LET IT GET TO BIG
6391  034216  006303                  ASL     R3
6392  034220  103433                  BCS     6$
6393  034222  006303                  ASL     R3
6394  034224  103431                  BCS     6$
6395  034226  060203                  ADD     R2,R3
6396  034230  004037  033662          JSR     R0,@#CK.OCT     ;IS THIS AN OCTAL DIGIT?
6397  034234  000401                  BR      2$              ;NO--FIND OUT WHAT IT IS
6398  034236  000764                  BR      1$              ;YES--MAKE IT PART OF THE NUMBER
6399  034240  010302          2$:     MOV     R3,R2           ;SAVE THE OCTAL NUMBER
6400  034242  005003                  CLR     R3              ;START WITH ZERO INDEX
6401  034244  004037  033736          JSR     R0,@#CK.CHR     ;CHECK ONE CHARACTER
6402  034250  034310                  6$                      ;ILLEGAL CHARACTER
6403  034252  034300                  5$                      ;CARRIAGE RETURN
6404  034254  034310                  6$                      ;"/"
6405  034256  034300                  5$                      ;"-"
6406  034260  034264                  3$                      ;"."
6407  034262  034310                  6$                      ;DIGIT 0-9
6408  034264  005723          3$:     TST     (R3)+           ;"PERIOD"
6409  034266  121127  000056          CMPB    (R1),#'.        ;"PERIOD-PERIOD"?
6410  034272  001002                  BNE     5$              ;NO--BRANCH
6411  034274  005201                  INC     R1              ;YES--ADVANCE THE POINTER
6412  034276  005723          4$:     TST     (R3)+           ;"PERIOD-PERIOD"
6413  034300  005723          5$:     TST     (R3)+           ;"COMMA"
6414  034302  105711                  TSTB    (R1)            ;"CR"?
6415  034304  001001                  BNE     6$              ;NO--BRANCH
6416  034306  060300                  ADD     R3,R0           ;YES--SAVE THE OCTAL NUMBER
6417  034310  012603          6$:     MOV     (SP)+,R3        ;RESTORE R3
6418  034312  011000                  MOV     (R0),R0         ;PICKUP EXIT ADDRESS
6419  034314  000200                  RTS     R0              ;RETURN
```

```
                                ;;*****************************************************************

                                .SBTTL  SINGLE/DUAL PORT RH11/RPO4/5/6 DRIVER (REV 1.0)

                                ;COPYRIGHT (C) 1976
                                ;DIGITAL EQUIPMENT CORP.
                                ;MAYNARD, MA 01754
                                ;AUTHOR(S): JIM LACEY/CHUCK HESS

                                ;;*****************************************************************

                                ;STORAGE FOR RPDS1, RPER1, RPER2, AND RPER3 ON AN ERROR "E"
                                        ;RPERRS   = RPDS1
                                        ;RPERRS+2 = RPER1
                                        ;RPERRS+4 = RPER2
                                        ;RPERRS+6 = RPER3

034316  000000  000000  000000  RPERRS: .WORD   0,0,0,0
034324  000000

                                ;TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)
                                        ;DRVACT=0 IF DRIVE IS IDLE
                                        ;DRVACT>0 IF DRIVE IS ACTIVE WITH A COMMAND
                                        ;DRVACT<0 IF DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION

034326      000                 DRVACT: .BYTE   0               ;DRIVE 0
034327      000                         .BYTE   0               ;DRIVE 1
034330      000                         .BYTE   0               ;DRIVE 2
034331      000                         .BYTE   0               ;DRIVE 3
034332      000                         .BYTE   0               ;DRIVE 4
034333      000                         .BYTE   0               ;DRIVE 5
034334      000                         .BYTE   0               ;DRIVE 6
034335      000                         .BYTE   0               ;DRIVE 7

                                ;TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
                                        ;DRVSTA=0 IF DRIVE IS OFFLINE OR NONEXSITENT
                                        ;DRVSTA>0 IF DRIVE IS ONLINE
                                        ;DRVSTA<0 IF DRIVE IS UNSAFE

034336      000                 DRVSTA: .BYTE   0               ;DRIVE 0
034337      000                         .BYTE   0               ;DRIVE 1
034340      000                         .BYTE   0               ;DRIVE 2
034341      000                         .BYTE   0               ;DRIVE 3
034342      000                         .BYTE   0               ;DRIVE 4
034343      000                         .BYTE   0               ;DRIVE 5
034344      000                         .BYTE   0               ;DRIVE 6
034345      000                         .BYTE   0               ;DRIVE 7

                                ;TABLE OF DRIVE TYPES (DRVTYP=8 BYTES)
                                        ;DRVTYP=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
                                        ;DRVTYP=1 IF DRIVE IS RPO4
                                        ;DRVTYP=2 IF DRIVE IS RPO5
                                        ;DRVTYP=4 IF DRIVE IS RPO6
                                        ;DRVTYP=-1 IF NOT RPO4/5/6
```

```
6476  034346  000                 DRVTYP: .BYTE   0               ;DRIVE 0
6477  034347  000                         .BYTE   0               ;DRIVE 1
6478  034350  000                         .BYTE   0               ;DRIVE 2
6479  034351  000                         .BYTE   0               ;DRIVE 3
6480  034352  000                         .BYTE   0               ;DRIVE 4
6481  034353  000                         .BYTE   0               ;DRIVE 5
6482  034354  000                         .BYTE   0               ;DRIVE 6
6483  034355  000                         .BYTE   0               ;DRIVE 7
6484
6485                              ;TABLE OF DUAL PORT INITIALIZATION INDICATORS
6486                                      ;DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
6487                                      ;DPINT<0 IF INITIALIZATION IS IN PROGRESS
6488
6489  034356  000                 DPINT:  .BYTE   0               ;DRIVE 0
6490  034357  000                         .BYTE   0               ;DRIVE 1
6491  034360  000                         .BYTE   0               ;DRIVE 2
6492  034361  000                         .BYTE   0               ;DRIVE 3
6493  034362  000                         .BYTE   0               ;DRIVE 4
6494  034363  000                         .BYTE   0               ;DRIVE 5
6495  034364  000                         .BYTE   0               ;DRIVE 6
6496  034365  000                         .BYTE   0               ;DRIVE 7
6497
6498                              ;TABLE OF PENDING DUAL PORT REQUESTS
6499                                      ;DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
6500                                      ;DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE
6501
6502  034366  000                 DPRQS:  .BYTE   0               ;DRIVE 0
6503  034367  000                         .BYTE   0               ;DRIVE 1
6504  034370  000                         .BYTE   0               ;DRIVE 2
6505  034371  000                         .BYTE   0               ;DRIVE 3
6506  034372  000                         .BYTE   0               ;DRIVE 4
6507  034373  000                         .BYTE   0               ;DRIVE 5
6508  034374  000                         .BYTE   0               ;DRIVE 6
6509  034375  000                         .BYTE   0               ;DRIVE 7
6510
6511                              ;TRANSFER WAIT FLAG (TRNSWT=1 WORD)
6512                                      ;THIS IS A ONE WORD QUEUE. IT WILL CONTAIN THE ADDRESS OF
6513                                      ;"DPB" OF THE I/O OPERATION.
6514
6515  034376  000000              TRNSWT: .WORD   0
6516
6517                              ;SEARCH WAIT KEYS (SRCHWT=1 WORD)
6518                                      ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
6519                                      ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
6520                                      ;REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
6521                                      ;EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.
6522
6523  034400  000000              SRCHWT: .WORD   0
6524
6525                              ;RP04/5/6 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
6526                                      ;ACTDRV=0 IF DRIVER IS INACTIVE
6527                                      ;ACTDRV>0 IF DRIVER IS ACTIVE
6528
6529  034402  000                 ACTDRV: .BYTE   0
6530
6531                              ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
```

```
                                        ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
                                        ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE

034403  000                     ACTSTR: .BYTE   0

                                ;UNLOAD FLAG (ULDFLG=8 BYTES)
                                        ;ULDFLG=0 IF NO UNLOAD COMMAND
                                        ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
                                        ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE

034404  000                     ULDFLG: .BYTE   0               ;DRIVE 0
034405  000                             .BYTE   0               ;DRIVE 1
034406  000                             .BYTE   0               ;DRIVE 2
034407  000                             .BYTE   0               ;DRIVE 3
034410  000                             .BYTE   0               ;DRIVE 4
034411  000                             .BYTE   0               ;DRIVE 5
034412  000                             .BYTE   0               ;DRIVE 6
034413  000                             .BYTE   0               ;DRIVE 7

                                ;LOOK AHEAD COUNT (LACNT=8 BYTES)
                                        ;LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED

034414  000                     LACNT:  .BYTE   0               ;DRIVE 0
034415  000                             .BYTE   0               ;DRIVE 1
034416  000                             .BYTE   0               ;DRIVE 2
034417  000                             .BYTE   0               ;DRIVE 3
034420  000                             .BYTE   0               ;DRIVE 4
034421  000                             .BYTE   0               ;DRIVE 5
034422  000                             .BYTE   0               ;DRIVE 6
034423  000                             .BYTE   0               ;DRIVE 7

                                ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
                                        ;SAVEFG <0 IF SAVE THE RH11/RP04/5/6 REGISTERS WHEN THE
                                        ;OPERATION IS COMPLETED AS PER (DPB+14).
                                        ;SAVEFG=0 IF SAVE THE RH11/RP04/5/6 REGISTERS, AS PER
                                        ;(DPB+14), AFTER AN ERROR.

034424  000000                  SAVEFG. .WORD   0

                                ;SEEK FLAG (SEEKFG=1 WORD)
                                        ;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
                                        ;FOR A DATA TRANSFER START A SEARCH COMMAND
                                        ;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
                                        ;DISREGARD THE WINDOW

034426  000000                  SEEKFG: .WORD   0

                                ;TIMEOUT TABLE (TIMER=8 WORDS)
                                        ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION

034430  177777                  TIMER:  .WORD   -1       ;DRIVE 0
034432  177777                          .WORD   -1       ;DRIVE 1
034434  177777                          .WORD   -1       ;DRIVE 2
034436  177777                          .WORD   -1       ;DRIVE 3
034440  177777                          .WORD   -1       ;DRIVE 4
034442  177777                          .WORD   -1       ;DRIVE 5
```

```
  6588  034444  177777                        .WORD   -1      ;DRIVE 6
  6589  034446  177777                        .WORD   -1      ;DRIVE 7
  6590
  6591                                  ;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
  6592                                          ;DTUW<0 IF NO DATA TRANSFER UNDERWAY
  6593                                          ;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
  6594
  6595  034450  177777                DTUW:   .WORD   -1
  6596
  6597                                  ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
  6598                                          ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
  6599                                          ;ATTENTION BIT
  6600
  6601  034452     001                 ATABIT: .BYTE   1       ;DRIVE 0
  6602  034453     002                         .BYTE   2       ;DRIVE 1
  6603  034454     004                         .BYTE   4       ;DRIVE 2
  6604  034455     010                         .BYTE   10      ;DRIVE 3
  6605  034456     020                         .BYTE   20      ;DRIVE 4
  6606  034457     040                         .BYTE   40      ;DRIVE 5
  6607  034460     100                         .BYTE   100     ;DRIVE 6
  6608  034461     200                         .BYTE   200     ;DRIVE 7
  6609
  6610                                  ;RPO4/5/6 TO RH11 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
  6611                                  ;CALLING IT FATAL (MCPEMX=1 WORD)
  6612
  6613  034462  000003                MCPEMX: .WORD   3
  6614
  6615                                  ;STORAGE FOR RPADR (THE FIRST ADDRESS (776700) OF THE RH11/RPO4/5/6),
  6616                                  ;RPVEC (THE VECTOR ADDRESS (254)), AND RPVEC+2 (THE BR LEVEL (5)).
  6617
  6618  034464  176700                RPADR:  .WORD   176700
  6619  034466  000254  000240        RPVEC:  .WORD   254,5*32.
  6620
  6621                                  ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
  6622
  6623  034472  000004                MXLACT: .WORD   4
  6624                                  ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
  6625
  6626  034474  001000                MXDLTA: .WORD   8.*64.
  6627                                  ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
  6628
  6629  034476  000200                MNDLTA: .WORD   2*64.
  6630                                  ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
  6631
  6632  034500  000005                MXWNDW: .WORD   5
  6633
  6634                                  ;DEFINITIONS OF THE RH11/RPO4/5/6 ADDRESS INDEXES
  6635
  6636          000000                  RPCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
  6637          000002                  RPWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
  6638          000004                  RPBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
  6639          000006                  RPDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
  6640          000010                  RPCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
  6641          000012                  RPDS=12                 ;DRIVE STATUS REGISTER (DRIVE REG 01)
  6642          000014                  RPER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
  6643          000016                  RPAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
```

```
                    000020                  RPLA=20                         ;LOOK AHEAD REGISTER (DRIVE REG. 07)
                    000022                  RPDB=22                         ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
                    000024                  RPMR=24                         ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
                    000026                  RPDT=26                         ;DRIVE TYPE REGISTER (DRIVE REG. 06)
                    000030                  RPSN=30                         ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
                    000032                  RPOF=32                         ;OFFSET REGISTER (DRIVE REG. 11)
                    000034                  RPCA=34                         ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
                    000036                  RPCC=36                         ;CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG. 13)
                    000040                  RPER2=40                        ;ERROR REGISTER #2 (DRIVE REG. 14)
                    000042                  RPER3=42                        ;ERROR REGISTER #3 (DRIVE REG. 15)
                    000044                  RPEC1=44                        ;ECC POSITION REGISTER (DRIVE REG. 16)
                    000046                  RPEC2=46                        ;ECC PATTERN REGISTER (DRIVE REG.  17)

                                            ;RH11/RP04/5/6 DRIVER INITIALIZATION CODE
                                                    ;THIS ROUTINE WILL DETERMINE WHICH RP04/5/6 DRIVES ARE
                                                    ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
                                                    ;TO THE PROPER STATE FOR EACH DRIVE.
                                                    ;NOTE: THIS ROUTINE CALLS DRVINT
                                            ;
                                            ;CALL
                                            ;
                                            ;       JSR     PC,RPINIT
                                            ;       RETURN
                                            ;
                                            ;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
                                            ;
 034502  104412                 RPINIT: SAVREG                          ;SAVE R0 - R5
 034504  013746  177776                 MOV     @#PS,-(SP)              ;SAVE THE PRESENT PROCESSOR STATUS
 034510  012737  000240  177776         MOV     #(5*32.),@#PS           ;CHANGE THE PRIORITY TO 5
 034516  004737  042466                 JSR     PC,CLRQUE               ;CLEAR ALL REQUEST QUEUES
 034522  012701  034316                 MOV     #RPERRS,R1              ;FIRST ADDRESS TO BE CLEARED
 034526  012702  034426                 MOV     #SEEKFG,R2              ;LAST ADDRESS TO BE CLEARED
 034532  005021                 1$:     CLR     (R1)+                   ;CLEAR
 034534  020102                         CMP     R1,R2                   ;ARE WE DONE?
 034536  101775                         BLOS    1$                      ;BRANCH IF NO
 034540  012702  034450                 MOV     #DTUM,R2                ;LAST ADDRESS
 034544  012721  177777         2$:     MOV     #-1,(R1)+               ;INITIALIZE
 034550  020102                         CMP     R1,R2                   ;DONE?
 034552  101774                         BLOS    2$                      ;LOOP IF NO
 034554  005037  034336                 CLR     DRVSTA                  ;SET ALL DRIVES TO OFFLINE
 034560  005037  034340                 CLR     DRVSTA+2
 034564  005037  034342                 CLR     DRVSTA+4
 034570  005037  034344                 CLR     DRVSTA+6
 034574  013703  034466                 MOV     RPVEC,R3                ;SETUP THE RH11/RP04/5/6 VECTOR
 034600  012723  037346                 MOV     #ISR,(R3)+
 034604  013713  034470                 MOV     RPVEC+2,(R3)
 034610  013704  034464                 MOV     RPADR,R4                ;FIRST ADDRESS OF RH11/RP04
 034614  012764  000040  000010         MOV     #BIT05,RPCS2(R4)        ;MASSBUS INIT
 034622  005001                         CLR     R1                      ;START WITH DRIVE 0
 034624  004037  034714         3$:     JSR     R0,DRVINT               ;INIT THE DRIVE
 034630  000401                         BR      4$                      ;'DVA' NOT SET OR PARITY ERROR
 034632  000402                         BR      5$                      ;NORMAL RETURN
 034634  105061  034336         4$:     CLRB    DRVSTA(R1)              ;SET DRIVE STATUS TO OFFLINE
 034640  005201                 5$:     INC     R1                      ;GO TO NEXT DRIVE
 034642  042701  177770                 BIC     #^C7,R1                 ;MASK OUT UNUSED BITS
 034646  001366                         BNE     3$                      ;BR IF MORE DRIVES TO GO
```

```
6700  034650  012701  000007                  MOV     #7,R1           ;START WITH DRIVE 7
6701  034654  005037  177776                  CLR     @#PS            ;CLEAR THE PROCESSOR STATUS
6702  034660  105761  034356          6$:     TSTB    DPINT(R1)       ;WAITING FOR DRIVE TO SWITCH PORTS ?
6703  034664  001405                          BEQ     8$              ;BR NOT WAITING
6704  034666  004737  042122                  JSR     PC,SET.IE       ;SET INTERRUPT
6705  034672  105761  034356          7$:     TSTB    DPINT(R1)       ;DRIVE SWITCHED PORTS ?
6706  034676  001375                          BNE     7$              ;BR IF NOT
6707  034700  005301                  8$:     DEC     R1              ;GO TO THE NEXT DRIVE
6708  034702  100366                          BPL     6$              ;CHECK NEXT DRIVE
6709  034704  012637  177776                  MOV     (SP)+,@#PS      ;RESTORE THE PROCESSOR STATUS
6710  034710  104413                          RESREG                  ;RESTORE R0 - R5
6711  034712  000207                          RTS     PC              ;BYE-BYE
6712
6713                                  ;DRIVE INITILIZATION ROUTINE
6714                                          ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
6715                                          ;AN RP04/5/6. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
6716                                          ;IS SET TO A "1". THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
6717                                          ;INSURE THEY ARE ALL ON A "1". AND DEPENDING ON THEIR STATE,
6718                                          ;DRVSTA IS SET TO THE PROPER CONDITION.
6719                                  ;CALL
6720                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
6721                                  ;       MOV     RPADR,R4        ;UNIBUS ADDRESS OF RH11/RP04/5/6 (RPCS1)
6722                                  ;       JSR     R0,DRVINT       ;CALLED BY A JSR
6723                                  ;       RETURN1                 ;ERROR OCCURRED (PARITY)
6724                                  ;       RETURN2                 ;NORMAL RETURN
6725                                  ;
6726
6727  034714  010546                  DRVINT: MOV     R5,-(SP)        ;SAVE R5
6728  034716  105061  034336                  CLRB    DRVSTA(R1)      ;START DRIVE STATUS AS OFFLINE
6729  034722  105061  034346                  CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
6730  034726  105061  034404                  CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
6731  034732  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT A DRIVE
6732  034736  112764  000111  000000          MOVB    #111,RPCS1(R4)  ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
6733  034744  032764  010000  000010          BIT     #BIT12,RPCS2(R4) ;NONEXISTENT DRIVE?
6734  034752  001403                          BEQ     1$              ;NO---BRANCH
6735  034754  004737  042122                  JSR     PC,SET.IE       ;GO SET "IE" WITHOUT A "TRE"
6736  034760  000520                          BR      6$              ;LEAVE THIS ROUTINE
6737  034762  105061  034336          1$:     CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
6738  034766  032764  004000  000000          BIT     #BIT11,RPCS1(R4) ;SEE IF DRIVE AVAILABLE
6739  034774  001514                          BEQ     7$              ;BR IF DRIVE NOT AVAILABLE
6740  034776  004037  041432                  JSR     R0,R0.RP        ;READ THE DRIVE TYPE REG.
6741  035002  000026                          RPDT
6742  035004  035246                          8$                      ;ERROR RETURN ADDRESS
6743  035006  012605                          MOV     (SP)+,R5        ;PUT DRIVE TYPE IN R5
6744  035010  112761  000001  034346          MOVB    #1,DRVTYP(R1)   ;SET RP04 INDICATOR
6745  035016  022705  020020                  CMP     #20020,R5       ;IS IT A SINGLE PORT RP04?
6746  035022  001431                          BEQ     2$              ;BRANCH IF YES
6747  035024  022705  024020                  CMP     #24020,R5       ;IS IT A DUAL PORT RP04?
6748  035030  001426                          BEQ     2$              ;BR IF YES
6749  035032  112761  000002  034346          MOVB    #2,DRVTYP(R1)   ;SET RP05 INDICATOR
6750  035040  022705  020021                  CMP     #20021,R5       ;SINGLE PORT RP05 ?
6751  035044  001420                          BEQ     2$              ;BR IF YES
6752  035046  022705  024021                  CMP     #24021,R5       ;DUAL PORT RP05 ?
6753  035052  001415                          BEQ     2$              ;BR IF YES
6754  035054  112761  000004  034346          MOVB    #4,DRVTYP(R1)   ;SET RP06 INDICATOR
6755  035062  022705  020022                  CMP     #20022,R5       ;SINGLE PORT RP06 ?
```

```
6756  035066  001407                          BEQ     2$              ;BR IF YES
6757  035070  022705  024022                  CMP     #24022,R5       ;DUAL PORT RP06 ?
6758  035074  001404                          BEQ     2$              ;BR IF YES
6759  035076  112761  177777  034346          MOVB    #-1,DRVTYP(R1)  ;SET INDICATOR TO 'OTHER'
6760  035104  000446                          BR      6$              ;EXIT
6761  035106  012746  000121          2$:     MOV     #121,-(SP)      ;DO A "READ-IN PRESET"
6762  035112  004037  041612                  JSR     R0,WRT.RP
6763  035116  000000                          RPCS1
6764  035120  035246                          B$
6765  035122  012746  010000                  MOV     #BIT12,-(SP)    ;SET FMT22=1
6766  035126  004037  041612                  JSR     R0,WRT.RP
6767  035132  000032                          RPOF
6768  035134  035246                          B$
6769  035136  004037  041432                  JSR     R0,R0.RP        ;READ RPDS1
6770  035142  000012                          RPDS1
6771  035144  035246                          B$
6772  035146  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
6773  035150  100015                          BPL     4$              ;BRANCH IF ATA=0
6774  035152  116164  024452  000016          MOVB    ATABIT(R1),RPAS(R4) ;CLEAR ATTENTION BIT
6775  035160  004037  041432                  JSR     R0,R0.RP        ;FIND OUT WHY ATA=1
6776  035164  000014                          RPER1
6777  035166  035246                          B$
6778  035170  006126                          ROL     (SP)+           ;IS IT UNSAFE?
6779  035172  100004                          BPL     4$              ;BR IF NOT
6780  035174  112761  177777  034336          MOVB    #-1,DRVSTA(R1)  ;SET UNSAFE INDICATOR
6781  035202  000407                          BR      6$              ;EXIT
6782  035204  005005                  4$:     CLR     R5              ;CHECK MOL, DPR, DRY, AND VV
6783  035206  042705  167077                  BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
6784  035212  001003                          BNE     6$              ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
6785  035214  112761  000001  034336          MOVB    #1,DRVSTA(R1)   ;SET DRIVE STATUS TO ONLINE
6786  035222  005720                  6$:     TST     (R0)+           ;STEP OVER THE ERROR RETURN
6787  035224  000410                          BR      8$              ;EXIT
6788  035226  006301                  7$:     ASL     R1              ;CHANGE INDEX TO ADDRESS WORDS
6789  035230  012761  003720  034430          MOV     #2000.,TIMER(R1)  ;START 2 SEC TIMER
6790  035236  006201                          ASR     R1              ;RESTORE R1
6791  035240  112761  177777  034356          MOVB    #-1,DPINT(R1)   ;SET PORT INITIALIZE INDICATOR
6792  035246  012605                  8$:     MOV     (SP)+,R5        ;RESTORE R5
6793  035250  000200                          RTS     R0              ;EXIT
6794
6795                                  ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
6796                                  ;
6797                                  ;CALL
6798                                  ;
6799                                  ;       JSR     R0,@#RP04       ;CALL THE RP04/5/6 DRIVER
6800                                  ;       PNTADR                  ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
6801                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
6802                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
6803                                  ;                               ;IS AN ERROR CONDITION
6804
6805  035252  013746  177776          RP04:   MOV     @#PS,-(SP)      ;SAVE THE CALLING STATUS
6806  035256  013737  034470  177776          MOV     RPVEC+2,@#PS    ;DON'T ALLOW ANY RP04/5/6 INTERRUPTS
6807  035264  112737  000001  034402          MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
6808  035272  104412                          SAVREG                  ;SAVE R0 - R5
6809  035274  011002                          MOV     (R0),R2         ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
6810  035276  005062  000016                  CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
6811  035302  111201                          MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
```

```
6812  035304  013704  034464                  MOV     RPADR,R4        ;UNIBUS ADDRESS OF RPCS1
6813  035310  105761  034336                  TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
6814  035314  003014                          BGT     1$              ;BRANCH IF ONLINE
6815  035316  105761  034404                  TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
6816  035322  001036                          BNE     3$              ;BRANCH IF YES
6817  035324  105761  034356                  TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
6818  035330  001042                          BNE     5$              ;BR IF YES
6819  035332  004037  034714                  JSR     R0,DRVINT       ;GO INIT. THE DRIVE
6820  035336  000434                          BR      4$              ;ERROR RETURN
6821  035340  105761  034336                  TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
6822  035344  003445                          BLE     6$              ;BR IF NOT
6823  035346  105761  034366          1$:     TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
6824  035352  001031                          BNE     5$              ;BR IF YES
6825  035354  012164  000010                  MOV     R1,RPCS2(R4)    ;SELECT THE DRIVE
6826  035360  004037  042564                  JSR     R0,DRVQUE       ;PUT THIS REQUEST IN QUEUE
6827  035364  000460                          BR      9$              ;QUEUE IS FULL
6828  035366  122762  000103  000002          CMPB    #103,2(R2)      ;IS THIS REQ. FOR AN UNLOAD?
6829  035372  001003                          BNE     2$              ;BR IF NO
6830  035376  112761  177777  034404          MOVB    #-1,ULDFLG(R1)  ;SET THE "UNLOAD IN QUEUE" FLAG
6831  035404  105761  034326          2$:     TSTB    DRVACT(R1)      ;IS THIS DRIVE ACTIVE?
6832  035410  001043                          BNE     8$              ;BR IF YES
6833  035412  004737  035544                  JSR     PC,OPT          ;CALL THE OPTIMIZER
6834  035416  000440                          BR      8$
6835  035420  012762  120000  000016  3$:     MOV     #BIT15!BIT13,16(R2) ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
6836  035426  000434                          BR      8$              ;EXIT
6837  035430  004737  036654          4$:     JSR     PC,CI7          ;GO HANDLE THE PARITY ERROR
6838  035434  000431                          BR      8$
6839  035436  004037  042564          5$:     JSR     R0,DRVQUE       ;PUT REQUEST IN QUEUE
6840  035442  000431                          BR      9$              ;QUEUE IS FULL
6841  035444  032714  000100                  BIT     #BIT06,(R4)     ;IS 'IE' SET ALREADY ?
6842  035450  001023                          BNE     8$              ;BR IF IT IS
6843  035452  004737  042122                  JSR     PC,SET.IE       ;SET INTERRUPT
6844  035456  000420                          BR      8$              ;RETURN, REQUEST IN QUEUE
6845  035460  105761  034336          6$:     TSTB    DRVSTA(R1)      ;SEE IF DRIVE OFFLINE OR UNSAFE
6846  035464  002412                          BLT     7$              ;BR IF UNSAFE
6847  035466  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE ERROR INDICATOR
6848  035474  105761  034346                  TSTB    DRVTYP(R1)      ;SEE IF OFFLINE OR NONEXISTENT
6849  035500  001007                          BNE     8$              ;BR IF OFFLINE
6850  035502  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)  ;REPORT DRIVE NONEXISTENT
6851  035510  000403                          BR      8$              ;GO TO EXIT
6852  035512  012762  110000  000016  7$:     MOV     #BIT15!BIT12,16(R2)  ;DRIVE IS UNSAFE
6853  035520  104413                  8$:     RESREG                  ;RESTORE R0 - R5
6854  035522  005720                          TST     (R0)+           ;SETUP FOR NORMAL RETURN
6855  035524  000401                          BR      10$             ;FINISH UP, THEN EXIT
6856  035526  104413                  9$:     RESREG                  ;RESTORE R0 - R5
6857  035530  005720                  10$:    TST     (R0)+           ;CORRECT THE RETURN ADDRESS
6858  035532  105037  034402                  CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
6859  035536  012637  177776                  MOV     (SP)+,@#PS      ;RETURN "PS" TO USER LEVEL
6860  035542  000200                          RTS     R0              ;RETURN TO CALLER
6861
6862                                  ;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
6863                                  ;
6864                                  ;CALL
6865                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
6866                                  ;       JSR     PC,OPT          ;SETUP A COMMAND
6867                                  ;
```

```
6868  035544  104412                    OPT:    SAVREG                  ;SAVE R0 - R5
6869  035546  013746  177776                    MOV     @#PS,-(SP)      ;SAVE PROC. STATUS
6870  035552  146137  034452  034400            BICB    ATABIT(R1),SRCHWT       ;CLEAR "SEARCH WAIT" KEY
6871  035560  004737  042640                    JSR     PC,GETREQ       ;GET "DPB" POINTER OF REQUEST
6872  035564  005702                            TST     R2              ;IS THERE A REQUEST IN QUEUE?
6873  035566  001505                            BEQ     7$              ;NO--BRANCH TO EXIT
6874  035570  032764  010000  000012            BIT     #BIT12,RPDS1(R4)  ;IS MOL STILL SET ?
6875  035576  001404                            BEQ     9$              ;BR IF NOT
6876  035600  032764  000100  000012            BIT     #BIT6,RPDS1(R4)  ;IS VV SET ?
6877  035606  001003                            BNE     10$             ;BR IF IT IS
6878  035610  004037  034714            9$:     JSR     R0,DRVINT       ;SEE IF DRIVE STILL ONLINE ?
6879  035614  000470                            BR      6$              ;PARITY OR 'DVA' NOT SET
6880  035616  105761  034336            10$:    TSTB    DRVSTA(R1)      ;IS DRIVE ONLINE?
6881  035622  003014                            BGT     1$              ;YES--BRANCH
6882  035624  004737  042662                    JSR     PC,POPQUE       ;NO--REMOVE REQUEST FROM QUEUE
6883  035630  012762  140000  000016            MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE STATUS/ERROR INDICATOR
6884  035636  105761  034336                    TSTB    DRVSTA(R1)      ;IS DRIVE UNSAFE ?
6885  035642  100064                            BPL     8$              ;BR TO EXIT IF NOT
6886  035644  012762  110000  000016            MOV     #BIT15!BIT12,16(R2)  ;SET UNSAFE STATUS/ERROR INDICATOR
6887  035652  000460                            BR      8$              ;BRANCH TO EXIT
6888  035654  012746  000111            1$:     MOV     #111,-(SP)      ;LOAD COMMAND ONTO THE STACK
6889  035660  004037  041612                    JSR     R0,WRT.RP       ;LOAD THE REGISTER
6890  035664  000000                            RPCS1                   ;REGISTER INCREMENT
6891  035666  035776                            6$                      ;ERROR RETURN ADDRESS
6892  035670  032714  004000                    BIT     #BIT11,(R4)     ;DRIVE AVAILABLE ?
6893  035674  001427                            BEQ     5$              ;BR IF NOT
6894  035676  122762  000150  000002            CMPB    #150,2(R2)      ;IS THE REQUEST FOR I/O?
6895  035704  002403                            BLT     2$              ;YES--BRANCH
6896  035706  004737  036240                    JSR     PC,CI4          ;CALL THE COMMAND INITIATOR
6897  035712  000440                            BR      8$              ;BRANCH TO EXIT
6898  035714  005737  034450            2$:     TST     DTUW            ;DATA TRANSFER UNDERWAY?
6899  035720  002012                            BGE     4$              ;YES--GO START A SEARCH
6900  035722  005737  034426                    TST     SEEKFG          ;DO IMPLIED SEEKS?
6901  035726  100404                            BMI     3$              ;YES---BRANCH
6902  035730  004037  037210                    JSR     R0,LA           ;NO--DO LOOK AHEAD
6903  035734  000427                            BR      8$              ;RETURN HERE ON A PARITY ERROR
6904  035736  000403                            BR      4$              ;GO START A SEARCH
6905  035740  004737  036024            3$:     JSR     PC,CI1          ;START A DATA TRANSFER
6906  035744  000423                            BR      8$
6907  035746  004737  036132            4$:     JSR     PC,CI3          ;START A SEARCH
6908  035752  000420                            BR      8$              ;GO TO THE EXIT
6909  035754  112761  177777  034366    5$:     MOVB    #-1,DPRQS(R1)   ;SET PORT REQUEST INDICATOR
6910  035762  010103                            MOV     R1,R3           ;SET UP TO ADDRESS WORDS
6911  035764  006303                            ASL     R3              ;CONVERT TO WORD INDEX
6912  035766  012763  023420  034430            MOV     #10000.,TIMER(R3)  ;START 10 SEC TIMER
6913  035774  000402                            BR      7$              ;EXIT
6914  035776  004737  036654            6$:     JSR     PC,CI7          ;PROCESS THE PARITY ERROR
6915  036002  032714  000100            7$:     BIT     #BIT06,(R4)     ;SEE IF 'IE' ALREADY SET
6916  036006  001002                            BNE     8$              ;BR IF SET
6917  036010  004737  042122                    JSR     PC,SET.IE       ;SET "IE" WITHOUT A "TRE"
6918  036014  012637  177776            8$:     MOV     (SP)+,@#PS      ;RESTORE PROC. STATUS
6919  036020  104413                            RESREG                  ;RESTORE R0 - R5
6920  036022  000207                            RTS     PC
6921
6922                                    ;COMMAND INITIATOR
6923                                    ;
```

```
6924                                    ;CALL
6925                                    ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
6926                                    ;       MOV     #DPB,R2         ;ADDRESS OF DPB
6927                                    ;       JSR     PC,CI?          ;CI?= CI1,CI3, OR CI4
6928                                    ;                               ;WHERE:
6929                                    ;                               ;CI1=DATA TRANSFER
6930                                    ;                               ;CI2=SEARCH REQUESTED BY DATA XFER
6931                                    ;                               ;CI3=NOT DATA TRANSFER
6932
6933  036024  004737  042662      CI1:  JSR     PC,POPQUE       ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
6934  036030  010237  034376            MOV     R2,TRNSWT       ;PUT REQ. IN TRANSFER WAIT QUEUE
6935  036034  010203                    MOV     R2,R3           ;DPB ADDRESS TO R3
6936  036036  013704  034464            MOV     RPADR,R4        ;RPCS1 ADDRESS
6937  036042  010164  000010            MOV     R1,RPCS2(R4)    ;SELECT DRIVE
6938  036046  062703  000004            ADD     #4,R3           ;DESIRED WORD COUNT
6939  036052  062704  000002            ADD     #2,R4           ;RPWC ADDRESS
6940  036056  012324                    MOV     (R3)+,(R4)+     ;LOAD WORD COUNT
6941  036060  012324                    MOV     (R3)+,(R4)+     ;LOAD BUFFER ADDRESS
6942  036062  012346                    MOV     (R3)+,-(SP)     ;LOAD SECTOR AND TRACK
6943  036064  004037  041612            JSR     R0,WRT.RP       ;CALL THE LOAD(WRITE) ROUTINE
6944  036070  000006                    RPDA                    ;INDEX OF REGISTER TO LOAD
6945  036072  036654                    CI7                     ;ERROR RETURN ADDRESS
6946  036074  012346                    MOV     (R3)+,-(SP)     ;LOAD CYLINDER ADDRESS
6947  036076  004037  041612            JSR     R0,WRT.RP
6948  036102  000034                    RPCA
6949  036104  036654                    CI7
6950  036106  016246  000002            MOV     2(R2),-(SP)     ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
6951  036112  004037  041612            JSR     R0,WRT.RP
6952  036116  000000                    RPCS1
6953  036120  036654                    CI7
6954  036122  010137  034450            MOV     R1,DTUW         ;SET "DATA TRANSFER UNDERWAY"
6955  036126  000137  036616            JMP     CI5
6956  036132  013704  034464      CI3:  MOV     RPADR,R4        ;RPCS1 ADDRESS
6957  036136  010164  000010            MOV     R1,RPCS2(R4)    ;SELECT DRIVE
6958  036142  016246  000014            MOV     12(R2),-(SP)    ;DESIRED CYLINDER ADDRESS
6959  036146  004037  041612            JSR     R0,WRT.RP
6960  036152  000034                    RPCA
6961  036154  036654                    CI7
6962  036156  116203  000010            MOVB    10(R2),R3       ;PICKUP SECTOR ADDRESS
6963  036162  163703  034500            SUB     MYWNDW,R3       ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
6964  036166  002002                    BGE     1$
6965  036170  062703  000026            ADD     #22.,R3
6966  036174  010346              1$:   MOV     R3,-(SP)        ;COMBINE THE ADJUSTED SECTOR WITH
6967  036176  116266  000011  000001    MOVB    11(R2),1(SP)    ;THE DESIRED TRACK
6968  036204  004037  041612            JSR     R0,WRT.RP       ;LOAD DESIRED TRACK & SECTOR
6969  036210  000006                    RPDA
6970  036212  036654                    CI7
6971  036214  012746  000131            MOV     #131,-(SP)      ;START A SEARCH
6972  036220  004037  041612            JSR     R0,WRT.RP
6973  036224  000000                    RPCS1
6974  036226  036654                    CI7
6975  036230  156137  034452  034400    BISB    ATABIT(R1),SRCHWT       ;SET "SEARCH WAIT" KEY
6976  036236  000567                    BR      CI5
6977  036240  013704  034464      CI4:  MOV     RPADR,R4        ;RPCS1 ADDRESS
6978  036244  010154  000010            MOV     R1,RPCS2(R4)    ;SELECT DRIVE
6979  036250  116203  000002            MOVB    2(R2),R3        ;PICKUP THE REQUESTED COMMAND
```

```
6980  036254  122703  000131              CMPB    #131,R3         ;IS IT A SEARCH COMMAND?
6981  036260  001007                      BNE     1$              ;BRANCH IF NO
6982  036262  016246  000010              MOV     10(R2),-(SP)    ;LOAD DESIRED TRACK & SECTOR
6983  036266  004037  041612              JSR     R0,WRT.RP
6984  036272  000006                      RPDA
6985  036274  036654                      CI7
6986  036276  000403                      BR      2$              ;GO LOAD CYLINDER
6987  036300  122703  000105      1$:     CMPB    #105,R3         ;IS IT A SEEK COMMAND
6988  036304  001007                      BNE     3$              ;BRANCH IF NO
6989  036306  016246  000012      2$:     MOV     12(R2),-(SP)    ;LOAD DESIRED CYLINDER
6990  036312  004037  041612              JSR     R0,WRT.RP
6991  036316  000034                      RPCA
6992  036320  036654                      CI7
6993  036322  000516                      BR      CI6
6994  036324  122703  000115      3$:     CMPB    #115,R3         ;IS IT AN "OFFSET" COMMAND?
6995  036330  001013                      BNE     4$              ;BR IF NO
6996  036332  004037  041432              JSR     R0,R0.RP        ;MERGE THE OFFSET VALUE INTO RPOF
6997  036336  000032                      RPOF                    ;BUT DON'T CHANGE THE UPPER
6998  036340  036654                      CI7
6999  036342  116216  000001              MOVB    1(R2),(SP)      ;BYTE WHEN LOADING THE
7000  036346  004037  041612              JSR     R0,WRT.RP       ;REGISTER (RPOF)
7001  036352  000032                      RPOF
7002  036354  036654                      CI7
7003  036356  000530                      BR      CI6             ;GO START THE COMMAND
7004  036360  122703  000107      4$:     CMPB    #107,R3         ;IS IT A "RECALIBRATE" COMMAND?
7005  036364  001525                      BEQ     CI6             ;BRANCH IF YES
7006  036366  122703  000117              CMPB    #117,R3         ;IS IT A RETURN TO CENTER?
7007  036372  001522                      BEQ     CI6             ;BRANCH IF YES
7008  036374  122703  000103              CMPB    #103,R3         ;IS IT AN "UNLOAD" COMMAND?
7009  036400  001016                      BNE     5$              ;BRANCH IF NO
7010  036402  112761  000001  034326      MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE INDICATOR
7011  036410  105061  034336              CLRB    DRVSTA(R1)      ;PUT DRIVE STATUS TO OFFLINE
7012  036414  112761  000001  034404      MOVB    #1,ULDFLG(R1)   ;SET "UNLOAD IN PROGRESS" FLAG
7013  036422  010346                      MOV     R3,-(SP)        ;START THE "UNLOAD" COMMAND
7014  036424  004037  041612              JSR     R0,WRT.RP
7015  036430  000000                      RPCS1
7016  036432  036654                      CI7
7017  036434  000207                      RTS     PC              ;RETURN TO USER
7018  036436  122703  000143      5$:     CMPB    #143,R3         ;IS IT A "SET FORMAT" COMMAND?
7019  036442  001014                      BNE     6$              ;BRANCH IF NO
7020  036444  004037  041432              JSR     R0,R0.RP        ;READ THE OFFSET REGISTER
7021  036450  000032                      RPOF
7022  036452  036654                      CI7
7023  036454  116266  000001  000001      MOVB    1(R2),1(SP)     ;COMBINE "FMT22","ECI", AND "HCI"
7024  036462  004037  041612              JSR     R0,WRT.RP       ;LOAD "FMT22", "ECI", AND/OR "HCI".
7025  036466  000032                      RPOF
7026  036470  036654                      CI7
7027  036472  000436                      BR      12$
7028  036474  122703  000141      6$:     CMPB    #141,R3         ;IS IT A "GET REGISTER" COMMAND?
7029  036500  001023                      BNE     10$             ;BRANCH IF NO
7030  036502  016203  000006      7$:     MOV     6(R2),R3        ;POINTS TO 1ST ADDRESS OF WHERE
7031                                                              ;TO PUT THE REGISTER(S)
7032  036506  116237  000010  036524      MOVB    10(R2),9$       ;INIT. THE INDEX FOR THE FIRST REG.
7033  036514  116205  000011              MOVB    11(R2),R5       ;INDEX OF LAST REG. TO MOVE
7034  036520  004037  041432      8$:     JSR     R0,R0.RP        ;READ RP04/5/6 REGISTER
7035  036524  000000              9$:     RPCS1                   ;INDEX OF REG. TO READ
```

```
7036  036526  036654                                  CI7
7037  036530  012623                                  MOV     (SP)+,(R3)+     ;GET THE CONTENTS OF RH11/RP04/5/6 REG.
7038  036532  023705  036524                          CMP     %5,R5           ;LAST REG. BEEN READ?
7039  036536  001414                                  BEQ     12$             ;GET OUT IF YES
7040  036540  062737  000002  036524                  ADD     #2,%5           ;INCREASE THE INDEX BY 2
7041  036546  000764                                  BR      8$              ;LOOP--MORE TO READ
7042  036550  122703  000145          10$:    CMPB    #145,R3         ;IS IT A "SELECT DRIVE" COMMAND?
7043  036554  001405                                  BEQ     12$             ;BRANCH IF YES
7044  036556  010346          11$:    MOV     R3,-(SP)        ;LOAD THE COMMAND
7045  036560  004037  041612                          JSR     R0,WRT.RP
7046  036564  000000                                  RPCS1
7047  036566  036654                                  CI7
7048  036570  004737  042662          12$:    JSR     PC,POPQUE       ;REMOVE REQ. FROM QUEUE
7049  036574  052762  000200  000016                  BIS     #BIT07,16(R2)   ;SET THE "DONE" BIT
7050  036602  005737  034424                          TST     SAVEFG          ;SAVE THE RH11/RP04/5/6 REGISTERS?
7051  036606  100002                                  BPL     13$             ;BRANCH IF NO
7052  036610  004737  042004                          JSR     PC,SVRH11       ;YES--GO SAVE THE REGISTERS
7053  036614  000207          13$:    RTS     PC              ;RETURN TO USER
7054  036616  006301          CI5:    ASL     R1
7055  036620  012761  001750  034430                  MOV     #1000.,TIMER(R1)        ;SET A ONE SECOND TIMER
7056  036626  006201                                  ASR     R1
7057  036630  112761  000001  034326                  MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE
7058  036636  000207                                  RTS     PC              ;RETURN TO THE USER
7059  036640  010346          CI6:    MOV     R3,-(SP)        ;LOAD THE COMMAND
7060  036642  004037  041612                          JSR     R0,WRT.RP
7061  036646  000000                                  RPCS1
7062  036650  036654                                  CI7
7063  036652  000761                                  BR      CI5
7064  036654  032764  010000  000010  CI7:    BIT     #BIT12,RPCS2(R4)  ;DRIVE NON-EXISTENT ?
7065  036662  001034                                  BNE     CI8             ;BR IF YES
7066  036664  005702          1$:     TST     R2              ;ANYTHING IN QUEUE ?
7067  036666  001405                                  BEQ     CI7B            ;BF IF NOT
7068  036670  012762  104000  000016                  MOV     #BIT15!BIT11,16(R2)     ;SET "PARITY" ERROR INDICATOR
7069  036676  004737  042004                          JSR     PC,SVRH11       ;GO SAVE THE RH11/RP04/5/6 REGISTERS
7070  036702  012746  000111          CI7B:   MOV     #111,-(SP)      ;DO A "DRIVE CLEAR"
7071  036706  004037  041612                          JSR     R0,WRT.RP
7072  036712  000000                                  RPCS1
7073  036714  036754                                  CI8
7074  036716  004737  042544                          JSR     PC,EMPTYQ       ;EMPTY THE QUEUE
7075  036722  105061  034404                          CLRB    UNLDFLG(R1)     ;CLEAR THE UNLOAD IN QUEUE FLAG
7076  036726  105061  034326                          CLRB    DRVACT(R1)      ;DRIVE IS IDLE
7077  036732  020137  034450                          CMP     R1,DTUN         ;IF THIS DRIVE HAD AN I/O REQUEST
7078  036736  001005                                  BNE     1$              ;IN PROGRESS CLEAR ALL OF THE FLAGS
7079  036740  005037  034376                          CLR     TRNSWT
7080  036744  012737  177777  034450                  MOV     #-1,DTUN
7081  036752  000207          1$:     RTS     PC
7082  036754  104412          CI8:    SAVREG                  ;SAVE R0 - R5
7083  036756  032764  010000  000010                  BIT     #BIT12,RPCS2(R4)  ;IS 'NED' SET ?
7084  036764  001002                                  BNE     1$              ;BR IF YES
7085  036766  005001                                  CLR     R1
7086  036770  005203                                  CLR     R3
7087  036772  105761  034326          1$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE?
7088  036776  001443                                  BEQ     5$              ;BRANCH IF NO
7089  037000  013702  034376                          MOV     TRNSWT,R2       ;GET THE "TRANSFER WAIT" QUEUE
7090  037004  020137  034450                          CMP     R1,DTUN         ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
7091  037010  001402                                  BEQ     2$              ;BRANCH IF YES
```

```
7092  037012  004737  042640                  JSR     PC,GETREQ       ;GET THE DPB POINTER
7093  037016  005702                  2$:     TST     R2              ;QUEUE ENTRY FOR DRIVE ?
7094  037020  001415                          BEQ     4$              ;BR IF NOT
7095  037022  032764  010000  000010          BIT     #BIT12,RPCS2(R4)  ;'NED' SET ?
7096  037030  001404                          BEQ     3$              ;BR IF NOT
7097  037032  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)  ;SET 'DRIVE NON-EXISTENT' INDICATOR
7098  037040  000405                          BR      4$              ;CONTINUE
7099  037042  012762  102000  000016  3$:     MOV     #BIT15!BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
7100  037050  004737  042004                  JSR     PC,SVRH11       ;SAVE RH11/RP04/5/6 REGISTERS
7101  037054  012763  177777  034430  4$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
7102  037062  105061  034326                  CLRB    DRVACT(R1)      ;SET "DRIVE ACTIVE" TO IDLE
7103  037066  020137  034450                  CMP     R1,DTUN         ;IS THIS DRIVE SETUP FOR A TRANSFER
7104  037072  001005                          BNE     5$              ;BR IF NOT
7105  037074  012737  177777  034450          MOV     #-1,DTUN        ;RESET THE INDICATOR
7106  037102  005037  034376                  CLR     TRNSWT          ;CLEAR THE TRANSFER QUEUE
7107  037106  105061  034404          5$:     CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
7108  037112  032764  010000  000010          BIT     #BIT12,RPCS2(R4)  ;'NED' SET ?
7109  037120  001021                          BNE     6$              ;BR IF YES
7110  037122  005201                          INC     R1              ;MOVE TO THE NEXT DRIVE
7111  037124  062703  000002                  ADD     #2,R3
7112  037130  042701  177770                  BIC     #^C7,R1
7113  037134  001316                          BNE     1$              ;BRANCH IF MORE DRIVES
7114  037136  012737  177777  034450          MOV     #-1,DTUN        ;NO DATA TRANSFERS UNDERWAY
7115  037144  005037  034376                  CLR     TRNSWT          ;CLEAR THE 'TRANSFER WAIT' QUEUE
7116  037150  004737  042466                  JSR     PC,CLRQUE       ;CLEAR ALL OF THE REQUEST QUEUES
7117  037154  012764  000040  000010          MOV     #BIT05,RPCS2(R4)  ;DO A MASSBUS INIT.
7118  037162  000406                          BR      7$              ;CONTINUE
7119  037164  004737  042544          6$:     JSR     PC,EMPTYQ       ;CLEAR THE DRIVE'S QUEUE
7120  037170  105061  034336                  CLRB    DRVSTA(R1)      ;SET DRIVE TO OFFLINE
7121  037174  105061  034346                  CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
7122  037200  004737  042122          7$:     JSR     PC,SET.IE       ;SET "IE" WITHOUT "TRE"
7123  037204  104413                          RESREG                  ;RESTORE R0 - R5
7124  037206  000207                          RTS     PC              ;RETURN
7125
7126                                  ;LOOK AHEAD ROUTINE
7127                                  ;
7128                                  ;CALL
7129                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
7130                                  ;       MOV     #DPB,R2         ;POINT TO DPB
7131                                  ;       JSR     R0,LA           ;GO CHECK THE WINDOW
7132                                  ;       RETURN1                 ;ERROR RETURN
7133                                  ;       RETURN2                 ;START A SEARCH
7134                                  ;       RETURN3                 ;START A DATA TRANSFER
7135
7136  037210  013704  034464          LA:     MOV     RPADR,R4        ;GET RPCS1'S ADDRESS
7137  037214  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
7138  037220  004037  041432                  JSR     R0,R0.RP        ;READ CURRENT CYLINDER
7139  037224  000036                          RPCC
7140  037226  037340                          4$                      ;ERROR RETURN ADDRESS
7141  037230  022662  000012                  CMP     (SP)+,12(R2)    ;IS CURRENT CYLINDER=DESIRED
7142                                                                  ;CYLINDER?
7143  037234  001037                          BNE     3$              ;EXIT IF NO
7144  037236  105261  034414                  INCB    LACNT(R1)       ;INCREMENT THE LOOK AHEAD COUNT
7145  037242  126137  034414  034472          CMPB    LACNT(R1),MXLACT ;EXCEED MAX?
7146  037250  003026                          BGT     2$              ;BRANCH IF YES
7147  037252  116203  000010                  MOVB    10(R2),R3       ;GET DESIRED SECTOR ADDRESS AND
```

```
7148  037256  000303                          SWAB    R3              ;MULT. BY 64--ALIGN WITH
7149  037260  006203                          ASR     R3              ;LOOK AHEAD REGISTER
7150  037262  006203                          ASR     R3
7151  037264  012737  000340  177776          MOV     #340,@#PS       ;PRIORITY LEVEL "7"
7152  037272  004037  041432                  JSR     R0,R0.RP        ;READ LOOK AHEAD REGISTER
7153  037276  000020                          RPLA
7154  037300  037340                          4$
7155  037302  162603                          SUB     (SP)+,R3        ;CALCULATE THE DELTA
7156  037304  002002                          BGE     1$
7157  037306  062703  002600                  ADD     #(22.*64.),R3   ;MAKE THE DELTA POSITIVE
7158  037312  023703  034474          1$:     CMP     MXDLTA,R3       ;CHECK THE DELTA TO SEE
7159  037316  002406                          BLT     3$              ;IF IT IS WITHIN THE
7160  037320  023703  034476                  CMP     MNDLTA,R3       ;WINDOW---IF YES, ZERO
7161  037324  002003                          BGE     3$              ;THE LOOK AHEAD COUNT
7162  037326  105061  034414          2$:     CLRB    LACNT(R1)       ;AND TAKE THE I/O EXIT
7163  037332  005720                          TST     (R0)+
7164  037334  005720                  3$:     TST     (R0)+           ;ADJUST THE RETURN ADDRESS
7165  037336  000402                          BR      5$              ;EXIT
7166  037340  004737  036654          4$:     JSR     PC,CI7          ;PROCESS THE ERROR
7167  037344  000200                  5$:     RTS     R0              ;RETURN
7168
7169                                          ;INTERRUPT SERVICE ROUTINE
7170
7171  037346  112737  000001  034402  ISR:    MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
7172  037354  104412                          SAVREG                  ;SAVE R0 - R5
7173  037356  013704  034464                  MOV     RPADR,R4        ;ADDRESS OF RHSCS1
7174  037362  013701  034450                  MOV     DTUW,R1         ;GET "DATA TRANSFER UNDERWAY" INDICATOR
7175  037366  002403                          BLT     1$              ;BRANCH IF NO DATA TRANSFER UNDERWAY
7176  037370  004737  037412                  JSR     PC,TD           ;CALL TRANSFER DONE
7177  037374  000402                          BR      2$              ;EXIT
7178  037376  004737  037552          1$:     JSR     PC,SC           ;CALL SPECIAL CONDITIONS
7179  037402  104413                  2$:     RESREG                  ;RESTORE R0 - R5
7180  037404  105037  034402                  CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
7181  037410  000002                          RTI                     ;RETURN
7182
7183                                          ;TRANSFER DONE ROUTINE
7184
7185  037412  105061  034326          TD:     CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE INDICATOR TO IDLE
7186  037416  012737  177777  034450          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
7187  037424  006301                          ASL     R1
7188  037426  012761  177777  034430          MOV     #-1,TIMER(R1)   ;CANCEL TIMEOUT
7189  037434  006201                          ASR     R1
7190  037436  013702  034376                  MOV     TRNSWT,R2       ;GET "DPB" ADDRESS FROM THE
7191  037442  005037  034376                  CLR     TRNSWT          ;TRANSFER WAIT QUEUE--CLEAR QUEUE
7192  037446  052762  000200  000016          BIS     #BIT07,16(R2)   ;SET DONE
7193  037454  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT THE DRIVE
7194  037460  004037  041432                  JSR     R0,R0.RP        ;TRANSFER ERROR(TRE=1)?
7195  037464  000000                          RPCS1
7196  037466  036654                          CI7
7197  037470  006126                          ROL     (SP)+
7198  037472  100413                          BMI     3$              ;BR IF YES
7199  037474  005737  034424                  TST     SAVEFG          ;SAVE THE RH11/RP04/5/6 REGISTERS?
7200  037500  100002                          BPL     1$              ;BRANCH IF NO
7201  037502  004737  042004                  JSR     PC,SVRH11       ;YES--SAVE THE REGISTERS
7202  037506  004737  035544          1$:     JSR     PC,OPT          ;CALL OPTIMIZER
7203  037512  000417                          BR      SC              ;CHECK OTHER DRIVES
```

```
7204  037514  012714  000113          2$:     MOV     #113,(R4)       ;RELEASE THE DRIVE
7205  037520  000414                  BR      SC                      ;CHECK FOR OTHER DRIVES
7206  037522  052762  100100  000016  3$:     BIS     #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
7207  037530  004737  042544                  JSR     PC,EMPTYQ       ;EMPTY THE "DRIVE'S WAIT" QUEUE
7208  037534  004737  042004                  JSR     PC,SVRH11       ;SAVE THE RH11/RP04/5/6 REGISTERS
7209  037540  012714  040111                  MOV     #40111,(R4)     ;ISSUE A "DRIVE CLEAR"
7210  037544  012714  000113                  MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
7211  037550  000400                  BR      SC                      ;CHECK FOR OTHER DRIVES
7212
7213                                  ;SPECIAL CONDITION ROUTINE
7214
7215  037552  116403  000016          SC:     MOVB    RPAS(R4),R3     ;READ "RPAS"
7216  037556  001014                  BNE     2$                      ;BRANCH IF ANY 'ATA' BITS SET
7217  037560  004037  041432                  JSR     R0,RD.RP        ;READ CONTROL AND STATUS REGISTER
7218  037564  000000                  RPCS1
7219  037566  036754                  CIB
7220  037570  106126                  ROLB    (SP)+                   ;IS "IE"=1?
7221  037572  100405                  BMI     1$                      ;YES  NO DRIVES TO CHECK
7222  037574  004037  042724                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN 'SESCAPE'
7223  037600  104001                  ERROR   1                       ;REPORT AN ILLEGAL INTERRUPT
7224  037602  004737  042122                  JSR     PC,SET.IE       ;SET INTERRUPT ENABLE
7225  037606  000207          1$:     RTS     PC                      ;RETURN
7226  037610  005046          2$:     CLR     -(SP)                   ;PROCESS ALL DRIVES THAT HAVE
7227  037612  110316                  MOVB    R3,(SP)                 ;AN "ATA"=1
7228  037614  012703  000001                  MOV     #1,R3
7229  037620  005001                  CLR     R1
7230  037622  030316          SC3:    BIT     R3,(SP)                 ;ATA=1?
7231  037624  001005                  BNE     SC5                     ;YES--BRANCH
7232  037626  005201          SC4:    INC     R1                      ;MOVE TO THE NEXT DRIVE
7233  037630  106303                  ASLB    R3
7234  037632  001373                  BNE     SC3                     ;BRANCH IF MORE TO CHECK?
7235  037634  005726                  TST     (SP)+                   ;CLEAN OFF THE STACK
7236  037636  000207                  RTS     PC                      ;RETURN TO USER
7237  037640  105761  034356          SC5:    TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
7238  037644  001402                  BEQ     1$                      ;BR IF NOT
7239  037646  000137  040544                  JMP     SC13            ;PROCESS THE DRIVE
7240  037652  105761  034366          1$:     TSTB    DPRQS(R1)       ;PORT REQUEST OUTSTANDING ?
7241  037656  001402                  BEQ     2$                      ;BR IF NOT
7242  037660  000137  040544                  JMP     SC13            ;START THE OUTSTANDING COMMAND
7243  037664  105761  034336          2$:     TSTB    DRVSTA(R1)      ;CHECK THE DRIVE STATUS
7244  037670  003025                  BGT     5$                      ;BRANCH IF ONLINE
7245  037672  105761  034404                  TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS?
7246  037676  003422                  BLE     5$                      ;BRANCH IF NOT
7247  037700  004737  042640                  JSR     PC,GETREQ       ;GET DPB POINTER
7248  037704  004737  042004                  JSR     PC,SVRH11       ;SAVE THE RH11/RP04/5/6 REGISTERS
7249  037710  004737  040474                  JSR     PC,SC12         ;SAVE RPOS1, RPER1, RPER2, AND RPER3
7250                                          ;ALSO DO A DRIVE INIT (DRVINT)
7251  037714  105761  034336                  TSTB    DRVSTA(R1)      ;DID DRIVE COME ONLINE?
7252  037720  003416                  BLE     6$                      ;NO---BRANCH
7253  037722  032737  040000  034316          BIT     #BIT14,RPERRS   ;WAS THERE AN ERROR?
7254  037730  001002                  BNE     3$                      ;BR IF ERROR
7255  037732  000137  040404                  JMP     SC11            ;NO ERROR
7256  037736  013705  034320          3$:     MOV     RPERRS+2,R5     ;YES -- PICKUP RPER1 AND
7257  037742  000502                  BR      SC6A                    ;GO PROCESS THE ERROR
7258  037744  105761  034326          5$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
7259  037750  001033                  BNE     SC6                     ;BR IF EITHER
```

```
7260  037752  004737  040474                  JSR     PC,SC12         ;SAVE RPDS1, RPER1, RPER2, AND RPER3
7261                                                                    ;ALSO DO A DRVINT
7262  037756  105761  034356          6$:     TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE ?
7263  037762  001321                          BNE     SC4             ;BR IF YES, CHECK ON MORE DRIVES
7264  037764  105761  034336                  TSTB    DRVSTA(R1)      ;CHECK ON DRIVE'S STATUS
7265  037770  100412                          BMI     7$              ;BR IF UNSAFE
7266  037772  032737  020000  034324          BIT     #BIT13,RPERRS+6 ;ADDRESS PLUG CHANGED ?
7267  040000  001013                          BNE     8$              ;BR IF YES
7268  040002  012746  000113                  MOV     #113,-(SP)      ;RELEASE COMMAND
7269  040006  004037  041612                  JSR     R0,WRT.RP       ;WRITE THE COMMAND INTO RPCS1
7270  040012  000000                          RPCS1                   ;REGISTER INDEX
7271  040014  040354                          SC8                     ;PARITY EXIT ADDRESS
7272  040016  011605                  7$:     MOV     (SP),R5         ;PICKUP (RPAS) BEFORE THE ERROR CALL
7273  040020  004037  042724                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN 'ESCAPE'
7274  040024  104002                          ERROR   2               ;REPORT THE UNEXPECTED ATTENTION
7275  040026  000677                          BR      SC4             ;GO CHECK FOR MORE ATA'S
7276  040030                          8$:
7277  040030  004037  042724                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN 'ESCAPE'
7278  040034  104005                          ERROR   5               ;REPORT THE ADDRESS PLUG CHANGE
7279  040036  000673                          BR      SC4             ;CHECK FOR MORE DRIVES
7280  040040  006301                  SC6:    ASL     R1              ;SETUP TO ADDRESS WORDS
7281  040042  012761  177777  034430          MOV     #-1,TIMER(R1)   ;STOP THE TIMER
7282  040050  006201                          ASR     R1              ;RESTORE THE DRIVE ADDRESS
7283  040052  004737  042640                  JSR     PC,GETREQ       ;GET THE DPB POINTER FROM THE QUEUE
7284  040056  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
7285  040062  004037  041432                  JSR     R0,R0.RP        ;READ THE RP04'S STATUS REG.
7286  040066  000012                          RPDS1
7287  040070  040354                          SC8
7288  040072  011605                          MOV     (SP),R5         ;AND PUT IT IN R5
7289  040074  006126                          ROL     (SP)+           ;WAS THERE AN ERROR?
7290  040076  100407                          BMI     1$              ;BR IF ERROR
7291  040100  105761  034326                  TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
7292  040104  003137                          BGT     SC11            ;BR IF DRIVE ACTIVE WITH ORDER
7293  040106  052762  100210  000016          BIS     #BIT15!BIT07!BIT03,16(R2)  ;INFORM USER OF ERROR RECOVER COMPLETION
7294  040114  000470                          BR      SC7
7295  040116  004037  041432          1$:     JSR     R0,R0.RP        ;READ ERROR REGISTER #1
7296  040122  000014                          RPER1
7297  040124  040354                          SC8
7298  040126  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
7299  040130  004737  042004                  JSR     PC,SVRH11       ;SAVE RH11/RP04/5/6 REGISTERS
7300  040134  012746  000111                  MOV     #111,-(SP)      ;ISSUE A DRIVE CLEAR
7301  040140  004037  041612                  JSR     R0,WRT.RP
7302  040144  000000                          RPCS1
7303  040146  040354                          SC8
7304  040150  006105                  SC6A:   ROL     R5              ;WAS "UNSAFE" CONDITION =1?
7305  040152  100406                          BMI     1$              ;BRANCH IF YES
7306  040154  005702                          TST     R2              ;ANYTHING IN QUEUE ?
7307  040156  001447                          BEQ     SC7             ;BR IF NOT
7308  040160  052762  100240  000016          BIS     #BIT15!BIT07!BIT05,16(R2)       ;INFORM USER OF ERROR
7309  040166  000443                          BR      SC7
7310  040170  004037  041432          1$:     JSR     R0,R0.RP        ;READ DRIVE STATUS REG. #1
7311  040174  000012                          RPDS1
7312  040176  040354                          SC8
7313  040200  011605                          MOV     (SP),R5         ;SAVE RPDS1 IN R5
7314  040202  006126                          ROL     (SP)+           ;"ERR"=1?
7315  040204  100011                          BPL     2$              ;BR IF NO--UNSAFE CLEARED
```

```
7316  040206  112761  177777  034336          MOVB    #-1,DRVSTA(R1)  ;DRIVE IS UNSAFE
7317  040214  004737  042004                  JSR     PC,$VRH11       ;SAVE RH11/RP04/5/6 REGISTERS
7318  040220  052762  110000  000016          BIS     #BIT15!BIT12,16(R2)     ;INFORM USER OF UNSAFE ERROR
7319  040226  000423                          BR      SC7
7320  040230  032705  010000          2$:     BIT     #BIT12,R5       ;"MOL" = 1 ?
7321  040234  001015                          BNE     3$              ;BR IF YES
7322  040236  112761  177777  034326          MOVB    #-1,DRVACT(R1)  ;ACTIVE ERROR RECOVER
7323  040244  112761  000001  034336          MOVB    #1,DRVSTA(R1)   ;ONLINE
7324  040252  006301                          ASL     R1
7325  040254  012761  072460  034430          MOV     #30000.,TIMER(R1) ;START 30 SECOND TIMER
7326  040262  006201                          ASR     R1
7327  040264  000137  037626                  JMP     SC4
7328  040270  052762  100220  000016  3$:     BIS     #BIT15!BIT07!BIT04,16(R2)       ;INFORM USER OF ERROR
7329  040276  105061  034326          SC7:    CLRB    DRVACT(R1)      ;DRIVE IS IDLE
7330  040302  004737  042544                  JSR     PC,EMPTYQ       ;DUMP THE QUEUE
7331  040306  105761  034404                  TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS OR QUEUE?
7332  040312  003002                          BGT     1$              ;BR IF NOT
7333  040314  105061  034404                  CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
7334  040320  116164  034452  000016  1$:     MOVB    ATABIT(R1),RPAS(R4)   ;CLEAR ATTENTION BIT
7335  040326  105761  034336                  TSTB    DRVSTA(R1)      ;IS THE DRIVE UNSAFE ?
7336  040332  100406                          BMI     2$              ;BR IF IT IS
7337  040334  012746  000113                  MOV     #113,-(SP)      ;RELEASE COMMAND
7338  040340  004037  041612                  JSR     R0,WRT.RP       ;WRITE THE COMMAND INTO RPCS1
7339  040344  000000                          RPCS1                   ;REGISTER INDEX
7340  040346  040354                          SC8                     ;PARITY EXIT ADDRESS
7341  040350  000137  037626          2$:     JMP     SC4             ;CHECK FOR MORE DRIVES
7342  040354  105761  034326          SC8:    TSTB    DRVACT(R1)      ;IS DRIVE IDLE?
7343  040360  001405                          BEQ     1$              ;YES--BRANCH
7344  040362  004737  042640                  JSR     PC,GETREQ       ;GET DPB POINTER
7345  040366  004737  036654                  JSR     PC,CI7          ;PROCESS THE PARITY ERROR
7346  040372  000402                          BR      2$              ;CONTINUE
7347  040374  004737  036702          1$:     JSR     PC,CI7B         ;PROCESS THE UNCORRECTABLE PARITY ERROR
7348  040400  000137  037626          2$:     JMP     SC4             ;CHECK MORE DRIVES
7349  040404  105761  034404          SC11:   TSTB    ULDFLG(R1)      ;"UNLOAD IN PROGRESS"?
7350  040410  003402                          BLE     1$              ;BRANCH IF NO
7351  040412  105061  034404                  CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
7352  040416  105061  034326          1$:     CLRB    DRVACT(R1)      ;SET DRIVE IDLE
7353  040422  136137  034452  034400          BITB    ATABIT(R1),SRCHWT       ;DOING A SEARCH OPERATION FOR
7354                                                                  ;AN I/O COMMAND?
7355  040430  001012                          BNE     2$              ;BRANCH IF YES
7356  040432  004737  042662                  JSR     PC,POPQUE       ;REMOVE REQUEST FROM QUEUE
7357  040436  052762  000200  000016          BIS     #BIT07,16(R2)   ;SET "DONE" BIT
7358  040444  005737  034424                  TST     SAVEFG          ;SAVE THE REGISTERS?
7359  040450  100002                          BPL     2$              ;BRANCH IF NO
7360  040452  004737  042004                  JSR     PC,SVRH11       ;YES--SAVE ALL OF THE RH11/RP04/5/6 REG'S
7361  040456  116164  034452  000016  2$:     MOVB    ATABIT(R1),RPAS(R4)   ;CLEAR ATTENTION BIT
7362  040464  004737  035544                  JSR     PC,OPT          ;START A REQUEST
7363  040470  000137  037526                  JMP     SC4             ;CHECK FOR MORE DRIVES
7364  040474  010164  000010          SC12:   MOV     R1,RPCS2(R4)    ;SELECT DRIVE
7365  040500  016437  000012  034316          MOV     RPOS1(R4),RPERRS        ;SAVE THE FOUR REGISTERS THAT
7366  040506  016437  000014  034320          MOV     RPER1(R4),RPERRS+2      ;WILL TELL US SOMETHING
7367  040514  016437  000040  034322          MOV     RPER2(R4),RPERRS+4
7368  040522  016437  000042  034324          MOV     RPER3(R4),RPERRS+6
7369  040530  004037  034714                  JSR     R0,DRVINT       ;INIT. THE STATE OF THE DRIVE
7370  040534  000401                          BR      1$              ;TAKE ERROR EXIT
7371  040536  000207                          RTS     PC              ;RETURN
```

```
7372  040540  005726                  1$:     TST     (SP)+           ;POP PC OFF OF THE STACK
7373  040542  000704                          BR      SC8             ;PROCESS THE PARITY ERROR
7374  040544  006301                  SC13:   ASL     R1              ;SETUP TO ADDRESS WORDS
7375  040546  012761  177777  034430          MOV     #-1,TIMER(R1)   ;STOP THE TIMER
7376  040554  006201                          ASR     R1              ;
7377  040556  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT THE DRIVE
7378  040562  116164  034452  000016          MOVB    ATABIT(R1),RPAS(R4)  ;CLEAR THE ATTENTION BIT
7379  040570  032714  004000                  BIT     #BIT11,(R4)     ;DRIVE AVAILABLE ?
7380  040574  001006                          BNE     1$              ;BR IF AVAILABLE
7381  040576  006301                          ASL     R1
7382  040600  012761  023420  034430          MOV     #10000.,TIMER(R1)  ;START 10 SEC TIMER AGAIN
7383  040606  006201                          ASR     R1
7384  040610  000433                          BR      3$              ;EXIT
7385  040612  105761  034356          1$:     TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
7386  040616  001424                          BEQ     2$              ;BR IF NOT
7387  040620  105061  034356                  CLRB    DPINT(R1)       ;CLEAR THE INIT INDICATOR
7388  040624  004037  034714                  JSR     R0,DRVINT       ;GO INIT THE DRIVE
7389  040630  000240                          NOP                     ;DUMMY PARITY ERROR RETURN
7390  040632  105761  034336                  TSTB    DRVSTA(R1)      ;DRIVE ONLINE ?
7391  040636  003014                          BGT     2$              ;BR IF YES -- START ORDER
7392  040640  005702                          TST     R2              ;QUEUE ENTRY FOR THE DRIVE
7393  040642  001416                          BEQ     3$              ;BR IF NOT
7394  040644  004737  042640                  JSR     PC,GETREQ       ;GET DPB ADDRESS
7395  040650  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)  ;INFORM USER THAT DRIVE OFFLINE
7396  040656  004737  042004                  JSR     PC,SVRH11       ;SAVE THE REGISTERS
7397  040662  004737  042544                  JSR     PC,EMPTYQ       ;EMPTY THE REQUEST QUEUE
7398  040666  000404                          BR      3$
7399  040670  105061  034366          2$:     CLRB    DPRQS(R1)       ;CLEAR THE PORT REQUEST INDICATOR
7400  040674  004737  035544                  JSR     PC,OPT          ;START THE PENDING REQUEST
7401  040700  000137  037626          3$:     JMP     SC4             ;PROCESS OTHER DRIVES
7402
7403                                  ;RP04/5/6 TIMER ROUTINE
7404                                  ;CALL
7405                                  ;       MOV     #TIME,-(SP)     ;ELAPSED TIME IN MILLISECONDS ON THE STACK
7406                                  ;       JSR     PC,RPTMR        ;CALL RP04/5/6 TIME ROUTINE
7407
7408  040704  005737  034402          RPTMR:  TST     ACTDRV          ;CHECK "ACTDRV & ACTSTR"
7409  040710  001030                          BNE     4$              ;IF NON ZERO EXIT
7410  040712  112737  000001  034403          MOVB    #1,ACTSTR       ;SET "ACTSTR"
7411  040720  104412                          SAVREG                  ;SAVE R0 - R5
7412  040722  005001                          CLR     R1              ;START WITH DRIVE 0
7413  040724  005003                          CLR     R3
7414  040726  005763  034430          1$:     TST     TIMER(R3)       ;IS THE TIMER RUNNING?
7415  040732  002407                          BLT     2$              ;BRANCH IF NO
7416  040734  166663  000002  034430          SUB     2(SP),TIMER(R3) ;COUNT THE INTERVAL
7417  040742  003003                          BGT     2$              ;BR IF NO SOFTWARE TIMEOUT
7418  040744  004737  040776                  JSR     PC,STO          ;CALL SOFTWARE TIMEOUT ROUTINE
7419  040750  000405                          BR      3$              ;GO TO THE EXIT
7420  040752  005201                  2$:     INC     R1              ;MOVE TO NEXT DRIVE
7421  040754  005723                          TST     (R3)+
7422  040756  022701  000010                  CMP     #8.,R1          ;OUT OF DRIVES?
7423  040762  003361                          BGT     1$              ;BRANCH IF NO
7424  040764  104413                  3$:     RESREG                  ;RESTORE R0 - R5
7425  040766  105037  034403                  CLRB    ACTSTR          ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
7426  040772  012616                  4$:     MOV     (SP)+,(SP)      ;ADJUST THE STACK
7427  040774  000207                          RTS     PC              ;RETURN
```

```
7428
7429                                    ;SOFTWARE TIMEOUT ROUTINE
7430                                    ;
7431                                    ;NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
7432                                    ;       OR GREATER
7433                                    ;
7434                                    ;CALL:  STO
7435                                    ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
7436                                    ;       JSR     PC,STO          ;CALL
7437                                    ;       RETURN
7438
7439 041776 010146                      STO:    MOV     R1,-(SP)        ;SAVE R1
7440 041000 010346                              MOV     R3,-(SP)        ;SAVE R3
7441 041002 013704 034464                       MOV     RPADR,R4        ;GET ADDRESS OF "RPCS1"
7442 041006 010164 000010                       MOV     R1,RPCS2(R4)    ;SELECT THE DRIVE
7443 041012 004037 041432                       JSR     R0,RD.RP        ;READ "DRIVE STATUS REG"
7444 041016 000012                              RPDS1
7445 041020 041320                              STOS
7446 041022 105726                              TSTB    (SP)+           ;IS "DRY"=1?
7447 041024 100477                              BMI     STO2            ;BR IF YES
7448 041026 105761 034356               STO1:   TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
7449 041032 001074                              BNE     STO2            ;BR IF YES
7450 041034 105761 034366                       TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
7451 041040 001071                              BNE     STO2            ;BR IF YES
7452 041042 013702 034376                       MOV     TRNSWT,R2       ;PICKUP TRANSFER WAIT QUEUE
7453 041046 020137 034450                       CMP     R1,DTUW         ;TRANSFER UNDERWAY ON THIS DRIVE?
7454 041052 001402                              BEQ     1$              ;BRANCH IF YES
7455 041054 004737 042640                       JSR     PC,GETREQ       ;GET DPB ADDRESS
7456 041060 052762 101000 000016        1$:     BIS     #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
7457 041066 004737 042004                       JSR     PC,SVRH11       ;SAVE RH11/RP04/5/6 REGISTERS
7458 041072 012764 000040 000010                MOV     #BIT05,RPCS2(R4) ;"INIT" THE MASS BUS
7459 041100 105061 034326                       CLRB    DRVACT(R1)      ;DRIVE IS IDLE
7460 041104 105061 034404                       CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
7461 041110 005001                              CLR     R1              ;START WITH DRIVE 0
7462 041112 005003                              CLR     R3
7463 041114 004037 034714               2$:     JSR     R0,DRVINT       ;INIT. THIS DRIVE
7464 041120 000477                              BR      STO5            ;PARITY ERROR RETURN
7465 041122 105761 034326                       TSTB    DRVACT(R1)      ;DRIVE IDLE BEFORE THE INIT.?
7466 041126 001414                              BEQ     4$              ;YES--BRANCH
7467 041130 013702 034376                       MOV     TRNSWT,R2       ;GET TRANSFER WAIT QUEUE
7468 041134 023701 034450                       CMP     DTUW,R1         ;WAS THERE I/O ON THIS DRIVE?
7469 041140 001402                              BEQ     3$              ;YES--BRANCH
7470 041142 004737 042640                       JSR     PC,GETREQ       ;GET THE DPB POINTER FROM QUEUE
7471 041146 052762 100400 000016        3$:     BIS     #BIT15!BIT08,16(R2) ;INFORM USER OF INIT.
7472 041154 105061 034326                       CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE TO IDLE
7473 041160 105061 034404               4$:     CLRB    ULDFLG(R1)      ;NO UNLOAD
7474 041164 012763 177777 034430                MOV     #-1,TIMER(R3)   ;STOP THE TIMER
7475 041172 005723                              TST     (R3)+           ;UPDATE THE INDEX
7476 041174 005201                              INC     R1              ;INCREMENT THE DRIVE NUMBER
7477 041176 022701 000010                       CMP     #8.,R1          ;LAST DRIVE BEEN CHECKED?
7478 041202 003344                              BGT     2$              ;NO--LOOP
7479 041204 012737 177777 034450                MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
7480 041212 005037 034376                       CLR     TRNSWT          ;CLEAR TRANSFER WAIT QUEUE
7481 041216 004737 042466                       JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
7482 041222 000500                              BR      STO9            ;EXIT
7483 041224 116405 000016               STO2:   MOVB    RPAS(R4),R5     ;READ ATTENTION REG
```

```
7484  041230  136105  034452                  BITB    ATABIT(R1),R5   ;IS ATTENTION FOR THIS DRIVE UP ?
7485  041234  001017                          BNE     ST03            ;YES--BRANCH
7486  041236  105761  034356                  TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
7487  041242  001031                          BNE     ST06            ;BR IF YES - DRIVE NOT ONLINE
7488  041244  105761  034366                  TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
7489  041250  001045                          BNE     ST07            ;BR IF YES - NO RESPONSE TO REQUEST
7490  041252  020137  034450                  CMP     R1,DTUW         ;DATA TRANSFER UNDERWAY FOR THIS DRIVE
7491  041256  001263                          BNE     ST01            ;BR IF NO
7492  041260  004037  041432                  JSR     R0,RD.RP        ;YES--CHECK "RDY"
7493  041264  000000                          RPCS1
7494  041266  041320                          ST05
7495  041270  105726                  ;       TSTB    (SP)+
7496  041272  100255                          BPL     ST01            ;BR IF "RDY"=0
7497  041274  105761  034356          ST03:   TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
7498  041300  001003                          BNE     1$              ;BR IF INIT PENDING
7499  041302  105761  034366                  TSTB    DPRQS(R1)       ;PORT REQUEST PENDING ?
7500  041306  001446                          BEQ     ST09            ;BR IF NOT
7501  041310  012763  177777  034430  1$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
7502  041316  000442                          BR      ST09            ;EXIT
7503  041320  004737  036754          ST05:   JSR     PC,CI8          ;GO HANDLE THE PARITY ERROR
7504  041324  000437                          BR      ST09
7505  041326  105061  034356          ST06:   CLRB    DPINT(R1)       ;CLEAR THE INITIALIZE INDICATOR
7506  041332  105061  034336                  CLRB    DRVSTA(R1)      ;SET UNIT OFFLINE
7507  041336  012763  177777  034430          MOV     #-1,TIMER(R3)   ;STOP THE TIMER
7508  041344  004737  042640                  JSR     PC,GETREQ       ;GET THE DPB ADDRESS
7509  041350  005702                          TST     R2              ;REQUEST IN QUEUE ?
7510  041352  001424                          BEQ     ST09            ;BR IF NOT
7511  041354  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)  ;INFORM THE USER DRIVE NOT AVAILABLE
7512  041362  000414                          BR      ST08            ;FINISH
7513  041364  012763  177777  034430  ST07:   MOV     #-1,TIMER(R3)   ;STOP THE TIMER
7514  041372  105061  034366                  CLRB    DPRQS(R1)       ;CLEAR PORT REQUEST INDICATOR
7515  041376  004737  042640                  JSR     PC,GETREQ       ;GET DPB ADDRESS
7516  041402  005702                          TST     R2              ;QUEUE ENTRY FOR DRIVE ?
7517  041404  001407                          BEQ     ST09            ;BR IF NONE
7518  041406  012762  100004  000016          MOV     #BIT15!BIT2,16(R2)  ;INFORM USER OF PORT REQUEST ERROR
7519  041414  004737  042544          ST08:   JSR     PC,EMPTYQ       ;CLEAR THE QUEUE FOR THE DRIVE
7520  041420  004737  042004                  JSR     PC,SVRH11       ;SAVE THE REGISTERS
7521  041424  012603                  ST09:   MOV     (SP)+,R3        ;RESTORE R3
7522  041426  012601                          MOV     (SP)+,R1        ;RESTORE R1
7523  041430  000207                          RTS     PC              ;RETURN
7524
7525                                  ;ROUTINE TO READ A RH11/RPO4/5/6 REGISTER
7526                                  ;
7527                                  ;CALL
7528                                  ;       JSR     R0,RD.RP        ;GO READ A REGISTER
7529                                  ;       INDEX                   ;REG. INDEX FROM BASE
7530                                  ;       ERRADR                  ;ERROR ADDRESS--PROCESS ERROR STARTING
7531                                  ;                               ;AT THIS ADDRESS
7532                                  ;       RETURN                  ;CONTENTS OF REG. IS ON THE STACK
7533                                  ;
7534  041432  013737  034462  041600  RD.RP:   MOV     MCPEMX,RD.RP2   ;MAX. RETRYS ALLOWED
7535  041440  011646                          MOV     (SP),-(SP)      ;SAVE R0 FOR RETURN
7536  041442  013737  034464  041456          MOV     RPADR,RD.ADR    ;FORM THE DESIRED ADDRESS
7537  041450  062037  041456                  ADD     (R0)+,RD.ADR    ;USING THE BASE AND THE INDEX
7538  041454  013727                  RD.RP1: MOV     @(PC)+,(PC)+    ;READ THE DESIRED REGISTER OF THE RPO4
7539  041456  000000                  RD.ADR: .WORD   0               ;ADDRESS IS FORMED HERE
```

```
7540  041460  000000                    RD.WRD: .WORD   0               ;REQ. CONTENTS PUT HERE
7541  041462  013766  041460  000002            MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
7542  041470  013746  034464                    MOV     RPADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
7543  041474  062716  000010                    ADD     #RPCS2,(SP)     ;FORM THE ADDRESS OF RPCS2
7544  041500  032736  010000                    BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
7545  041504  001037                            BNE     RD.RP3          ;BR IF DRIVE NON-EXISTENT
7546  041506  017746  172752                    MOV     @RPADR,-(SP)    ;READ RPCS1
7547  041512  032716  020000                    BIT     #BIT13,(SP)     ;DID MCPE SET?
7548  041516  001002                            BNE     1$              ;BRANCH IF YES
7549  041520  022620                            CMP     (SP)+,(RO)+     ;ADJUST FOR RETURN
7550  041522  000432                            BR      RD.RP4          ;EXIT
7551  041524                            1$:
7552  041524  004037  042724                    JSR     RO,ES.SAV       ;SAVE THE ADDRESS IN 'ESCAPE'
7553  041530  104003                            ERROR   3               ; REPORT "MCPE" ERROR
7554  041532  005737  034450                    TST     DTLW            ;DATA TRANSFER UNDERWAY?
7555  041536  100405                            BMI     2$              ;NO--BRANCH
7556  041540  032716  040000                    BIT     #BIT14,(SP)     ;NO--"TRE"=1?
7557  041544  001402                            BEQ     2$              ;NO--BRANCH
7558  041546  005726                            TST     (SP)+           ;YES--CLEAN OFF THE STACK AND
7559  041550  000415                            BR      RD.RP3          ;TAKE THE FATAL ERROR EXIT
7560  041552  052716  040000            2$:     BIS     #BIT14,(SP)     ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
7561  041556  000316                            SWAB    (SP)            ;POSITION BEFORE WRITING
7562  041560  013737  034464  041574            MOV     RPADR,3$        ;FORM ADDRESS OF HIGH BYTE
7563  041566  005237  041574                    INC     3$
7564  041572  112637                            MOVB    (SP)+,@(PC)+    ;WRITE THE HIGH BYTE OF RPCS1
7565  041574  000000            3$:     .WORD   0               ;ADDRESS STORAGE
7566  041576  005327                            DEC     (PC)+           ;EXCEEDED MAX. RETRYS
7567  041600  000003            RD.RP2: .WORD   3
7568  041602  002324                            BGE     RD.RP1          ;BRANCH IF NO
7569  041604  011000            RD.RP3: MOV     (RO),RO         ;FATAL ERROR EXIT
7570  041606  012616                            MOV     (SP)+,(SP)
7571  041610  000200            RD.RP4: RTS     RO
7572
7573                                    ;ROUTINE TO WRITE A REGISTER
7574                                    ;
7575                                    ;CALL
7576                                    ;       MOV     DATA,-(SP)      ;DATA TO BE LOADED ON THE STACK
7577                                    ;       JSR     RO,WRT.RP       ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
7578                                    ;       INDEX                   ;INDEX OF THE REGISTER TO BE LOADED
7579                                    ;       ERRADR                  ;ADDRESS TO RETURN TO ON AN ERROR
7580                                    ;       RETURN                  ;ERROR FREE RETURN
7581
7582  041612  013737  034462  041766  WRT.RP: MOV     MCPEMX,WRT.R2   ;MAX RETRYS ALLOWED
7583  041620  016637  000002  041700            MOV     2(SP),WRT.WD    ;SAVE THE WORD TO WRITE
7584  041626  012616                            MOV     (SP)+,(SP)      ;ADJUST THE STACK
7585  041630  012037  041702                    MOV     (RO)+,WRT.RD    ;GET INDEX OF REGISTER TO BE WRITTEN
7586  041634  001015                            BNE     1$              ;BRANCH IF NOT RPCS1
7587  041636  122737  000150  041700            CMPB    #150,WRT.WD     ;IS THE COMMAND FOR DATA TRANSFERS?
7588  041644  002411                            BLT     1$              ;YES--DON'T GET THE OLD A16 & A17, & PSEL
7589  041646  004037  041432                    JSR     RO,RD.RP        ;NO---COMBINE A16&A17, & PSEL WITH
7590  041652  000000                            RPCS1                   ;THE COMMAND BEFORE SENDING IT TO
7591  041654  041774                            WRT.R3                  ;THE RH11/RP04
7592  041656  000316                            SWAB    (SP)
7593  041660  042716  177770                    BIC     #^C7,(SP)
7594  041664  112637  041701                    MOVB    (SP)+,WRT.WD+1
7595  041670  063737  034464  041702    1$:     ADD     RPADR,WRT.RD    ;FORM THE ADDRESS OF THE DISK REG.
```

```
7596  041676  012737                  WRT.R1: MOV     (PC)+,@(PC)+    ;LOAD THE DESIRED REG.
7597  041700  000000                  WRT.WD: .WORD   0               ;WORD TO WRITE GOES HERE
7598  041702  000000                  WRT.AD: .WORD   0               ;ADDRESS IS FORMED HERE
7599  041704  113746  034464                  MOV     RPADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
7600  041710  062716  000010                  ADD     #RPCS2,(SP)     ;FORM THE ADDRESS OF RPCS2
7601  041714  032736  010000                  BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
7602  041720  001025                          BNE     WRT.R3          ;BR IF DRIVE NON-EXISTENT
7603  041722  004037  041432                  JSR     R0,RD.RP        ;CHECK FOR PARITY ERROR ON WRITE
7604  041726  000014                          RPER1
7605  041730  041774                          WRT.R3
7606  041732  032726  000010                  BIT     #BIT03,(SP)+
7607  041736  001420                          BEQ     WRT.R4          ;BRANCH IF "PAR=0"
7608  041740  016037  177776  041752          MOV     -2(R0),1$       ;PICKUP THE INDEX
7609  041746  004037  041432                  JSR     R0,RD.RP        ;READ THE REG.
7610  041752  000000                  1$:     .WORD   0               ;REG. INDEX
7611  041754  041774                          WRT.R3                  ;RETURN TO THIS ADDRESS ON ERROR
7612  041756  004037  042724                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
7613  041762  104004                          ERROR   4               ;REPORT THE PARITY ON WRITE ERROR
7614  041764  005327                          DEC     (PC)+           ;DECREMENT THE ERROR COUNT
7615  041766  000003                  WRT.R2: .WORD   3               ;RETRY COUNTER
7616  041770  002342                          BGE     WRT.R1          ;TRY AGAIN IF NOT FINISHED
7617  041772  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
7618  041774  011000                  WRT.R3: MOV     (R0),R0         ;TAKE THE "PARITY ON WRITE" ERROR EXIT
7619  041776  000401                          BR      WRT.R5          ;EXIT
7620  042000  005720                  WRT.R4: TST     (R0)+           ;ADJUST FOR ERROR FREE EXIT
7621  042002  000200                  WRT.R5: RTS     R0
7622
7623                                  ;ROUTINE TO SAVE THE RH11/RP04/5/6 REGISTERS AS PER DPB+14
7624                                  ;
7625                                  ;CALL
7626                                  ;       MOV     #DPBNUM,R2      ;DPB POINTER TO R2
7627                                  ;       JSR     PC,SVRH11       ;SAVE THE DRIVES REG'S
7628
7629  042004  104412                  SVRH11: SAVREG                  ;SAVE R0 - R5
7630  042006  005702                          TST     R2              ;QUEUE ENTRY FOR THE DRIVE ?
7631  042010  001430                          BEQ     4$              ;BR IF NONE
7632  042012  013704  034464                  MOV     RPADR,R4
7633  042016  111264  000010                  MOVB    (R2),RPCS2(R4)  ;SELECT DRIVE
7634  042022  016203  000014                  MOV     14(R2),R3       ;GET THE ERROR TABLE POINTER
7635  042026  001433                          BEQ     6$              ;EXIT IF NO ADDRESS
7636  042030  005037  042064                  CLR     3$              ;COUNTER & POINTER
7637  042034  023727  042064  000122  1$:     CMP     3$,#RPDB        ;REACHED THE BUFFER REGISTER ?
7638  042042  001006                          BNE     2$              ;BR IF NOT
7639  042044  032764  000200  000010          BIT     #BIT07,RPCS2(R4)  ; OR' SET ?
7640  042052  001002                          BNE     2$              ;BR IF SET
7641  042054  005023                          CLR     (R3)+           ;STORE RPDB AS ZEROES
7642  042056  000405                          BR      4$              ;CONTINUE
7643  042060  004037  041432          2$:     JSR     R0,RD.RP        ;READ THE SELECTED REGISTER
7644  042064  000000                  3$:     .WORD   0               ;REGISTER INDEX
7645  042066  042112                          5$                      ;ERROR RETURN ADDRESS
7646  042070  012623                          MOV     (SP)+,(R3)+     ;STORE THE REGISTER CONTENTS
7647  042072  023727  042064  000046  4$:     CMP     3$,#RPEC2       ;REACHED THE END ?
7648  042100  001406                          BEQ     6$              ;BR IF YES
7649  042102  062737  000002  042064          ADD     #2,3$           ;INCREMENT THE REGISTER INDEX
7650  042110  000751                          BR      1$              ;CONTINUE READING THE REGISTERS
7651  042112  004737  036654          5$:     JSR     PC,CI7          ;PROCESS THE UNCORRECTABLE PARITY ERROR
```

```
 7652  042116  104413                  6$:     RESREG                  ;RESTORE R0 - R5
 7653  042120  000207                          RTS     PC              ;RETURN
 7654
 7655                                  ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
 7656                                  ;CALL
 7657                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 7658                                  ;       JSR     PC,SET.IE       ;SET "IE"
 7659                                  ;       RETURN
 7660
 7661  042122  010446                  SET.IE: MOV     R4,-(SP)        ;SAVE R4
 7662  042124  013704  034464                  MOV     RPADR,R4        ;PICKUP ADDRESS OF RPCS1
 7663  042130  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
 7664  042134  011446                          MOV     (R4),-(SP)      ;READ RPCS1
 7665  042136  052716  040000                  BIS     #BIT14,(SP)     ;SET THE "TRE" BIT OF THE WORD READ
 7666  042142  000316                          SWAB    (SP)            ;ADJUST FOR DATO
 7667  042144  112714  000100                  MOVB    #BIT06,(R4)     ;SET "IE"
 7668  042150  032764  010000  000010          BIT     #BIT12,RPCS2(R4) ;IS "NED"=1?
 7669  042156  001002                          BNE     1$              ;YES--CLEAR "TRE"
 7670  042160  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
 7671  042162  000402                          BR      2$
 7672  042164  112664  000001          1$:     MOVB    (SP)+,1(R4)     ;CLEAR "TRE"
 7673  042170  012604                  2$:     MOV     (SP)+,R4        ;RESTORE R4
 7674  042172  000207                          RTS     PC              ;RETURN TO CALLER
 7675
 7676                                  ;QUEUE COUNT
 7677  042174      000                 QCNT:   .BYTE   0               ;DRIVE 0
 7678  042175      000                         .BYTE   0               ;DRIVE 1
 7679  042176      000                         .BYTE   0               ;DRIVE 2
 7680  042177      000                         .BYTE   0               ;DRIVE 3
 7681  042200      000                         .BYTE   0               ;DRIVE 4
 7682  042201      000                         .BYTE   0               ;DRIVE 5
 7683  042202      000                         .BYTE   0               ;DRIVE 6
 7684  042203      000                         .BYTE   0               ;DRIVE 7
 7685
 7686                                  ;QUEUE INPUT POINTERS
 7687
 7688  042204  042266                  QINPT:  .WORD   QDRV0            ;DRIVE 0
 7689  042206  042306                          .WORD   QDRV1           ;DRIVE 1
 7690  042210  042326                          .WORD   QDRV2           ;DRIVE 2
 7691  042212  042346                          .WORD   QDRV3           ;DRIVE 3
 7692  042214  042366                          .WORD   QDRV4           ;DRIVE 4
 7693  042216  042406                          .WORD   QDRV5           ;DRIVE 5
 7694  042220  042426                          .WORD   QDRV6           ;DRIVE 6
 7695  042222  042446                          .WORD   QDRV7           ;DRIVE 7
 7696
 7697                                  ;QUEUE OUTPUT POINTERS
 7698
 7699  042224  042266                  QOUTPT: .WORD   QDRV0            ;DRIVE 0
 7700  042226  042306                          .WORD   QDRV1           ;DRIVE 1
 7701  042230  042326                          .WORD   QDRV2           ;DRIVE 2
 7702  042232  042346                          .WORD   QDRV3           ;DRIVE 3
 7703  042234  042366                          .WORD   QDRV4           ;DRIVE 4
 7704  042236  042406                          .WORD   QDRV5           ;DRIVE 5
 7705  042240  042426                          .WORD   QDRV6           ;DRIVE 6
 7706  042242  042446                          .WORD   QDRV7           ;DRIVE 7
 7707
```

```
7708  042244  042266                  QSTART: .WORD   QDRV0           ;DRIVE 0 START ADDRESS
7709  042246  042306                  QSTOP:  .WORD   QDRV1           ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
7710  042250  042326                          .WORD   QDRV2           ;STOP DRIVE 1--START DRIVE 2
7711  042252  042346                          .WORD   QDRV3           ;STOP DRIVE 2--START DRIVE 3
7712  042254  042366                          .WORD   QDRV4           ;STOP DRIVE 3--START DRIVE 4
7713  042256  042406                          .WORD   QDRV5           ;STOP DRIVE 4--START DRIVE 5
7714  042260  042426                          .WORD   QDRV6           ;STOP DRIVE 5--START DRIVE 6
7715  042262  042446                          .WORD   QDRV7           ;STOP DRIVE 6--START DRIVE 7
7716  042264  042466                          .WORD   QTERM           ;STOP DRIVE 7
7717
7718                                  ;DRIVE REQUEST QUEUES
7719
7720  042266  000010                  QDRV0:  .BLKW   10
7721  042306  000010                  QDRV1:  .BLKW   10
7722  042326  000010                  QDRV2:  .BLKW   10
7723  042346  000010                  QDRV3:  .BLKW   10
7724  042366  000010                  QDRV4:  .BLKW   10
7725  042406  000010                  QDRV5:  .BLKW   10
7726  042426  000010                  QDRV6:  .BLKW   10
7727  042446  000010                  QDRV7:  .BLKW   10
7728          042466                  QTERM=.
7729
7730                                  ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
7731                                  ;
7732                                  ;CALL
7733                                  ;       JSR     PC,CLRQUE
7734
7735  042466  104412                  CLRQUE: SAVREG                  ;SAVE R0 - R5
7736  042470  012702  042174                  MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
7737  042474  005022                          CLR     (R2)+           ;DRIVES 0 & 1
7738  042476  005022                          CLR     (R2)+           ;DRIVES 2 & 3
7739  042500  005022                          CLR     (R2)+           ;DRIVES 4 & 5
7740  042502  005022                          CLR     (R2)+           ;DRIVES 6 & 7
7741  042504  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING
7742  042510  012701  042244                  MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
7743  042514  012122                  1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
7744  042516  005303                          DEC     R3
7745  042520  001375                          BNE     1$
7746  042522  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
7747  042526  012701  042244                  MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
7748  042532  012122                  2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
7749  042534  005303                          DEC     R3
7750  042536  001375                          BNE     2$
7751  042540  104413                          RESREG                  ;RESTORE R0 - R5
7752  042542  000207                          RTS     PC
7753
7754                                  ;EMPTY THE QUEUE SPECIFIED BY R1
7755                                  ;
7756                                  ;CALL
7757                                  ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
7758                                  ;       JSR     PC,EMPTYQ
7759
7760  042544  105061  042174          EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
7761  042550  006301                          ASL     R1
7762  042552  016161  042204  042224          MOV     QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
7763  042560  006201                          ASR     R1
```

```
7764  042562  000207                          RTS     PC
7765
7766                                  ;ROUTINE TO PUT A REQUEST IN QUEUE
7767                                  ;
7768                                  ;CALL
7769                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
7770                                  ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
7771                                  ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
7772                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
7773                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
7774
7775  042564  122761  000010  042174 DRVQUE: CMPB    #10,QCNT(R1)    ;IS QUEUE FULL?
7776  042572  001421                  BEQ     2$              ;BR IF YES-TAKE RETURN1
7777  042574  105261  042174          INCB    QCNT(R1)        ;INCREMENT QUEUE COUNT
7778  042600  006301                  ASL     R1
7779  042602  010271  042204          MOV     R2,@QINPT(R1)   ;PUT THIS REQUEST IN QUEUE
7780  042606  062761  000002  042204  ADD     #2,QINPT(R1)    ;UPDATE THE QUEUE POINTER
7781  042614  026161  042204  042246  CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
7782  042622  001003                  BNE     1$              ;BRANCH IF NO
7783  042624  016161  042244  042204  MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
7784  042632  006201          1$:     ASR     R1
7785  042634  005720                  TST     (R0)+           ;TAKE RETURN 2
7786  042636  000200          2$:     RTS     R0              ;RETURN TO USER
7787
7788                                  ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
7789                                  ;
7790                                  ;CALL
7791                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
7792                                  ;       JSR     PC,GETREQ       ;GO GET THE REQUEST
7793                                  ;       RETURN                  ;R2="DPB" ADDRESS OF THE REQUEST
7794                                  ;                               ;R2=0 IF NO REQUEST IN QUEUE
7795
7796  042640  005002          GETREQ: CLR     R2
7797  042642  105761  042174          TSTB    QCNT(R1)        ;IS THERE ANY REQUEST IN QUEUE?
7798  042646  001404                  BEQ     2$              ;NO---BRANCH
7799  042650  006301          1$:     ASL     R1
7800  042652  017102  042224          MOV     @QOUTPT(R1),R2  ;PICK UP "DPB" POINTER FOR THIS DRIVE
7801  042656  006201                  ASR     R1
7802  042660  000207          2$:     RTS     PC              ;RETURN TO USER
7803
7804                                  ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
7805                                  ;
7806                                  ;CALL
7807                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
7808                                  ;       JSR     PC,POPQUE       ;CALL TO REMOVE REQUEST
7809                                  ;       RETURN                  ;R2=ADDRESS OF DPB REMOVED
7810
7811  042662  105361  042174          POPQUE: DECB    QCNT(R1)        ;DECREMENT QUEUE COUNT
7812  042666  006301                  ASL     R1
7813  042670  017102  042224          MOV     @QOUTPT(R1),R2  ;GET THE "DPB" POINTER
7814  042674  062761  000002  042224  ADD     #2,QOUTPT(R1)   ;UPDATE THE QUEUE POINTER
7815  042702  026161  042224  042246  CMP     QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
7816  042710  001003                  BNE     1$              ;NO--BRANCH TO EXIT
7817  042712  016161  042244  042224  MOV     QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
7818  042720  006201          1$:     ASR     R1
7819  042722  000207                  RTS     PC              ;RETURN TO USER
```

```
                                        ;ROUTINE TO SAVE THE CONTENTS OF 'SESCAPE' WHEN THE DRIVER
                                        ;REPORTS AN ERROR DIRECTLY.
                                        ;
                                        ;CALL
                                        ;       JSR     R0,ES.SAV
                                        ;       ERROR   N                       ;;THE ERROR CALL
                                        ;       RETURN                          ;THE RETURN IS PAST THE ERROR CALL

042724  012037  042740        ES.SAV: MOV     (R0)+,1$                ;GET THE ERROR CALL
042730  013746  001206                MOV     $ESCAPE,-(SP)           ;SAVE THE ADDRESS IN 'SESCAPE'
042734  005037  001206                CLR     $ESCAPE                 ;CLEAR THE ESCAPE RETURN
042740  000000                1$:     .WORD   0                       ;THE ERROR CALL IS MOVED HERE
042742  012637  001206                MOV     (SP)+,$ESCAPE           ;RESTORE THE ESCAPE ADDRESS
042746  000200                        RTS     R0                      ;RETURN

                                        .SBTTL  ASCIZ MESSAGES

042750  000122                MSG.R:  .ASCIZ  /R/
042752  041506  000000        MSG.FC: .ASCIZ  /FC/
042755  000114  000103        MSG.LC: .ASCIZ  /LC/
042760  041506  000047        MSGFCP: .ASCIZ  /FC'/
042764  041514  000047        MSGLCP: .ASCIZ  /LC'/
042770  041511  000000        MSG.IC: .ASCIZ  /IC/
042773  000106  000124        MSG.FT: .ASCIZ  /FT/
042776  052114  000000        MSG.LT: .ASCIZ  /LT/
043001  000111  000124        MSG.IT: .ASCIZ  /IT/
043004  051506  000000        MSG.FS: .ASCIZ  /FS/
043007  000114  000123        MSG.LS: .ASCIZ  /LS/
043012  040520  000124        MSG.PAT:        .ASCIZ  /PAT/
043016  000075                MSG.EQ: .ASCIZ  /=/
043020  005015  047503  052116  MSG.CS: .ASCIZ  <CR><LF>/CONTROL SWITCHES=/
043026  047522  020114  053523
043034  052111  044103  051505
043042  000075

043044  027440  000040        SLASH:  .ASCIZ  @ / @
043050  047125  052111  051440  SYSTAT: .ASCIZ  /UNIT STATUS:/<CR><LF><LF>
043056  040524  052524  035123
043064  005015  000012
043070  051104  053111  000105  UNTMSG: .ASCIZ  /DRIVE/
043076  047440  043106  044514  UNTOFF: .ASCIZ  / OFFLINE/
043104  042516  000000
043107  000040  047117  044514  UNTON:  .ASCIZ  / ONLINE/
043114  042516  000000
043117  000040  047516  020124  NOTPRS: .ASCIZ  / NOT PRESENT/
043124  051120  051525  047105
043132  000124
043134  042440  051516  043101  NOTSAF: .ASCIZ  / UNSAFE/
043142  000105
043144  047040  052117  051040  NOTRP:  .ASCIZ  @ NOT RP04/5/6@
043152  030120  027464  027465
043160  000066
043162  050122  032060  000000  RP04$:  .ASCIZ  /RP04/
043167  000122  030120  000065  RP05:   .ASCIZ  /RP05/
```

```
 7876  043174  050122  033060     000  RP06:   .ASCIZ  /RP06/
 7877  043201     015  042012  044522  DRIVES: .ASCIZ  <CR><LF>/DRIVE(S) TO BE TESTED /
 7878  043206  042526  051450  020051
 7879  043214  047524  041040  020105
 7880  043222  042524  052123  042105
 7881  043230  000040
 7882  043232  047516  042516     000  NONE:   .ASCIZ  /NONE/
 7883  043237     054     000          COMMA:  .ASCIZ  /,/
 7884  043241     015  047012  020117  NOCLOK: .ASCIZ  <CR><LF>/NO KW11-P CLOCK, TIMING TESTS WILL NOT BE PERFORMED/
 7885  043246  053513  030461  050055
 7886  043254  041440  047514  045503
 7887  043262  020054  044524  044515
 7888  043270  043516  052040  051505
 7889  043276  051524  053440  046111
 7890  043304  020114  047516  020124
 7891  043312  042502  050040  051105
 7892  043320  047506  046522  042105
 7893  043326     000
 7894  043327     015  005012  042524  TESTNG: .ASCIZ  <CR><LF><LF>/TESTING DRIVE /
 7895  043334  052123  047111  020107
 7896  043342  051104  053111  020105
 7897  043350     000
 7898  043351     123  051105  040511  SERIAL: .ASCIZ  /SERIAL NUMBER /
 7899  043356  020114  052516  041115
 7900  043364  051105  000040
 7901
 7902  043370  005015  051012  052117  MSG7:   .ASCIZ  <CR><LF><LF>/ROTATIONAL SPEED TIMES/
 7903  043376  052101  047511  040516
 7904  043404  020114  050123  042105
 7905  043412  020104  044524  042515
 7906  043420  000123
 7907  043432  005015  047412  042516  MSG10A: .ASCIZ  <CR><LF><LF>/ONE CYLINDER SEEK TIMES/<CR><LF>/ * FORWARD/
 7908  043430  041440  046131  047111
 7909  043436  042504  020122  041523
 7910  043444  045505  052040  040511
 7911  043452  051505  005015  020040
 7912  043460  043040  051117  043527
 7913  043466  042122     000
 7914  043471     015  020012  020052  MSG10B: .ASCIZ  <CR><LF>/ * REVERSE/
 7915  043476  042522  042526  051522
 7916  043504  000105
 7917  043506  005015  040412  041503  MSG11A: .ASCIZ  <CR><LF><LF>/ACCESS TIME MEASURMENT/<CR><LF>/ * FORWARD/
 7918  043514  051505  020123  044524
 7919  043522  042515  046440  040505
 7920  043530  052523  046522  041105
 7921  043536  006524  020012  020052
 7922  043544  047506  053522  051101
 7923  043552  000104
 7924  043554  005015  025040  051040  MSG11B: .ASCIZ  <CR><LF>/ * REVERSE/
 7925  043562  053105  051105  042523
 7926  043570     000
 7927  043571     015  005012  040515  MSG12A: .ASCIZ  <CR><LF><LF>/MAXIMUM SEEK TIMES/<CR><LF>/ * FORWARD/
 7928  043576  044530  052515  020115
 7929  043604  042523  045505  052040
 7930  043612  046511  051505  005015
 7931  043620  025040  043040  051117
```

```
7932  043626  040527  042122     000
7933  043633     015  020012  020052  MSG12B: .ASCIZ  <CR><LF>/ * REVERSE/
7934  043640  042522  042526  051522
7935  043646  000105
7936
7937  043650  005015  044515  036516  MSGMIN: .ASCIZ  <CR><LF>/MIN=/
7938  043656     000
7939  043657     015  046412  054101  MSGMAX: .ASCIZ  <CR><LF>/MAX=/
7940  043664  000075
7941  043666  005015  053101  036507  MSGAVG: .ASCIZ  <CR><LF>/AVG=/
7942  043674     000
7943  043675     060  052440  000123  MSGOUS: .ASCIZ  /0 US/
7944  043703  041040  046105  053517  MBELOW: .ASCIZ  / BELOW THE MINIMUM OF /
7945  043710  052040  042510  046440
7946  043716  047111  046511  046525
7947  043724  047440  020106     000
7948  043731     040  041101  053117  MABOVE: .ASCIZ  / ABOVE THE MAXIMUM OF /
7949  043736  020105  044124  020105
7950  043744  040515  044530  052515
7951  043752  020115  043117  000040
7952  043760  051440  042505  051513  MSGNUM: .ASCIZ  / SEEKS TIMED/
7953  043766  052040  046511  042105
7954  043774     000
7955  043775     040  047516  020124  MSGNON: .ASCIZ  / NOT TIMED/
7956  044002  044524  042515  000104
7957  044010  020040     000          MSG.SP: .ASCIZ  /  /            ;TWO (2) SPACES
7958
7959                                  .SBTTL  ERROR HEADER (EM) MESSAGES
7960
7961  044013     122  030510  020061  EM1:    .ASCIZ  /RH11 INTERRUPT OCCURRED (RPAS = 0)/
7962  044020  047111  042524  051122
7963  044026  050125  020124  041517
7964  044034  052503  051122  042105
7965  044042  024040  050122  051501
7966  044050  036440  030040  000051
7967  044056  047125  054105  042520  EM2:    .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
7968  044064  052103  042105  046440
7969  044072  052124  047105  044524
7970  044100  047117  047440  041503
7971  044106  051125  042522  000104
7972  044114  040515  051523  052502  EM3:    .ASCIZ  /MASSBUS PARITY ERROR(MCPE=1)/
7973  044122  020123  040520  044522
7974  044130  054524  042440  051122
7975  044136  051117  046450  050103
7976  044144  036505  024461     000
7977  044151     115  051501  041123  EM4:    .ASCIZ  /MASSBUS PARITY ERROR(PAR=1)/
7978  044156  051525  050040  051101
7979  044164  052111  020131  051105
7980  044172  047522  024122  040520
7981  044200  036522  024461     000
7982  044205     101  042104  042522  EM5:    .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
7983  044212  051523  050040  052514
7984  044220  020107  044103  047101
7985  044226  042507  041040  052111
7986  044234  051440  052105     000
7987  044241     122  030510  027461  EM10:   .ASCIZ  "RH11/RP04/5/6 FAILED TO RESPOND TO ADDRESSING"
```

```
7988  044246  050122  032060  032457
7989  044254  033057  043040  044501
7990  044262  042514  020104  047524
7991  044270  051040  051505  047520
7992  044276  042116  052040  020117
7993  044304  042101  051104  051505
7994  044312  044523  043516     000
7995  044317     104  044522  042526  EM11:   .ASCIZ  /DRIVE SELECTED IS NOT ONLINE/
7996  044324  051440  046105  041505
7997  044332  042524  020104  051511
7998  044340  047040  052117  047440
7999  044346  046116  047111  000105
8000  044354  046511  051120  050117  EM12:   .ASCIZ  /IMPROPER HEADER DATA/
8001  044362  051105  044040  040505
8002  044370  042504  020122  040504
8003  044376  040524     000
8004  044401     104  052101  020101  EM13:   .ASCIZ  /DATA COMPARE FAILURE/
8005  044406  047503  050115  051101
8006  044414  020105  040506  046111
8007  044422  051125  000105
8008  044426  044504  045523  042440  EM17:   .ASCIZ  /DISK ERROR IN TIMING TEST/
8009  044434  051122  051117  044440
8010  044442  020116  044524  044515
8011  044450  043516  052040  051505
8012  044456  000124
8013  044460  046103  041517  020113  EM20:   .ASCIZ  /CLOCK (KW11-P) OVERFLOW IN TIMING TEST/
8014  044466  045450  030527  020461
8015  044474  024520  047440  042526
8016  044502  043122  047514  020127
8017  044510  047111  052040  046511
8018  044516  047111  020107  042524
8019  044524  052123     000
8020  044527     104  051511  020113  EM23:   .ASCIZ  /DISK ERROR DURING SEEK/
8021  044534  051105  047522  020122
8022  044542  052504  044522  043516
8023  044550  051440  042505  000113
8024  044556  042523  045505  047040  EM24:   .ASCIZ  /SEEK NOT COMPLETE WITHIN 120 MS/
8025  044564  052117  041440  046517
8026  044572  046120  052105  020105
8027  044600  044527  044124  047111
8028  044606  030440  030062  046440
8029  044614  000123
8030  044616  044122  030461  051057  EM41:   .ASCIZ  "RH11/RP04/5/6 ERROR"
8031  044624  030120  027464  027465
8032  044632  020066  051105  047522
8033  044640  000122
8034  044642  040506  040524  020114  EM46:   .ASCIZ  /FATAL WRITE CHECK ERROR/
8035  044650  051127  052111  020105
8036  044656  044103  041505  020113
8037  044664  051105  047522  000122
8038
8039                                  .SBTTL  STATUS/ERROR INDICATOR MESSAGES
8040
8041  044672  043117  046106  047111  MSGB14: .ASCIZ  /OFFLINE OR UNSAFE DRIVE REQUESTED/
8042  044700  020105  051117  052440
8043  044706  051516  043101  020105
```

```
8044  044714  051104  053111  020105
8045  044722  042522  052521  051505
8046  044730  042524  000104
8047  044734  047125  047514  042101  MSGB13: .ASCIZ  /UNLOADED DRIVE REQUESTED/
8048  044742  042105  042040  044522
8049  044750  042523  051040  050505
8050  044756  042525  052123  042105
8051  044764     000
8052  044765     120  051105  044523  MSGB12: .ASCIZ  /PERSISTENT UNSAFE/
8053  044772  052123  047105  020124
8054  045000  047125  040523  042506
8055  045006     000
8056  045007     120  051101  052111  MSGB11: .ASCIZ  /PARITY ERROR OCCURRED/
8057  045015  020131  051105  047522
8058  045023  020122  041517  052503
8059  045031  051122  042105     000
8060  045035     106  052101  046101  MSGB10: .ASCIZ  /FATAL PARITY ERROR/
8061  045042  050040  051101  052111
8062  045050  020131  051105  047522
8063  045056  000122
8064  045060  047523  052106  040527  MSGB09: .ASCIZ  /SOFTWARE TIMEOUT ON THIS DRIVE/
8065  045066  042522  052040  046511
8066  045074  047505  052125  047440
8067  045102  020116  044124  051511
8068  045110  042040  044522  042526
8069  045116     000
8070  045117     123  043117  053524  MSGB08: .ASCIZ  /SOFTWARE TIMEOUT ON ANOTHER DRIVE/
8071  045125  051101  020105  044524
8072  045132  042515  052517  020124
8073  045140  047117  040440  047516
8074  045146  044124  051105  042040
8075  045154  044522  042526     000
8076  045161     105  051122  051117  MSGB06: .ASCIZ  "ERROR OCCURRED DURING I/O OPERATION"
8077  045166  047440  041503  051125
8078  045174  042522  020104  052504
8079  045202  044522  043516  044440
8080  045210  047457  047440  042520
8081  045216  040522  044524  047117
8082  045224     000
8083  045225     105  051122  051117  MSGB05: .ASCIZ  "ERROR OCCURRED DURING NON-I/O OPERATION"
8084  045232  047440  041503  051125
8085  045240  042522  020104  052504
8086  045246  044522  043516  047040
8087  045254  047117  044455  047457
8088  045262  047440  042520  040522
8089  045270  044524  047117     000
8090  045275     125  051516  043101  MSGB04: .ASCIZ  /UNSAFE OCCURRED/
8091  045302  020105  041517  052503
8092  045310  051122  042105     000
8093  045315     101  052125  046517  MSGB03: .ASCIZ  /AUTOMATIC RECALIBRATE SEQUENCE OCCURRED/
8094  045322  052101  041511  051040
8095  045330  041505  046101  041111
8096  045336  040522  042524  051440
8097  045344  050505  042525  041516
8098  045352  020105  041517  052503
8099  045360  051122  042105     000
```

```
8100  045365     104  044522  042526  MSGB02: .ASCIZ  /DRIVE HAS NOT RESPONDED TO PORT REQUEST/
8101  045372  044040  051501  047040
8102  045400  052117  051040  051505
8103  045406  047520  042116  042105
8104  045414  052040  020117  047520
8105  045422  052122  051040  050505
8106  045430  042525  052123     000
8107  045435     104  044522  042526  MSGB01: .ASCIZ  /DRIVE HAS BECOME NON-EXISTENT/
8108  045442  044040  051501  041040
8109  045450  041505  046517  020105
8110  045456  047516  026516  054105
8111  045464  051511  042524  052116
8112  045472     000
8113
8114                                  .SBTTL  DATA HEADER (DT) MESSAGES
8115
8116  045473     105  051122  050040  DH1:    .ASCIZ  /ERR PC  RPAS/
8117  045500  020103  051040  040520
8118  045506  000123
8119  045510  051105  020122  041520  DH2:    .ASCIZ  /ERR PC  DRIVE   RPAS    RPDS1   RPER1   RPER2   RPER3/
8120  045516  020040  051104  053111
8121  045524  020105  020040  050122
8122  045532  051501  020040  020040
8123  045540  050122  051504  020061
8124  045546  020040  050122  051105
8125  045554  020061  020040  050122
8126  045562  051105  020062  020040
8127  045570  050122  051105  000063
8128  045576  042524  052123  020040  DH3:    .ASCIZ  /TEST    ERR PC  ADDRESS DATA/
8129  045604  020040  051105  020122
8130  045612  041520  020040  042101
8131  045620  051104  051505  020123
8132  045626  040504  040524     000
8133  045633     124  051505  020124  DH4:    .ASCIZ  /TEST    ERR PC  ADDRESS GDDATA  BDDATA/
8134  045640  020040  042440  051122
8135  045646  050040  020103  040440
8136  045654  042104  042522  051523
8137  045662  043440  042104  052101
8138  045670  020101  041040  042104
8139  045676  052101  000101
8140  045702  050122  051503  020061  DH10:   .ASCIZ  /RPCS1   ERR PC/
8141  045710  020040  051105  020122
8142  045716  041520     000
8143  045721     104  044522  042526  DH11:   .ASCIZ  /DRIVE   ERR PC/
8144  045726  020040  042440  051122
8145  045734  050040  000103
8146  045740  042524  052123  020040  DH12:   .ASCIZ  /TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR/
8147  045746  020040  051105  020122
8148  045754  041520  020040  051524
8149  045762  020124  041520  020040
8150  045770  051104  053111  020105
8151  045776  020040  054503  047114
8152  046004  051104  020040  051124
8153  046012  041501  020113  020040
8154  046020  042523  052103  051117
8155  046026     000
```

```
8156  046027    107   041504  046131  DH12A:  .ASCIZ  /GDCYL   GDTRK   GDSCTR  BDCYL   BDTRK   BDSCTR/
8157  046034  020040  043440  052104
8158  046042  045522  020040  043440
8159  046050  051504  052103  020122
8160  046056  041040  041504  046131
8161  046064  020040  041040  052104
8162  046072  045522  020040  041040
8163  046100  051504  052103  000122
8164  046106  042107  040504  020124  DH13A:  .ASCIZ  /GDDAT   BDDAT   WRDCNT  GDADR   BDADR/
8165  046114  020040  042102  040504
8166  046122  020124  020040  051127
8167  046130  041504  052116  020040
8168  046136  042107  042101  020122
8169  046144  020040  042102  042101
8170  046152  000122
8171  046154  042524  052123  020040  DH17:   .ASCIZ  /TEST    ERR PC  DRIVE   RPCS1   RPDS1   RPER1   RPER2   RPER3/
8172  046162  020040  051105  020122
8173  046170  041520  020040  051104
8174  046176  053111  020105  020040
8175  046204  050122  051503  020061
8176  046212  020040  050122  051504
8177  046220  020061  020040  050122
8178  046226  051105  020061  020040
8179  046234  050122  051105  020062
8180  046242  020040  050122  051105
8181  046250  000063
8182  046252  042524  052123  020040  DH21:   .ASCIZ  /TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK/
8183  046260  020040  051105  020122
8184  046266  041520  020040  051524
8185  046274  020124  041520  020040
8196  046302  051104  053111  020105
8187  046310  020040  054503  047114
8188  046316  051104  020040  051124
8189  046324  041501  000113
8190  046330  042107  040504  020124  DH21A:  .ASCIZ  /GDDAT   BDDAT   WRDCNT  SECTOR/
8191  046336  020040  042102  040504
8192  046344  020124  020040  051127
8193  046352  041504  052116  020040
8194  046360  042523  052103  051117
8195  046366     000
8196  046367     124  051505  020124  DH23:   .ASCIZ  /TEST    ERR PC  DRIVE   CYLNDR  RPCS1   RPCS2   RPDS1/
8197  046374  020040  042440  051122
8198  046402  050040  020103  042040
8199  046410  044522  042526  020040
8200  046416  041440  046131  042116
8201  046424  020122  051040  041520
8202  046432  030523  020040  051040
8203  046440  041520  031123  020040
8204  046446  051040  042120  030523
8205  046454     000
8206  046455     122  042520  030522  DH23A:  .ASCIZ  /RPER1   RPER2   RPER3   RPCA    RPCC/
8207  046462  020040  051040  042520
8208  046470  031122  020040  051040
8209  046476  042520  031522  020040
8210  046504  051040  041520  020101
8211  046512  020040  051040  041520
```

```
8212  046520  000103
8213  046522  042524  052123  020040  DH41:   .ASCIZ  /TEST   ERR PC  TST PC  DRIVE/
8214  046530  020040  051105  020122
8215  046536  041520  020040  051524
8216  046544  020124  041520  020040
8217  046552  051104  053111  000105
8218  046560  042524  052123  020040  DH42:   .ASCIZ  /TEST   ERR PC  TST PC  DRIVE   RPCS1   RPCS2   RPDS1/
8219  046566  020040  051105  020122
8220  046574  041520  020040  051524
8221  046602  020124  041520  020040
8222  046610  051104  053111  020105
8223  046616  020040  050122  051503
8224  046624  020061  020040  050122
8225  046632  051503  020062  020040
8226  046640  050122  051504  000061
8227  046646  050122  051105  020061  DH43A:  .ASCIZ  /RPER1   RPER2   RPER3/
8228  046654  020040  050122  051105
8229  046662  020062  020040  050122
8230  046670  051105  000063
8231  046674  050122  051503  020061  DH44A:  .ASCIZ  /RPCS1   RPCS2   RPDS1   RPCC    RPCA    RPDA/
8232  046702  020040  050122  051503
8233  046710  020062  020040  050122
8234  046716  051504  020061  020040
8235  046724  050122  041503  020040
8236  046732  020040  050122  040503
8237  046740  020040  020040  050122
8238  046746  040504     000
8239  046751     122  042520  030522  DH44B:  .ASCIZ  /RPER1   RPER2   RPER3/
8240  046756  020040  051040  042520
8241  046764  031122  020040  051040
8242  046772  042520  031522     000
8243  046777     122  042520  030522  DH45A:  .ASCIZ  /RPER1   RPER2   RPER3   RPWC    RPBA    RPDB/
8244  047004  020040  051040  042520
8245  047012  031122  020040  051040
8246  047020  042520  031522  020040
8247  047026  051040  053520  020103
8248  047034  020040  051040  041120
8249  047042  020101  020040  051040
8250  047050  042120  000102
8251
8252
8253                          .EVEN
8254
8255                          .SBTTL  DATA TABLE (DT)
8256
8257  047054  001116  001170          DT1:    .WORD   $ERRPC,$REG3
8258  047060  001116  001164  001170  DT2:    .WORD   $ERRPC,$REG1,$REG3,RPERRS,RPERRS+2,RPERRS+4,RPERRS+6
8259  047066  034316  034320  034322
8260  047074  034324
8261  047076  001176  001116  041456  DT3:    .WORD   $TMPO,$ERRPC,RD.ADR,RD.WRD
8262  047104  041460
8263  047106  001176  001116  041702  DT4:    .WORD   $TMPC,$ERRPC,WRT.ADR,WRT.WD,RD.WRD
8264  047114  041700  041460
8265  047120  001176  001116  001164  DT5:    .WORD   $TMPO,$ERRPC,$REG1,$REG5,RPERRS,RPERRS+2,RPERRS+4,RPERRS+6
8266  047126  001174  034316  034320
8267  047134  034322  034324
```

```
8268  047140  001366  001116          DT10:   .WORD   RH.ADR,$ERRPC
8269  047144  001166  001116          DT11:   .WORD   $REG2,$ERRPC
8270  047150  001176  001116  001162  DT12:   .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
8271  047156  001254  001270  001274
8272  047164  001272
8273  047166  001270  001274  001272  DT12A:  .WORD   CYL.DS,TRK.DS,SEC.DS,CYL.RD,TRK.RD,SEC.RD
8274  047174  001262  001264  001266
8275  047202  001176  001116  001162  DT13:   .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
8276  047210  001254  001270  001274
8277  047216  001272
8278  047220  001124  001126  001172  DT13A:  .WORD   $GDDAT,$BDDAT,$REG4,$GDADR,$BDADR
8279  047226  001120  001122
8280  047232  001176  001116  001254  DT17:   .WORD   $TMPO,$ERRPC,CHKDRV,RP.REG,RP.REG+12,RP.REG+14,RP.REG+40,RP.REG+42
8281  047240  004204  004216  004220
8282  047246  004244  004246
8283  047252  001176  001116  001162  DT21:   .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL.DS,TRK.DS
8284  047260  001254  001270  001274
8285  047266  001164  001126  001172  DT21A:  .WORD   $REG1,$BDDAT,$REG4,$REG1
8286  047274  001164
8287  047276  001176  001116  001254  DT23:   .WORD   $TMPO,$ERRPC,CHKDRV,CYL.DS,RP.REG,RP.REG+10,RP.REG+12
8288  047304  001270  004204  004214
8289  047312  004216
8290  047314  004220  004244  004246  DT23A:  .WORD   RP.REG+14,RP.REG+40,RP.REG+42,RP.REG+34,RP.REG 36
8291  047322  004240  004242
8292  047326  001176  001116  001162  DT41:   .WORD   $TMPO,$ERRPC,$REGO,CHKDRV
8293  047334  001254
8294  047336  001176  001116  001162  DT42:   .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,RP.REG,RP.REG+10,RP.REG+12
8295  047344  001254  004204  004214
8296  047352  004216
8297  047354  001176  001116  001162  DT43:   .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,RP.REG,RP.REG+10,RP.REG+12
8298  047362  001254  004204  004214
8299  047370  004216
8300  047372  004220  004244  004246  DT43A:  .WORD   RP.REG+14,RP.REG+40,RP.REG+42
8301  047400  001176  001116  001162  DT44:   .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
8302  047406  001254  001270  001274
8303  047414  001272
8304  047416  004204  004214  004216  DT44A:  .WORD   RP.REG,RP.REG+10,RP.REG+12,RP.REG+36,RP.REG+34,RP.REG+06
8305  047424  004242  004240  004212
8306  047432  004220  004244  004246  DT44B:  .WORD   RP.REG+14,RP.REG+40,RP.REG+42
8307  047440  001176  001116  001162  DT45:   .WORD   $TMPO,$ERRPC,$REGO,CHKDRV,CYL.DS,TRK.DS,SEC.DS
8308  047446  001254  001270  001274
8309  047454  001272
8310  047456  004204  004214  004216  DT45A:  .WORD   RP.REG,RP.REG+10,RP.REG+12,RP.REG+36,RP.REG+34,RP.REG+06
8311  047464  004242  004240  004212
8312  047472  004220  004244  004246  DT45B:  .WORD   RP.REG+14,RP.REG+40,RP.REG+42,RP.REG+2,RP.REG+4,RP.REG+22
8313  047500  004206  004210  004226
8314
8315                                  .SBTTL  DATA FORMAT (DF) TABLE
8316
8317  047506  000001                  DF1:     WORD   1                ;NUMBER OF DATA HEADERS
8318  047510     002                           .BYTE   2                ;NUMBER OF WORDS IN DATA TABLE
8319  047511     000                           .BYTE   0                ;ALL 3 NUMBERS ARE OCTAL
8320
8321  047512  000001                  DF2:    .WORD   1
8322  047514     007                          .BYTE   7
8323  047515     000                          .BYTE   0
```

```
8324
8325  047516  000001              DF3:    .WORD   1
8326  047520     004                      .BYTE   4
8327  047521     000                      .BYTE   0
8328
8329  047522  000001              DF4:    .WORD   1
8330  047524     005                      .BYTE   5
8331  047525     000                      .BYTE   0
8332
8333  047526  000001              DF10:   .WORD   1
8334  047530     002                      .BYTE   2
8335  047531     000                      .BYTE   0
8336
8337  047532  000001              DF11:   .WORD   1
8338  047534     002                      .BYTE   2
8339  047535     000                      .BYTE   0
8340
8341  047536  000002              DF12:   .WORD   2               ;2 DH'S TO BE TYPED
8342  047540     007                      .BYTE   7               ;7 DATA WORDS FOLLOW THE 1ST DH
8343  047541     160                      .BYTE   160             ;WORDS 1-4 ARE OCTAL 5-7 ARE DECIMAL
8344  047542  046027                      .WORD   DH12A           ;ADDRESS OF 2ND DH
8345  047544     006                      .BYTE   6               ;6 DATA WORDS FOLLOW THE 2ND DH
8346  047545     000                      .BYTE   0               ;ALL WORDS ARE OCTAL
8347
8348  047546  000002              DF13:   .WORD   2
8349  047550     007                      .BYTE   7
8350  047551     160                      .BYTE   160
8351  047552  046106                      .WORD   DH13A
8352  047554     005                      .BYTE   5
8353  047555     004                      .BYTE   4               ;WORD 3 IS DECIMAL
8354
8355  047556  000000              DF14:   .WORD   0
8356  047560     005                      .BYTE   5
8357  047561     004                      .BYTE   4               ;WORD 3 IS DECIMAL
8358
8359  047562  000001              DF17:   .WORD   1
8360  047564     010                      .BYTE   10B
8361  047565     000                      .BYTE   0
8362
8363  047566  000002              DF21:   .WORD   2
8364  047570     006                      .BYTE   6
8365  047571     060                      .BYTE   60
8366  047572  046330                      .WORD   DH21A
8367  047574     004                      .BYTE   4
8368  047575     014                      .BYTE   14
8369
8370  047576  000000              DF22:   .WORD   0
8371  047600     004                      .BYTE   4
8372  047601     014                      .BYTE   14
8373
8374  047602  000002              DF23:   .WORD   2
8375  047604     007                      .BYTE   7
8376  047605     010                      .BYTE   10              ;WORD 4 IS DECIMAL
8377  047606  046455                      .WORD   DH23A
8378  047610     005                      .BYTE   5
8379  047611     000                      .BYTE   0
```

```
8380
8381
8382  047612  000001                  DF41:   .WORD   1
8383  047614     004                          .BYTE   4
8384  047615     000                          .BYTE   0
8385
8386  047616  000001                  DF42:   .WORD   1
8387  047620     007                          .BYTE   7
8388  047621     000                          .BYTE   0
8389
8390  047622  000002                  DF43:   .WORD   2
8391  047624     007                          .BYTE   7
8392  047625     000                          .BYTE   0
8393  047626  046645                          .WORD   DH43A
8394  047630     003                          .BYTE   3
8395  047631     000                          .BYTE   0
8396
8397  047632  000003                  DF44:   .WORD   3
8398  047634     007                          .BYTE   7
8399  047635     160                          .BYTE   160
8400  047636  046674                          .WORD   DH44A
8401  047640     006                          .BYTE   6
8402  047641     000                          .BYTE   0
8403  047642  046751                          .WORD   DH44B
8404  047644     003                          .BYTE   3
8405  047645     000                          .BYTE   0
8406
8407  047646  000003                  DF45:   .WORD   3
8408  047650     007                          .BYTE   7
8409  047651     160                          .BYTE   160
8410  047652  046674                          .WORD   DH44A
8411  047654     006                          .BYTE   6
8412  047655     000                          .BYTE   0
8413  047656  046777                          .WORD   DH45A
8414  047660     006                          .BYTE   6
8415  047661     000                          .BYTE   0
8416
8417                                  .EVEN
8418          047662                  BUFFER=.
8419
8420  047662  005015  046412  044501  TITLE:  .ASCII  <CR><LF><LF>/MAINDEC-11-DZRJA-A/<CR><LF>
8421  047670  042116  041505  030455
8422  047676  026461  055104  045122
8423  047704  026501  006501     012
8424  047711     122  030120  027464          .ASCIZ  #RP04/5/6 MECHANICAL & READ-WRITE TEST#<CR><LF><LF>
8425  047716  027465  020066  042515
8426  047724  044103  047101  041511
8427  047732  046101  023040  051040
8428  047740  040505  026504  051127
8429  047746  052111  020105  042524
8430  047754  052123  005015  000012
8431  047762  005015  047524  052040  LOADRV: .ASCII  <CR><LF>/TO TEST DRIVE 0 REPLACE THE 'XXDP' PACK ON DRIVE 0/<CR><LF>
8432  047770  051505  020124  051104
8433  047776  053111  020105  020060
8434  050004  042522  046123  041501
8435  050012  020105  044124  020105
```

```
8436  050020  054047  042130  023520
8437  050026  053040  041501  020113
8438  050034  047117  042040  044522
8439  050042  042526  030040  005015
8440  050050  044527  044124  040440          .ASCII  /WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40, AND RESTART/<CR><LF>
8441  050056  047516  044124  051105
8442  050064  030040  041501  026113
8443  050072  041440  042514  051101
8444  050100  046440  046505  051117
8445  050106  020131  047514  040503
8446  050114  044524  047117  032040
8447  050122  026060  040440  042116
8448  050130  051040  051505  040524
8449  050136  052122  005015
8450  050142  044124  020105  051120          .ASCIZ  /THE PROGRAM/<CR><LF>
8451  050150  043517  040522  020515
8452  050156  000012
8453  050160  005015  054523  052123  NOLOAD: .ASCIZ  <CR><LF>/SYSTEM HAS 16K MEMORY, 'XXDP' LOADER WILL BE OVERWRITTEN/<CR><L
8454  050166  046505  044040  051501
8455  050174  030440  045466  046440
8456  050202  046505  051117  026131
8457  050210  023440  054130  050104
8458  050216  020047  047514  042101
8459  050224  051105  053440  046111
8460  050232  020114  042502  047440
8461  050240  042526  053522  044522
8462  050246  052124  047105  020515
8463  050254  000012
8464
8465                                  .EVEN
8466
8467                                  .SBTTL  ROUTINE TO SIZE MEMORY
8468
8469                                  ;;*****************************************************************
8470                                  ;#CALL:
8471                                  ;#      JSR     PC,$SIZE
8472                                  ;#      RETURN
8473                                  ;#$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
8474
8475  050256  010046                  $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
8476  050260  010146                          MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
8477  050262  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
8478  050266  013746  000006                  MOV     @#ERRVEC+2,-(SP)
8479  050272  010600                          MOV     SP,R0           ;;SAVE THE STACK POINTER
8480                                  ;;SET THE ERRVEC PS TO THE PRESENT PS
8481  050274  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
8482  050276  012637  000006                  MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
8483  050302  012737  050322  000004          MOV     #2$,@#ERRVEC    ;;SET FOR TIMEOUT
8484  050310  012701  020000                  MOV     #20000,R1       ;;FIRST ADDRESS
8485  050314  005711                  1$:     TST     (R1)            ;;TEST THIS ADDRESS
8486  050316  005721                          TST     (R1)+           ;;STEP TO NEXT ADDRESS
8487  050320  000775                          BR      1$              ;;TRY ANOTHER
8488  050322  162701  000002          2$:     SUB     #2,R1           ;;DROP BACK
8489  050326  010006                          MOV     R0,SP           ;;RESTORE THE STACK
8490  050330  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
8491  050334  012637  000004                  MOV     (SP)+,@#ERRVEC
```

```
                                ROUTINE TO SIZE MEMORY

 8492  050340  010137  050352           MOV     R1,$LSTAD       ;;LAST ADDRESS
 8493  050344  012601                   MOV     (SP)+,R1        ;;RESTORE R1
 8494  050346  012600                   MOV     (SP)+,R0        ;;RESTORE R0
 8495  050350  000207                   RTS     PC
 8496  050352  000000           $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
 8497
 8498                           ;;*****************************************************************
 8499                           .SBTTL  GETADR - GET BUS ADDRESS AND VECTOR ADDRESS
 8500                           ;THIS ROUTINE IS USED TO ENSURE THE BUS ADDRESS
 8501                           ;OF THE RH11/RP04 IS SETUP TO READ THE PROPER VALUE.
 8502                           ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
 8503                           ;REQUIRED.
 8504                           ;NOTE: THIS ROUTINE DESTROYS R0-R4
 8505                           ;CALL
 8506                           ;
 8507                           ;       JSR     PC,@#GETADR
 8508                           ;       RETURN
 8509
 8510  050354  005737  001226   GETADR: TST     @#BUSADR        ;INPUT FROM TTY REQUESTED?
 8511  050360  001427                   BEQ     7$              ;NO--BRANCH
 8512  050362  005037  001226           CLR     @#BUSADR        ;YES--CLEAR THE REQUEST FLAG
 8513  050366  012700  001366   1$:     MOV     #RH.AUR,R0      ;FIRST ADDRESS
 8514  050372  012703  050532           MOV     #MRPCS1,R3      ;"RPCS1="
 8515  050376  011004                   MOV     (R0),R4         ;PRESENT RPCS1 ADDRESS
 8516  050400  004037  031466           JSR     R0,@#GETNUM     ;GET NEW RPCS1
 8517  050404  000402                   BR      2$              ;COMMA
 8518  050406  000767                   BR      1$              ;PERIOD
 8519  050410  000412                   BR      5$              ;DOUBLE PERIOD
 8520  050412  010420           2$:     MOV     R4,(R0)+        ;SAVE NEW RPCS1
 8521  050414  012703  050543           MOV     #MRHVEC,R3      ;"RHVEC="
 8522  050420  011004                   MOV     (R0),R4         ;PRESENT RH11 VECTOR ADDRESS
 8523  050422  004037  031466           JSR     R0,@#GETNUM     ;GET NEW RHVEC
 8524  050426  000402                   BR      3$              ;COMMA
 8525  050430  000756                   BR      1$              ;PERIOD
 8526  050432  000401                   BR      5$              ;DOUBLE PERIOD
 8527  050434  010420           3$:     MOV     R4,(R0)+        ;SAVE NEW RHVEC
 8528  050436  010410           5$:     MOV     R4,(R0)         ;SAVE INPUT
 8529  050440  013701  000004   7$:     MOV     @#ERRVEC,R1     ;SAVE THE ERROR VECTOR
 8530  050444  012737  050500  000004   MOV     #8$,@#ERRVEC    ;SETUP FOR TRAP
 8531  050452  005777  130710           TST     @RH.ACR         ;CHECK FOR RH11/RP04
 8532  050456  010137  000004           MOV     R1,@#ERRVEC     ;RESTORE ERROR VECTOR
 8533  050462  012700  001366           MOV     #RH.AOR,R0      ;FIRST ADDRESS OF NEW PARAMETERS
 8534  050466  012701  034464           MOV     #RPAOR,R1       ;FIRST ADDRESS OF WHERE TO PUT THEM
 8535  050472  012021                   MOV     (R0)+,(R1)+     ;BUS ADDRESS
 8536  050474  012021                   MOV     (R0)+,(R1)+     ;VECTOR ADDRESS
 8537  050476  000207                   RTS     PC              ;RETURN
 8538  050500  010137  000004   8$:     MOV     R1,@#ERRVEC     ;RESTORE ERROR VECTOR
 8539  050504  022626                   CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
 8540  050506  104010                   ERROR   10              ;REPORT THE ERROR
 8541  050510  005737  000042           TST     @#42            ;IS THERE A MONITOR?
 8542  050514  001724                   BEQ     1$              ;NO--GO ASK FOR ADDRESS
 8543  050516  005037  001232           CLR     @#DRVSEL        ;YES--NO DRIVES SELECTED
 8544  050522  005037  017532           CLR     @#SEOPCT        ;NO PASSES
 8545  050526  000137  017356           JMP     @#SEOP          ;GO TO END OF PROGRAM
 8546
 8547  050532  005015  050122  051503 MRPCS1: .ASCIZ  <CR><LF>/RPCS1=/
```

```
8548  050540  036461     000
8549  050543     015  051012  053110  MRHVEC: .ASCIZ  <CR><LF>/RHVEC=/
8550  050550  041505  000075
8551          000001                  .END
```

F13


```
ABS   = 000200          384#
ACL   = 000040          418#    427#
ACTDRV  034402         6529#   6807#   6858#   7171#   7180#   7408
ACTSTR  034403         6535#   7410#   7425#
ACU   = 100000          373#
AOE   = 001000          290#
ATA   = 100000          277#
ATABIT  034452         1861    5875    5881    6601#   6774    6870    6975    7334    7353    7361    7378    7484
AT0   = 000001          310#
AT1   = 000002          311#
AT2   = 000004          312#
AT3   = 000010          313#
AT4   = 000020          314#
AT5   = 000040          315#
AT6   = 000100          316#
AT7   = 000200          317#
A16   = 000400          209#
A17   = 001000          205#
BAI   = 000010          226#
BITS    001424          591#    2022    2056    2094    2137    2177    2257    2300    2350    2450    2607    2673    2777
                       2867    2965    3079    3150    3279    3325    3399    3488    5942    5973    5990    6043    6217
BIT0  = 000001          183#
BIT00 = 000001          173#     183     591     607    1832    1849    1871    1924    4689
BIT01 = 000002          172#     182     592     608    1835    5015    5043    6850    7097
BIT02 = 000004          171#     181     593     609    4752    5015    5046
BIT03 = 000010          170#     180     594     610    5052    7293    7606
BIT04 = 000020          167#     179     595     611    5049    7328
BIT05 = 000040          168#     178     596     612    2493    2509    2585    2715    2729    2744    2749    2814    2838
                       2843    2902    2920    2937    2942    3000    3018    3035    3040    5052    5162    6691    7117
                       7308    7458
BIT06 = 000100          167#     177     597     613    5055    6783    6841    6915    7206    7667
BIT07 = 000200          166#     176     598     614    6783    7049    7192    7293    7308    7328    7357    7639
BIT08 = 000400          165#     175     599    5043    6793    7471
BIT09 = 001000          164#     174     600    3615    4237    5058    7456
BIT1    000002          182#
BIT10 = 002000          163#     601    3599    4701    5015    5046    6236    7099
BIT11 = 004000          162#     602    4244    5046    6738    6892    7068    7379
BIT12 = 010000          161#     603    5015    5049    6733    6765    6783    6852    6874    6886    7064    7083    7095
                       7108    7318    7320    7344    7601    7668
BIT13 = 020000          160#     604    3606    5015    5043    6835    7266    7547
BIT14 = 040000          159#     605    2504    2710    2724    2809    2897    2915    2995    3013    4219    4943    5015
                       5049    6847    6883    7253    7395    7511    7556    7560    7665
BIT15 = 100000          158#     606    5207    5208    6835    6847    6850    6852    6883    6886    7068    7097    7099
                       7206    7293    7308    7318    7328    7395    7456    7471    7511    7518
BIT2  = 000004          181#    7518
BIT3  = 000010          180#
BIT4  = 000020          179#
BIT5  = 000040          178#
BIT6  = 000100          177#    6876
BIT7  = 000200          176#
BIT8  = 000400          175#
BIT9  = 001000          174#
BPTVEC= 000014          190#
BUFFER= 047662         1248    1272    1296    1320    2470    3097    3112    3125    3168    3178    3192    3313    3415
                       3439    3447    3455    3461    3469    5126#   5127    5129    5131    5132    5133    5206    5243
                       5386    5405    5406    5430    5519    5551    5594    5636    5653    5654    5671    8418#
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
BUSADR  001226          528#  1709*  1711*  1714*  1716*  8510  8512*
BYPASS  001252          537#  1922*  2036*  2072*  2107*  2150* 2190*  2270*  2313*  2364*  2477*  2624*  3089*
                       3160*  3314*  3410*  4997   5477   5623  5688
CALL.A  025002         1928   1930   2039   2383   3502   3507  4795#  4798   5141
CALL.B  025114         2040   2076   2112   2120   2154   2159  2200   2206   2222   2228   2276   2278   2322
                       2324   2326   2328   2330   2332   2405  2628   4826#  4829
CALL.C  025304         2075   2627   4869#  4872
CHKDRV  001254          538#  1901*  1906*  1911   3533   4740  8270   8275   8280   8283   8287   8292   8294
                       8297   8301   8307
CI1     036024         6905   6933#
CI3     036132         6907   6956#
CI4     036240         6896   6977#
CI5     036616         6955   6976   7054#  7063
CI6     036640         6993   7003   7005   7007   7059#
CI7     036654         6837   6914   6945   6949   6953  6961   6970   6974   6985   6992   6998   7002   7016
                       7022   7026   7036   7047   7062  7064#  7166   7196   7345   7651
CI7B    036702         7067   7070#  7347
CI8     036754         7065   7073   7082#  7219   7503
CKSCTR  027760         3111   5427#
CKSWR = 104407         3589   3614   4218   4346#
CK.CHR  033136         5809   5859   6086   6123   6163  6289#  6326   6339   6350   6401
CK.DEC  033710         6267#  6297
CK.DIG  034012         5869   6107   6320#
CK.NUM  034176         5818   6139   6172   6384#
CK.OCT  033662         5922   5926   5946   5950   6249# 6386   6396
CLKSTA  001244          533#  1893   2482   2512   2683  2787   2877   2975   4619*  4621*  4627*
CLOSE   033602         6207   6211   6220#
CLR   = 000040          228#
CLRBUF  027712         3108   5403#
CLRQUE  042466         2476   6673   7116   7481   7735#
CLSMOS  033502         6191   6195   6198#
CNTRLC  001224          527#  1712*  1717*  1801   1645
COMMA   043237         1889   7883#
COUNT   027070         2733   2818   2907   2925   3005  3023   5221#
CR    = 000015           98#   523#  3526   3557   3791  3801   5776  5956   7852   7858   7877   7884   7894
                       7902   7907   7914   7917   7924  7927   7933  7937   7939   7941   8420   8424   8431
                       8440   8450   8453   8547   8549
CRLF  = 000200           99#  3762   3801
CSF   = 000002          360#   378#
CSU   = 000010          362#   380#
CYL.DS  001270          545#  2496*  4803*  4834*  4877*  4926*  4956*  4982*  5136*  5174*  5493*  5607*  8270
                       8273   8275   8283   8287   8301   8307
CYL.RD  001262          542#  5131*  8273
C.SWR   001220          525#  1848   1851   1866   1924   2192  2697   2753   3417   4634   4644   4654   4689
                       4779   4857   4900   4998   5082   5278  5782   5783*
DATCMP  030420         3365   5550#
DCK   = 100000          296#
DCL   = 000100          419#   428#
DCU   = 000001          425#
DDISP = 177570          100#   496   1745
DECSEC  027630         5371#
DECSK   007056         2119#  2123
DELTA   001350          566#  2468*  2558*  2559*  2560  2562*  2571
DE1   = 000040          267#
DFF20 = 000002          263#
DFLT    001664          683#  4678
```

```
DF1     047506          1409    8317#
DF10    047526          1473    8333#
DF11    047532          1483    8337#
DF12    047536          1496    8341#
DF13    047546          1509    1530    8348#
DF14    047556          1517    1538    8355#
DF17    047562          1548    1558    8359#
DF2     047512          1419    1449    8321#
DF21    047566          1571    8363#
DF22    047576          1579    8370#
DF23    047602          1591    1603    8374#
DF3     047516          1429    8325#
DF4     047522          1439    8329#
DF41    047612          1636    8382#
DF42    047616          1646    8386#
DF43    047622          1658    8390#
DF44    047632          1673    8397#
DF45    047646          1688    1703    8407#
DH1     045473          1407    8116#
DH10    045706          1471    8140#
DH11    045721          1481    8143#
DH12    045740          1494    1507    1528    1671    1686    1701    8146#
DH12A   046027          8156#   8344
DH13A   046106          8164#   8351
DH17    046154          1546    1556    8171#
DH2     045510          1417    1447    8119#
DH21    046252          1569    8182#
DH21A   046330          8190#   8366
DH23    046367          1589    1601    8196#
DH23A   046455          8206#   8377
DH3     045576          1427    8129#
DH4     045633          1437    8136#
DH41    046522          1634    8213#
DH42    046560          1644    1656    8218#
DH43A   046646          8227#   8393
DH44A   046674          8231#   8400    8410
DH44B   046751          8239#   8403
DH45A   046777          8243#   8413
DIGB  = 000004          264#
DISPLA  001142          436#    1745#   1753#   2029#   2063#   2101#   2144#   2184#   2264#   2307#   2357#   2457#   2614#
                        2680#   2784#   2874#   2972#   3086#   3157#   3286#   3406#   3495#   3598#   4256#
DISPRE  000174          63#     1753
DLT   = 100000          238#
DL64  = 000020          266#
DM0   = 000001          300#
DORTI   027022          2491    2704    2802    2890    2988    5193#
DP8.A   004104          1243#   1917#   1923#   1926#   1927#   1929#   2032#   2361#   2379#   2386    2462#   2465#   2474
                        2495    2496    2506    2567#   2574#   2576    2578    2582    3500#   3501#   3505#   3506#   4797
                        4799    4803    4804    4805    4806    4814    5140#
DP8.B   004124          1267#   1918#   2033#   2034#   2035#   2066#   2068#   2070#   2105#   2106#   2108#   2113#   2114
                        2116#   2121#   2122    2148#   2149#   2151#   2158#   2188#   2189#   2199#   2205#   2221#   2227#
                        2268#   2269#   2275#   2277#   2311#   2312#   2321#   2323#   2325#   2327#   2329#   2331#   2360#
                        2362#   2367#   2377#   2379    2406#   2407    2409#   2618#   2620#   2622#   2633#   2634    4781#
                        4784#   4828    4830    4834    4835    4836    4837    4849    4852    4855
DP8.C   004144          1291#   1919#   2067#   2069#   2071#   2619#   2621#   2623#   2629#   2630    2632#   4782#   4785#
                        4871    4873    4877    4878    4879    4880    4892    4895    4898
```

```
DPINT   034356          6489#  6702   6705   6791*  6817   7237   7262   7385   7387*  7448   7486   7497   7505#
DPR   = 000400           270#
DPRQS   034366          6502#  6823   6909*  7240   7399*  7450   7488   7499   7514*
DRIVES  043201          1872   7877#
DRQ   = 004000           333#
DRVACT  034326          2545*  6446#  6831   7010*  7057*  7076*  7087   7102*  7185*  7258   7291   7322*  7329*
                        7342   7352*  7459*  7465   7472*
DRVCAL  025474          3101   3105   3110   3120   3127   3173   3184   3200   3206   3346   3352   3362   3424
                        3444   3450   3458   3464   3472   4915#  4919
DRVCLR= 000111           446#  2538
DRVCL1  025514          2551   4920#  4921
DRVINT  034714          6693   6727#  6819   6878   7369   7388   7463
DRVMSK  001256           539#  1902*  1903   1907*  1915
DRVOK   006112          1904   1911#
DRVQUE  042564          6826   6839   7775#
DRVSEL  001232           530#  1856*  1861*  1871*  1875   1898   1903   1915*  3527   5852*  5875*  5881*  8543*
DRVSTA  034336          1815   1859   1912   6460#  6683*  6684*  6685*  6686*  6696*  6728*  6737*  6780*  6785*
                        6813   6821   6845   6880   6884   7011*  7120*  7243   7251   7264   7316*  7323*  7335
                        7390   7506*
DRVTYP  034346          1818   1832   1835   4752   6476#  6729*  6744*  6749*  6754*  6759*  6848   7121*
DRY   = 000200           269#
DSWR  = 177570           104#   495    1744
DTADPB  ^^4164          1315#  1920*  2395*  2398   2401*  2403*  2463*  2464*  2466*  2469*  2470*  2528*  2531
                        2534*  2536*  2537   2542*  2543   2549   2576   2578*  2713   2716   2727   2730   2742
                        2745   2750   2797*  2798*  2804   2812   2815   2822*  2823   2827*  2828   2836   2839
                        2844   2900   2903   2918   2921   2935   2938   2943   2998   3001   3016   3019   3033
                        3036   3041   3093*  3094*  3095*  3096*  3097*  3100*  3102*  3109*  3116*  3117*  3118*
                        3119*  3123*  3124*  3125*  3126*  3164*  3165*  3166*  3167*  3168*  3172*  3174*  3178*
                        3182*  3183*  3185*  3190*  3192*  3195*  3199*  3201*  3202*  3205*  3207   3312*  3313*
                        3328*  3329*  3330*  3335*  3345*  3351*  3361*  3413*  3414*  3415*  3416*  3420*  3425*
                        3426   3428*  3429   3431*  3436   3437   3438*  3440*  3446*  3447*  3452*  3453*  3454*
                        3455*  3460*  3461*  3466*  3467*  3468*  3469*  4918   4920   4926   4927   4928   4929
                        4939   4941   4947*  4950   4952   4956   4957   4958   4959   4967*  4969   4971   4975
                        4977   4982   4983   4984   4985   4993   5162*  5164*  5165*  5168   5170   5174   5175
                        5176   5177   5493   5494   5520   5552   5607   5608   5609   5758*  5759*  5760*
DTE   = 010000           293#
DTSY  = 000200           306#
DTUM    034450          2544*  6595#  6679   6838   6954*  7077   7080*  7090   7103   7105*  7114*  7174   7186*
                        7453   7468   7479*  7490   7554
DT00  = 000001           324#
DT01  = 000002           325#
DT02  = 000004           326#
DT03  = 000010           327#
DT04  = 000020           328#
DT05  = 000040           329#
DT06  = 000100           330#
DT07  = 000200           331#
DT08  = 000400           332#
DT1     047054          1408   8257#
DT10    047140          1472   8268#
DT11    047144          1482   8269#
DT12    047150          1495   8270#
DT12A   047166          8273#
DT13    047202          1508   1529   8275#
DT13A   047220          1516   1537   8278#
DT17    047232          1547   1557   8280#
```

```
DT2     047060          1418    1448    8258#
DT21    047252          1570    8283#
DT21A   047266          1578    8285#
DT23    047276          1590    1602    8287#
DT23A   047314          8290#
DT3     047076          1428    8261#
DT4     047106          1438    8263#
DT41    047326          1635    8292#
DT42    047336          1645    8294#
DT43    047354          1657    8297#
DT43A   047372          8300#
DT44    047400          1672    8301#
DT44A   047416          8304#
DT44B   047432          8306#
DT45    047440          1687    1702    8307#
DT45A   047456          8310#
DT45B   047472          8312#
D.S     047120          8265#
DVA   = 004000           258#
ECH   = 000100           287#
ECI   = 004000           401#
EMPTYQ  042544          7074    7119    7207    7330    7397    7519    7760#
EMTVEC= 000030           193#    1729*   1730*
EM11    044013          1406    7961#
EM10    044241          1470    7987#
EM11    044317          1480    7995#
EM12    044354          1493    8000#
EM13    044401          1506    1527    1568    8004#
EM17    044426          1545    8008#
EM2     044056          1416    7967#
EM20    044460          1555    8013#
EM23    044527          1588    8020#
EM24    044556          1600    8024#
EM3     044114          1426    7972#
EM4     044151          1436    7977#
EM41    044616          1633    1643    1655    1670    1685    8030#
EM46    044642          1700    8034#
EM5     044205          1446    7982#
ERINDX  026216          4807    4838    4881    4930    4960    4986    5037#   5178
ERR   = 043000           276#
ERRVEC= 000004           185#    1742    1743*   1754*   4224    4225*   4227*   4230*   4588*   4589*   4594*   4614    8477
                        8478    8482*   8483*   8490*   8491*   8529    8530*   8532*   8538*
ERR.CT  001364           572#    1956    3088    3159    3288    3408    4995    5475    5506    5621    5680
ES.SAV  042724          7222    7273    7277    7552    7612    7829#
EXIT0   006504          2036    2041#
EXIT1   006662          2072    2077#
EXIT10  012012          2477    2557    2588    2590#
EXIT11  012232          2624    2636#
EXIT12  012770          2689    2694    2757    2761#
EXIT13  013434          2793    2795    2849#
EXIT14  014152          2883    2885    2948#
EXIT15  014670          2981    2983    3046#
EXIT16  015266          3089    3132#
EXIT17  015706          3160    3212#
EXIT2   007100          2107    2124#
EXIT20  016426          3314    3323    3369#
```

```
EXIT21  017204            3410    3473#
EXIT22  017354            3510#
EXIT3   007274            2150    2163#
EXIT4   007662            2190    2240#
EXIT5   010062            2270    2282    2285#
EXIT6   010326            2313    2320    2335#
EXIT7   010706            2364    2408    2410#
EXT1    = 000001           340#
EXT10   = 000010           343#
EXT2    = 000002           341#
EXT20   = 000020           344#
EXT4    = 000004           342#
EXT40   = 000040           345#
FC      001510             620#   2070    2108    2122    2151    2195    2205    2238    2271    2283    2314    2367    2368
                          2375    2465    2466    2467    2564    2622    2623    2632    2690    2797    2910    3008    3093
                          3165    3317    3413    3505    5735    5753
FEN     = 000200           366#
FER     = 000020           285#
FILBUF  027654            3092    3163    5384#
FILRAN  030736            3419    5636#
FMT22   = 010000            402#    466#   1923    5126
FS      001524             626#   2033    2066    2105    2148    2188    2268    2311    2618    2619    2691    3166    3293
                          3307    3312
FT      001516             623#   2034    2068    2106    2149    2189    2269    2312    2360    2409    2464    2620    2621
                          2692    3094    3327
F1      = 000002           253#
F2      = 000004           254#
F3      = 000010           255#
F4      = 000020           256#
F5      = 000040           257#
GETADR  050354            1790    8510#
GETNUM  031466            5779    5801#   8516    8523
GETREG= 000141             459#
GETREQ  042640            6871    7092    7247    7283    7344    7394    7455    7470    7508    7515    7796#
GETSWR  031374            1795    5771#
GNS     = ****** U          62    3525    3531    3539    3556    3668    3685    4338    4339    4340    4341    4342    4344
                          4346    4347    4348    4349    4350    5775    5855    5888    6047    6153
GO      = 000001           252#
GRV     = 000010           265#
GTSWR   = 104406          1784    4344#
GTTST1  031762            5880    5885#   5934    5947    5958    5961    5987    5994    6003    6009    6018    6035
GTTST2  032046            5902#   5938
GTTST3  032410            5905    5909    5913    5917    5921    5980#
GTTST4  032412            5923    5944    5978    5981#
GTTST5  032622            5982    6004#
GTTST6  032562            6006    6015#
GT.PRM  031624            1853    5851#
GT.PR1  031626            5852#   5860    5861    5862    5863    5864    5870    5871
GT.PR2  031756            5882#   5897    6014    6023    6216
HCE     = 000200           288#
HCI     = 002000           400#
HCRC    = 000400           289#
HT      = 000011            96#   3760    3801
IAE     = 002000           291#
IC      001514             622#   2113    2121    2155    2215    2237    2279    2280    2333    2468    2555    2560    2562
                          2629    2633    5358
```

```
IE     = 000100            206#
ILF    = 000001            281#
ILR    = 000002            282#
INCCYL   027600           3322    5356#
INCSK    007014           2111#   2115
INCTRK   027550           3366    5339#
IOTVEC=  000020            191#   1727*  1728*
IR     = 000100            229#
ISR      037346           2511    2752    2846    2945    3043    5160    6688    7171#
IT       001522            625#   5341
ITEMH1   004546           1626#   3652
IXE    = 004000            370#    388#
KIPAR0=  ****** U          8473
LA       037210           6902    7136#
LACNT    034414           6554#   7144*   7145    7162*
LC       001512            621#   2035    2071    2114    2116    2161    2216    2218    2227    2272    2281    2315    2366
                           2373    2556    2558    2564    2630    2634    2823    2892    2990    3331    3429    3500    5356
                           5359    5361    5748
LDCMD    024736           1921    4779#
LF     = 000012             97#    524#    3526    3557    3795    3801    5776    5856    7852    7858    7877    7884    7894
                           7902    7907    7914    7917    7924    7927    7933    7937    7939    7941    8420    8424    8431
                           8440    8450    8453    8547    8549
LKS      001412            584#   4626    4658*
LKV      001406            583#   2488    2489    4656*   4657*
LOADRV   047762           1769    8431#
LOOFLT   024322           1852    1855    4671#
LOOPRM   024560           2027    2061    2099    2142    2182    2262    2305    2355    2455    2612    2678    2782    2872
                           2970    3084    3155    3284    3404    3493    3729#
LOP.CK   026156           4815    4850    4893    4940    4994    5011#
LPB      001422            588#   3595
LPS      001420            587#   3594    4590
LPTAVL   001230            529#   3592    4587*   4591*
LP.AVL   024002           1792    4587#
LS       001526            627#   2067    3292
LST    = 002000            572#
LT       001520            624#   2069    2907    5339    5342    5344
MABOVE   043731           5303    7948#
MBELOW   043702           5287    7944#
MCLK   = 000002            301#
MCPE   = 020000            211#
MCPEMX   034462           6613#   7534    7582
MHS    = 001000            368#    386#
MIMX   = 000004            302#
MNOLTA   034476           6629#   7160
MOH    = 020000            334#
MOL    = 010000            274#
MPE    = 000400            231#
MRD    = 000020            304#
MRHVEC   050543           8521    8549#
MRPCSI   050532           8514    8547#
MSE    = 000020            363#
MSGAVG   043666           5308    7941#
MSGB01   045435           1377    8107#
MSGB02   045365           1376    8100#
MSGB03   045315           1375    8093#
MSGB04   045275           1374    8090#
```

```
MSGBOS  045225          1373    8093#
MSGBO6  045161          1372    8076#
MSGBO8  045117          1371    8070#
MSGBO9  045060          1370    8064#
MSGB10  045035          1369    8060#
MSGB11  045007          1368    8056#
MSGB12  044765          1367    8052#
MSGB13  044734          1366    8047#
MSGB14  044672          1365    8041#
MSGFCP  042760           672    7842#
MSGLCP  042764           673    7843#
MSGMAX  043657          5292    7939#
MSGMIN  043650          5276    7937#
MSGNON  043775          5329    7955#
MSGNUM  043760          5324    7952#
MSG.CS  043020          5777    7852#
MSG.EQ  043016          6079    6117    6158    7851#
MSG.FC  042752           670    7840#
MSG.FS  043004           678    7848#
MSG.FT  042773           675    7845#
MSG.IC  042770           674    7844#
MSG.IT  043001           677    7847#
MSG.LC  042755           671    7841#
MSG.LS  043007           679    7849#
MSG.LT  042776           676    7846#
MSG.PA  043012          6116    7850#
MSG.R   042750           669    7839#
MSG.SP  044010          1813    1814    1830    1936    3680    3712    3719    3726    5283    5299    5320    6075    6115
                        6120    7957#
MSGOUS  043675          5280    5291    5296    5307    5319    7943#
MSG10A  043422           935    7907#
MSG10B  043471           936    7914#
MSG11A  043506           940    7917#
MSG11B  043554           941    7924#
MSG12A  043571           945    7927#
MSG12B  043633           946    7933#
MSG7    043370           925     930    7902#
MSTCK = 000010           303#
MWR   = 000040           305#
MXDLTA  034474          6626#   7158
MXF   = 001000           232#
MXLACT  034472          6623#   7145
MXSTAL  001362           571#   2212    2234
MXWNOW  034560          6632#   6963
NBA   = 100000           336#
NC1     001532           629#   2467*   2555*   2556    2559    2567
NC2     001534           630#
NED   = 010000           235#
NEM   = 004000           234#
NHS   = 002000           369#    387#
NOCLOK  043241          1897    7834#
NOLOAD  050160          1776    8453#
NONE    043232          1877    7882#
NOOP  = 000101           442#
NOTPRS  043117          1823    7866#
NOTRP   042144          1821    7871#
```

```
NOTSAF  [illegible]          1827    7869#
OCYL    = 100000              421#    431#
OFFSET= 000115                448#
OFREV   = 000200              399#
OF100   = 000004              395#
OF200   = 000010              396#
OF25    = 000001              393#
OF400   = 000020              397#
OF50    = 000002              394#
OF800   = 000040              398#
OPE     = 020000              429#
OPI     = 020000              294#
OPNFLG  001240                532#  5900*  5901*  5990*  5999   6001   6010   6012   6019   6021   6043
OPNPAT  033170              6074    6115#  6124   6133   6136   6140
OPNFRM  032766              6061    6066#  6070   6125   6127   6146
OPNTST  032614              6000    6002   6011   6013   6020   6022   6027#
OPNWDS  033312              6148#   6182
OPN.CT  032620              6029#   6031*  6032   6036   6038   6049
OPN.N1  033524              6111    6142   6205#
OPN.N2  033530              6098    6131   6193   6206#
OPN.X1  033542              6112    6143   6209#
OPN.X2  033546              6103    6135   6197   6210#
OPN.1   032624              6031#   6044   6208
OPN.2   032646              6030    6033   6036#
OPT     035544              6833    6868#  7202   7362   7400
OR      = 000200              230#
PACK    = 000123              451#
PAR     = 000010              284#
PAT     001530                628#  3325
PAT.PT  003044                950#  5521   5554   5597
PAT0    003104                950    969#  4685   6149   6198
PAT1    003144                951    986#
PAT10   003604                960   1139#
PAT11   003644                961   1156#
PAT12   003704                962   1173#
PAT13   003744                963   1190#
PAT14   004004                964   1207#
PAT15   004044                965   1224#
PAT2    003204                952   1003#
PAT3    003244                953   1020#
PAT4    003304                954   1037#
PAT5    003344                955   1054#
PAT6    003404                956   1071#
PAT7    003444                957   1088#
PAT8    003504                958   1105#  4684
PAT9    003544                959   1122#  4687
PGE     = 002000              233#
PGM     = 001000              271#
PIP     = 020000              275#
PIRQ    = 17  72              103#
PIRQVE= 000240                197#
PKB     001402                561#  2705*  2719*  2803*  2891*  2909*  2989*  3007*  4648*  5110*
PKC     001404                582#  5226   5228   5229   5233   5235   5236   5240   5245
PKCS    001400                [illegible]#  2706*  2709*  2721*  2723*  2737*  2806*  2808*  2831*  2894*  2896*  2912*  2914#
                              [illegible]#  2992*  2994*  3010*  3012*  3028*  4620   4649*  5111*
PKV     001374                573#  2485   2486   2703*  2801*  2823*  2927*  4646*  4647*  5108   5109*  5114#
```

```
PLU     = 020000            372#   389#
POPQUE  042662             6882   6933   7048   7356   7911#
PRM     001504              618#   4738   4742#  4755#  4758#  6057   6066   6221
PRMLMT  001606              655#   2396   2529   3130   3295   3306   4691#  4692#  4695#  4696#  4712   4714   5374
                           5392   5471   5637   5723   6073   6106
PRMMSG  001636              669#   6076
PRMPT   001536              633#   4698   4737   6055   6222
PRM0    002330              633     738#  4679
PRM1    002344              634     746#
PRM10   002544              641     824#
PRM11   002564              642     834#
PRM12   002606              643     845#
PRM13   002622              644     853#
PRM14   002636              645     861#
PRM15   002652              646     869#
PRM16   002666              647     877#
PRM17   002700              648     884#
PRM2    002372              635     759#
PRM20   002712              649     891#
PRM21   002744              650     906#
PRM22   002760              651     914#
PRM3    002414              636     770#
PRM4    002436              637     781#
PRM5    002460              638     792#
PRM6    002502              639     803#
PRM7    002524              640     814#
PR0     = 000000            120#
PR1     = 000040            121#
PR2     = 000100            122#
PR3     = 000140            123#
PR4     = 000200            124#
PR5     = 000240            125#
PR6     = 000300            126#
PR7     = 000340            127#   1803   2481   2503
PS      = 177776            100#    101   1793#  1803#  1844#  2481#  2498#  2503#  2550#  2738#  2832#  2931#  3029#
                           5106   5107#  5115#  5159#  6671   6672#  6701#  6709#  6805   6806#  6859#  6869   6918#
                           7151#
PSEL    = 002000            210#
PSU     = 000001            415#
PSW     = 177776            101#
PTRN15  002742              903#
PWRVEC= 000024              192#
QCNT    042174             7677#  7736   7760#  7775   7777#  7797   7811#
QDRV0   042256             7688   7699   7708   7720#
QDRV1   042306             7689   7700   7709   7721#
QDRV2   042326             7690   7701   7710   7722#
QDRV3   042346             7691   7702   7711   7723#
QDRV4   042366             7692   7703   7712   7724#
QDRV5   042406             7693   7704   7713   7725#
QDRV6   042426             7694   7705   7714   7726#
QDRV7   042446             7695   7706   7715   7727#
QINPT   042204             7688#  7762   7779#  7780#  7781   7783#
QOUTPT  042224             7699#  7762#  7800   7813   7814#  7815   7817#
QSTART  042244             7708#  7742   7747   7783   7817
QSTOP   042246             7709#  7781   7815
QTERM   = 042466           7716   7728#
```

```
RAMADR  031212          3435    3445    3459    5717#
RAMCK   030760          3451    3465    5651#
RAMPAT  031156          3441    5638    5658    5699#
RAW   = 000020           381#
ROCHR = 104410          4148    4347#
ROLIN = 104411          4348#   5807    5857    5890    6084    6121    6161
RDY   = 000200           207#
RD.ADR  041456          7536*   7537*   7539#   8261
RD.RP   041432          6740    6769    6775    6996    7020    7034    7138    7152    7194    7217    7285    7295    7 10
                        7443    7492    7534#   7589    7603    7609    7643
RD.RP1  041454          7538#   7568
RD.RP2  041600          7534*   7567#
RD.RP3  041604          7545    7559    7569#
RD.RP4  041610          7550    7571#
RD.WRD  041460          7540#   7541    8261    8263
READ  = 000171           457#   3109    3361    3446    3460
READHD= 000173           458#   2362    4781    4782    4852    4895
READIN= 000121           450#
RECAL = 000107           445#   1929    2032    5140    5165
RELEAS= 000113           447#
RESREG= 104413          2717    2731    2746    2916    2840    2904    2922    2939    3002    3020    3037    3728    4350#
                        4410    5209    5394    5417    5479    5540    5591    5882    6034    6192    6196    6215    6238
                        6710    6853    6856    6919    7123    7179    7424    7652    7751
RESTAR  006072          1905#   1916    3568
RESVEC= 000010           187#
RHVEC   001370           578#
RH.ADR  001366           577#   A268    8513    8531    8533
RMR   = 000004           283#
RPADR   034464          2363    2475    5161    6618#   6690    6812    6936    6956    6977    7136    7173    7441    7536
                        7542    7546    7562    7595    7599    7632    7662    8534
RPAS  = 000016          6643#   6774*   7215    7334*   7361*   7378*   7483
RPBA  = 000004          2540*   6638#
RPCA  = 000034          2495*   2690*   2904*   2892*   2910*   2990*   3008*   5188*   6650#   6948    6960    6991
RPCC  = 000036          6651#   7139
RPCS1 = 000000          2497*   2538*   2549*   2893*   2911*   2991*   3009*   6636#   6732*   6738    6763    6890    6952
                        6973    7015    7035    7046    7061    7072    7195    7218    7270    7302    7339    7493    7590
RPCS2 = 000010          2385*   2493*   2494*   2509*   2510*   2519*   2585*   2586*   2715*   2716*   2729*   2730*   2744*
                        2745*   2749*   2750*   2814*   2815*   2838*   2834*   2843*   2844*   2902*   2903*   2920*   2921*
                        2937*   2938*   29 2*   2943*   3000*   3001*   3018*   3019*   3035*   3036*   3040*   3041*   5162*
                        6640#   6691#   6731#   6733    6825*   6937*   6957*   6978*   7064    7063    7095    7108    7117*
                        7137*   7193*   7284*   7364*   7377*   7442*   7458*   7543    7600    7633*   7639    7663*   7668
RPDA  = 000006          2541*   2693*   5187*   5371    6639#   6944    6969    6984
RPDB  = 000022          6645#   7637
RPDS1 = 000012          2499    2504    2710    2724    2809    2897    2915    2995    3013    6641#   6770    6874    6876
                        7286    7311    7365    7444
RPDT  = 000026          6647#   6741
RPEC1 = 000044          6654#
RPEC2 = 000046          6655#   7647
RPERRS  034316          6638#   6674    7253    7256    7266    7365*   7366*   7367*   7368*   8258    8265
RPER1 = 000014          4944    4846    4887    4889    6642#   6776    7296    7366    7604
RPER2 = 000040          6652#   7367
RPER3 = 000042          6653#   7368
RPINIT  034502          1797    6670#
RPLA  = 000020          2387    2520    6644#   7153
RPMR  = 000024          6646#
RPOF  = 000032          6649#   6767    6997    7001    7021    7025
```

```
RPSN  = 000C?0  6648#
RPT     001506   619#  2030  2064  2102  2145  2185  2265  2308  2358  2458  2615  2681  2785
                 2875  2973  3087  3158  3287  3407  3496  6080  6113*  6118  6144*  6205*  6209*
RPTMR   040704  4662  7408#
RPVEC   034466  2491#  2511#  2704#  2752#  2802#  2846#  2890#  2945#  2988#  3043#  5160#  6619#  6687
                 6689  6806
RPWC  = 000002  2539#  6637#
RP.REG  004204  1255  1279  1303  1327  1340#  1939  4844  4846  4387  4889  4943  5371  8280
                 8287  8290  8294  8297  8300  8304  8306  8310  8312
RP04    035252  4796  4827  4870  4917  4949  4968  4974  5167  6805#
?P04B   043162  1831  7874#
RP05    043167  1834  7875#
RP06    043174  1837  7876#
RSTRT1  006036  1894  1896  1898#  1908
?STRT2  006062  1903#  1909
RTC   = 000117   449#
SAVCSW  001222   526#  1847  1851*  5782*
SAVEFG  034424  1798*  6569#  7050  7199  7358
SAVFSG= 104412  2712  2726  2741  2811  2835  2899  2917  2934  2997  3015  3032  3637  4349#
                [illegible]  6868  7082  7172  7411  7629  7735
SC      037552  7178  7203  7205  7211  7215#
SCTRMC= 177400   465#  3117  3167  3431
SC1   = 000100   346#
SC10  = 001000   349#
SC11    040404  7255  7292  7349#
SC12    040474  7249  7260  7364#
SC13    040544  7239  7242  7374#
SC2   = 000200   347#
SC20  = 002000   350#
SC3     037622  7230#  7234
SC4     037626  7232#  7263  7275  7279  7327  7341  7348  7363  7401
SC5     037640  7231  7237#
SC6     040040  7259  7280#
SC6A    040150  7257  7304#
SC7     040276  7294  7307  7309  7319  7329#
SC8     040354  7271  7287  7297  7303  7312  7340  7342#  7373
SEARCH= 000131   452#  2707  2720
SEC.DS  001272   546#  4805*  4936*  4879*  4928*  4958*  4984*  5134*  5176*  5609*  8270  8273  8275
                 8301  8307
SEC.RD  001266   544#  5133*  8273
SEEK  = 000106   444#  2361  2462  2497  2805  2893  2911  2991  3009  3501  3506  4784  4785
SEEKFG  034426  1865#  1868#  6577#  6675  6900
SEKCNT  001346   565#  2552  2554#  2563#
SEKTMR  001344   564#  2492#  2580#
SELDRV= 000145   461#
SERIAL  043351  1937  7898#
SETBUF  030330  2472  3340  5518#
SETFOR= 000143   460#  1927  1929  2032  2361  2462  3501  3506
SETVEC  005326  1797#
SET.IE  042122  6704  6735  6843  6917  7122  7224  7661#
SKI   = 040000   420#  430#
SLASH   043044  5836  6083  7857#
SRCHMT  034400  6523#  6870*  6975*  7353
SRCHOO  026640  2687  2791  2881  2979  5158#
SR1DRV  005562  1802  1845#
```

```
SRTINT  005274          1789    1791#
SRVCLK  024310          4646    4656    4661#
STACK = 001100            91#   1725    1905    2038    2074    2110    2118    2153    2157    2198    2220    2274    2318
                        2366    2382    2385    2480    2518    2625    2700    2796    2886    2984    3091    3099    3104
                        3107    3115    3122    3162    3171    3181    3198    3204    3315    3344    3350    3360    3422
                        3434    3443    3449    3457    3463    3471    3499
STALL   026334          2203    2209    2225    2231    3503    3508    4859    4902    5000    5081#
STALL0  001336           562#   2196*   2204    2210    2211*   2212    2214*   2226    2232    2233*   2234    2236*
STALL1  001354           568#   4860    4903
STALL2  001356           569#   5001
STALL3  001360           570#   3504    3509
START   004672          1713    1718#
START1  004636            80    1711#
START2  004660            82    1716#   1900    3989    4072
START3  004624            84    1709#
START4  004650            86    1714#
STATBL  004254          1365#   3664
STKLMT= 177774           102#
ST0     040776          7418    7439#
ST01    041026          7448#   7491    7496
ST02    041224          7447    7449    7451    7483#
ST03    041274          7485    7497#
ST05    041320          7445    7464    7494    7503#
ST06    041326          7487    7505#
ST07    041364          7489    7513#
ST08    041414          7512    7519#
ST09    041424          7482    7500    7502    7504    7510    7517    7521#
STRTMR  027024          2702    2800    2888    2986    5200#
STRT1A  004642          1710    1712#
STRT2A  004664          1715    1717#
ST.CLK  024044          1796    2751    2845    2944    3042    4613#
ST.LCL  024254          4628    4653#
ST.PCL  024212          4622    4643#
SVAOR   001340           563#   3436*   3437*   3452    3453    3466    3467
SVRH11  042004          2507    2583    2714    2728    2743    2813    2837    2901    2919    2936    2999    3017    3034
                        7052    7069    7100    7201    7208    7248    7299    7317    7360    7396    7457    7520    7629#
SVSTAT  001260           540#   3661    5040*
SWR     001140           495#   1723    1744#   1746    1752#   1782    3336    3341    3347    3355    3357    3363    3590
                        3599    3606    3611    3615    3992    4031    4086#   4219    4237    4244    4945    5011    5143
                        5265    5502    5504    5508    5619    5678    5682    5771
SWREG   000176            64#   1752    1782    3992    4031    4054
SW0   = 000001           155#
SW00  = 000001           145#    155    3355
SW01  = 000002           144#    154    3363    5504    5619    5682
SW02  = 000004           143#    153    3357
SW03  = 000010           142#    152    3338    3347
SW04  = 000020           141#    151    3338    3341    4945
SW05  = 000040           140#    150    4644    4654    5508
SW06  = 000100           139#    149    2697    2753    4634    5265
SW07  = 000200           138#    148    4779    5771
SW08  = 000400           137#    147    1866    3590
SW09  = 001000           136#    146    5502    5678
SW1   = 000002           154#
SW10  = 002000           135#
SW11  = 004000           134#
SW12  = 010000           133#   2192
```

```
SW13  = 020000          132#   5082
SW14  = 040000          131#   4857    4900    4998
SW15  = 100000          130#   3417
SW2   = 000004          153#
SW3   = 000010          152#
SW4   = 000020          151#
SW5   = 000040          150#
SW6   = 000100          149#
SW7   = 000200          148#
SW8   = 000400          147#
SW9   = 001000          146#   5011    5143
SYSTAT  043050         1806    7858#
TAP   = 040000          335#
TBITVE= 000014          188#
TD      037412         7176    7185#
TDF   = 000040          364#    382#
TESTNG  043327         1931    7894#
TEST0   006470         2028    2037    2038#
TEST1   006646         2062    2073    2074#
TEST10  011116         2456    2478#   2566    2579
TEST11  012154         2613    2625#
TEST12  012422         2698    2700#
TEST13  013116         2796#
TEST14  013562         2894#
TEST15  014300         2984#
TEST16  014766         3085    3090    3091#
TEST17  015364         3156    3161    3162#
TEST2   006774         2100    2108#
TEST20  016112         3285    3315#
TEST21  016700         3405    3432    3433    3434#
TEST22  017274         3494    3438    3499#
TEST3   007212         2143    2151#
TEST4   007422         2183    2193    2195#
TEST5   007774         2263    2271#
TEST6   010174         2306    2314#
TEST7   010452         2356    2365    2366#
TICKMS  001246          535#   4636*   4639*   4661
TICKUS  001250          536#   4637*   4640*
TIMER   034430         2547*   6582#   6789*   6912*   7055*   7101*   7188*   7281*   7325*   7375*   7382*   7414    7416*
                       7474*   7501*   7507*   7513*
TIM.DN  001314          554#   5208*   5225
TIM.PT  001332          560#   5202    5243    5245*   5246*
TIM.UP  001276          548#   5201    5207*   5222    5271
TITLE   047662         1764    8420#
TKVEC = 000060          195#   3969*   3970*
TPB     001416          586#   1761    3623
TPS     001414          585#   1760    3622
TPVEC = 000064          196#
TRAPVE= 000034          194#   1731*   1732*
TRCKWC  001352          567#     96    3124    3414    4693*   4697*
TRE   = 040000          212#
TRK.DS  001274          547#    204*   4835*   4878*   4927*   4957*   4983*   5135*   5175*   5494*   5608*   8270    8273
                       8275     233    8301    8307
TRK.RD  001264          543#   5132*   8273
TRK1  = 004000          351#
TRK10 = 040000          354#
```

```
TRK2  = 010000            352#
TRK20 = 100000            355#
TRK4  = 020000            353#
TRNSWT  034376           2543#  6515#  6934#  7079#  7089   7106#  7115#  7190   7191#  7452   7467   7480#
TRTVEC= 000C14            189#
TSTNMS  001234            531#  1895   2022   2056   2094   2137   2177   2257   2300   2350   2450   2607   2673
                         2777   2867   2965   3079   3150   3279   3399   3488   4676#  4677#  5878#  5879#  5898#
                         5899#  5904#  5908#  5912#  5916#  5920#  5942#  5973#  5983   5985
TST0    006344           2020#
TST1    006506           2024   2054#
TST10   010710           2352   2448#
TST10A  011354           2513   2516#
TST10B  011746           2490   2580#
TST11   012014           2452   2605#
TST12   012234           2609   2671#  2679
TST13   012772           2675   2695   2775#  2783
TST14   013436           2779   2789   2865#  2873
TST15   014154           2959   2879   2963#  2971
TST16   014672           2967   2977   3077#
TST17   015270           3081   3148#
TST2    006664           2058   2092#
TST20   015710           3152   3277#
TST21   016430           3291   3397#
TST22   017206           3401   3495#
TST3    007102           2096   2135#
TST4    007276           2139   2175#
TST5    007664           2179   2255#
TST6    010064           2259   22  3#
TST7    010330           2302   2348#
TUF   = 000100            365#   383#
TWOMS   026416           2819   2908   2926   3006   3024   5106#
TYPOS = 104405           3709   4342#
TYPE  = 104401           1764   1769   1776   1791   1805   1806   1813   1814   1821   1823   1825   1827   1829
                         1830   1838   1840   1872   1877   1889   1897   1931   1936   1937   1952   1955   3523
                         3529   3537   3554   3558   3601   3609   3656   3657   3666   3675   3676   3680   3683
                         3689   3690   3699   3712   3716   3719   3721   3723   3726   3765   3862   3938   3986
                         3998   4052   4053   4056   4067   4077   4088   4107   4155   4161   4166   4170   4175
                         4176   4178   4181   4185   4338#  4452   5272   5276   5280   5283   5287   5291   5292
                         5296   5299   5303   5307   5308   5319   5320   5324   5329   5773   5802   5806   5853
                         5836   6045   6054   6075   6077   6079   6083   6115   6116   6117   6120   6151   6158
TYPERR  020032           3608   3636#
TYPOC = 104402           3542   3705   4055   4339#  5805   6119   6160
TYPON = 104404           4341#
TYPOS = 104403           1810   1894   1933   3534   4340#  6051
TYPTIM  027222           2755   2759   2847   2946   3044   5264#
T10     003014            935#  2794   2848
T11     003024            940#  2884   2947
T12     003034            945#  2932   3045
T7A     002774            925#  2636   2756
T7B     003004            930#  2699   2760
ULDFLG  034404           6542#  6730#  6815   6830#  7012#  7075#  7107#  7245   7331   7333#  7349   7351#  7460#
                         7473#
UNLOAD= 000103            443#
UNS   = 040000            295#
UNTMSG  043070           7861#
UNTOFF  043076           1825   7862#
```

```
UNTON   043107          1829    7864#
UPE     = 020000         236#
US1     = 000001         223#
US2     = 000002         224#
US4     = 000004         225#
UWR     = 000010         417#
VERIFY  026476          4854    4897    5124#
VUF     = 000002         416#
VU30    = 010000         371#
VV      = 000100         268#
WA0     = 000002         426#
WCE     = 040000         237#
WCEFLG  001334          5611    3316#   3353    4916#   4980#
WCF     = 000040         286#
WCU     = 000001         359#    377#
WLE     = 004000         292#
WRCXD   = 000151         453#    3102    3123    3182    3205    3351    3454    3468    4941    4967
WRCXHD= 000153           454#
WRITE   = 000161         455#    2463    3100    3116    3164    3199    3315    3416    3438    4947
WRL     = 004000         273#
WRTHD   = 000163         456#
WRT.A0  041702          7585#   7595#   7598#   8263
WRT.RP  041612          6762    6766    6889    6943    6947    6951    6959    6968    6972    6983    6990    7000    7014
                        7024    7045    7060    7071    7269    7301    7338    7582#
WRT.R1  041676          7596#   7516
WRT.R2  041766          7582#   7615#
WRT.R3  041774          7591    7602    7605    7611    7618#
WRT.R4  042000          7607    7620#
WRT.R5  042002          7619    7621#
WRT.W0  041700          7583#   7587    7594#   7597#   8263
WRU     = 000400         367#    395#
WSU     = 000004         361#    379#
$AUTOB  001134           492#   1786#   4045    4202
$BOOOR  001122           487#   5401#   5666#   8278
$BOOAT  001126           489#   5490#   5603#   5668#   8278    8285
$BELL   001210           518#   1758    3601    3625    3998    4195
$CHARC  020634          3767#   3777#   3784    3793#   3798#
$CKSUM  021610          4031#   4346
$CNTAG  001100           475#   1720    1721    1729    1735    1736
$CM1    = 000006         507#    508#    509#    510#    511#    512#    513#
$CM2    = 000014         507#    508#    509#    510#    511#    512#    513#
$CM3    = 000006         505#    507
$CMM    = 000003         513#    514#    515#    516#
$CNTLC  022522          3686    4067    4195#
$CNTLG  022534          4052    4197#
$CNTLU  022527          4077    4170    4196#
$CRLF   001215           520#   1791    1805    1840    1955    3609    3625    3656    3675    3689    3699    3716    3723
                        3765    3801    4068    4175    4195    6054
$DCLK   021302          3904    3938    3946#
$DB2D   023224          4365    4384#
$DECVL  023404          4386    4432#
$DIV    023562          2376    2573    4520#   5312
$DOAGN  017614          3550    3560    3566#
$DTBL   021272          3907    3942#
$ENDAD  017604            72    3562#
$ENDCT  017540          1873#   1882#   1892    3552#
```

```
$ENULL  017620          3558    3569#
$EOP    017356          1879    1922    3490    3522#   5018    8545
$EOPCT  017532          1892*   3549#   3553    8544*
$ERFLG  001103           478#   3596*   3625    4209    4233    4235    4241*   4260    4995    5475    5506    5621    5680
$ERMAX  001115           484#   1737*   1956*   2031*   2065*   2103*   2146*   2186*   2266*   2309*   2359*   2459*   2616*
                        2682*   2786*   2876*   2974*   3088*   3159*   3288*   3408*   3497*   4235    4255*   4260
$ERROR  017624          1729    3588#
$ERRPC  001116           485#   3603*   3604*   3605    3625    8257    8258    8261    8263    8265    8268    8269    8270
                        8275    8280    8283    8287    8292    8294    8297    8301    8307
$ERRTB  004306          1392#   3552    3553
$ERTTL  001112           482#   3541    3543*   3602*   3625
$ESCAP  001206           517#   1736*   2694*   2795*   2885*   2983*   3618    3620    3625    4254*   4795*   4802*   4826*
                        4833*   4869*   4876*   4915*   4925*   4948*   4955*   4981*   5014*   5137*   5142*   5166*   5173*
                        5495*   5500*   5611*   5617*   5664*   7222    7273    7277    7552    7612    7821    7830    7831*
                        7833*
$FILLC  001156           503#   3770    3801
$FILLS  001155           502#   3801
$GDAOR  001120           486#   5602*   5667*   8278
$GODAT  001124           488#   5604*   5669*   8278
$GET42  017574          3559#
$GTSWR  021720          4053#   4344
$HD     = 000000          13
$HIMUM  023556          1733*   2371    2569    3411*   4474    4482    4487*   4492#   5651    5657*   5660*   5704    5720
$ICNT   001104           479#   1794*   4246*   4247    4249*   4259
$INTAG  001135           493#   4050*   4069    4089    4202
$ITEMB  001114           463#   3605*   3625    3645
$LF     001216           521#   3625    3801    4185    4195
$LONUM  023560          1734*   3412*   4473    4480    4486*   4493#   5085    5652    5656*   5659*   5703    5718    5719
$LPAOR  001106           480#   1738*   2037*   2073*   2104*   2147*   2187*   2267*   2310*   2365*   2460*   2617*   2686*
                        2790*   2880*   2978*   3090*   3161*   3289*   3409*   3432*   3498*   4239*   4252*   4257    4259
$LPERR  001110           481#   1739*   2028*   2062*   2100*   2109*   2117*   2143*   2152*   2156*   2183*   2197*   2219*
                        2263*   2273*   2306*   2317*   2356*   2381*   2384*   2456*   2479*   2517*   2613*   2679*   2783*
                        2873*   2971*   3085*   3098*   3103*   3106*   3114*   3121*   3156*   3170*   3180*   3197*   3203*
                        3285*   3343*   3349*   3359*   3405*   3421*   3433*   3442*   3448*   3456*   3462*   3470*   3494*
                        3617    4239    4253*   4259    5013    5145
$LSTAD  050352          1774    8492*   8496#
$MAIL   = ****** U      1756    1782    3611    3754    4252
$MNEW   022552          4056    4200#
$MSWR   022541          4053    4198#
$MXCNT  023006          4250    4259#
$NULL   001154           501#   3772    3801
$NWTST= 000001          2011#   2013    2043#   2045    2079#   2081    2126#   2128    2165#   2167    2242#   2244    2287#
                        2289    2337#   2339    2413#   2415    2592#   2594    2659#   2661    2764#   2766    2852#   2854
                        2950#   2952    3064#   3066    3134#   3136    3215#   3217    3372#   3374    3477#   3479
$OCNT   021062          3834*   3863*   3876#
$OMODE  021064          3829*   3833*   3938    3841*   3852*   3878#
$OVER   022772          4220    4240    4248    4256#
$PASS   001100           476#   3546*   3547*   3569
$QUES   001214           519#   3625    3801    4107    4178    4195
$RAND   023460          2370    2568    4469#   5084    5702    5717
$ROCNR  022152          4120#   4347
$RODEC= ****** U        4319
$ROLIN  022242          4143#   4348
$RODCT= ****** U        4349
$ROSZ   = 000024        4136#
$REGAD  001160           505#   1756
```

```
$REG0   001162          507#  3639*  8270  8275  8283  8292  8294  8297  8301  8307
$REG1   001164          508#  3640*  8258  8265  8285
$REG2   001166          509#  3641*  8269
$REG3   001170          510#  3642*  8257  8258
$REG4   001172          511#  3643*  8278  8285
$REG5   001174          512#  3644*  8265
$RESRE  023046         4294#  4350
$RTNAD  017616         3568#
$R2A  = ******  U      4351
$SAVRE  023010         4278#  4349
$SB2D   023170         4363#  5278  5285  5289  5294  5301  5305  5317  5322  6081  6156
$SCOPE  022564          1727  4217#
$SETUP= 000147          1709#  1726  1727  1729  1731  1733  1735  1736  1738  1779  3544  3589  3614
                        3628   3990  3995  3996  4026  4202  4218
$SIZE   050256          177?   8475#
$STUP = 177777          1705#
$SUPRS  023420         4444#  5279  5286  5290  5295  5302  5306  5318  5323  6082  6157
$SVLAD  022744          4228   4259#
$SVPC = 000200           70#    75
$SWR  = 167000            3#    13    41    42    43    44    45    46   468#   516   517   518  1735
                        1736   1753  1739  2032  2066  2104  2147  2187  2267  2310  2360  2460  2617
                        2683   2787  2877  2975  3089  3160  3289  3409  3498  3519  3545  3561  3567
                        3569   3580  3581  3582  3583  3584  3599  3606  3611  3615  3625  4210  4211
                        4212   4213  4214  4219  4231  4233  4234  4235  4242  4243  4244  4253  4256
                        4259
$SWPMK= 000000          4214
$TIMES  001204          516#  1735*  2030*  2064*  2102*  2145*  2185*  2265*  2308*  2358*  2458*  2615*  2681*
                       2785*  2875*  2973*  3087*  3158*  3287*  3407*  3496*  3545*  4242*  4247   4250*  4259
$TKB    001146          498#  3971   3971   3982   4007   4035   4062
$TKCNT  021312         3952#  3966*  3996   4013*  4127   4129*
$TKINT  021322          1778  3966#  3987   4048
$TKQEN= 021322         3956#  4021   4132
$TKQIN  021314         3953#  3967*  3968   4019*  4020*  4021   4023*
$TKQOU  021316         3954#  3968*  4130   4131*  4132   4134*
$TKXSR  021320         3955#  3967   4023   4134
$TKS    001144          497#  3951   3972*  4003*  4005   4011*  4033  4049*  4059  4071*  4091*
$TKSRV  021372         3969   3982#
$TMP0   001176          513#  3636*  8261  8263  8265  8270  8275  8280  8283  8287  8292  8294  8297
                       8301   8307
$TMP1   001200          514#
$TMP2   001202          515#
$TN   = 000023            3#    13  2011  2020  2022  2031  2032#  2043  2054  2056  2065  2066#  2079
                       2092   2094  2103  2104#  2126  2135  2137  2146  2147#  2165  2175  2177  2186
                       2187#  2242  2255  2257  2266  2267#  2287  2298  2300  2309  2310#  2337  2348
                       2350   2359  2360#  2413  2448  2450  2459  2460#  2592  2605  2607  2616  2617#
                       2659   2671  2673  2682  2683#  2764  2775  2777  2786  2787#  2852  2865  2867
                       2876   2877#  2950  2963  2965  2974  2975#  3064  3077  3079  3088  3089#  3134
                       3148   3150  3159  3160#  3215  3277  3279  3288  3289#  3372  3397  3399  3408
                       3409#  3477  3486  3488  3497  3498#  5933  5957  6032
$TMPWR  023334         4391   4392  4412#
$TPB    001152          500#  1761*  3595*  3623*  3790*  380'
$TPFLG  001157          504#  3748   3801
$TPS    001150          499#  1760*  3594*  3622*  3788   3801
$TRAP   023104         1731   4215#
$TRAP2  023126         4326#  4337
$TRP  = 000014         4330#  4339#  4340#  4341#  4342#  4343#  4344  4345#  4346  4347#  4348#  4349#  4350#
```

CROSS REFERENCE TABLE -- USER SYMBOLS

```
                        4351#
$TRPAD  023140          4320    4337#
$TSTNM  001102           477#   2025*   2029    2059*   2063    2097*   2101    2140*   2144    2180*   2184    2260*   2264
                        2303*   2307    2353*   2357    2453*   2457    2610*   2614    2676*   2680    2780*   2784    2870*
                        2874    2968*   2972    3082*   3086    3153*   3157    3282*   3286    3402*   3406    3491*   3495
                        3544*   3598    3625    3636    4209    4251*   4256    4260    4735    4937
$TTYIN  022476          4145    4146    4158    4176    4190    4194#
$TYPBN= ****** U        4343
$TYPOS  021066          3892#   4342
$TYPE   020420          3748#   4330    4338
$TYPEC  020570          3769    3776    3783    3788#   3789    4093
$TYPEX  020636          3794    3796    3799#
$TYPOC  020664          3832#   4339
$TYPON  020700          3831    3834#   4341
$TYPOS  020640          3827#   4340
$XTSTR  022576          4222#
$$GET4= 000000          3561#
$OFILL  021063          3828*   3832*   3842    3877#
$40CAT= ****** U        3608    4219
.       = 050554          58#     62#     70      71#     73#     75#     78#     474#    522     1724    1738    1739    2109
                        2117    2152    2156    2197    2219    2273    2317    2381    2384    2479    2517    3098    3103
                        3106    3114    3121    3170    3180    3197    3203    3289    3343    3349    3359    3409    3421
                        3442    3448    3456    3462    3470    3526#   3569    3570#   3625    3669#   3801    3946#   3951
                        3955#   3956    3957    4194#   4195    4202#   4259    4260    4432#   6048#   7720#   7721#   7722#
                        7723#   7724#   7725#   7726#   7727#   7728    8418
```

CROSS REFERENCE TABLE -- MACRO NAMES

```
CKCHR    1705#  5809   5859   6086   6123   6163   6326   6339   6350   6401
CKDIG    1705#  5867   6105
CKNUM    1705#  5817   6138   6171
COMMEN    198#  1960   1995   2641   3050
COMND    1705#  1929   2032   2361   2461   3501   3506
DO       1705#  1928   1930   2039   2040   2075   2076   2112   2120   2154   2159   2200   2206   2222   2228
         2276   2278   2322   2324   2326   2328   2330   2332   2383   2404   2626   2628   3502   3507   5141
DODTA    1705#  3101   3105   3110   3120   3127   3173   3184   3200   3206   3346   3352   3362   3423   3444
         3450   3458   3464   3472
DRV.IN   1705#  2748   2842   2941   3039
ENDCOM    198#  1988   2006   2654   3059
ENDPAS   1705#  3554
ERRCAL   6421#  7222   7273   7276   7551   7612
ERREND   1705#  3622
ERROR     92#  1914   2506   2584   2718   2732   2747   2817   2841   2905   2923   2940   3003   3021   3028
         4809   4810   4811   4812   4813   4840   4841   4842   4843   4848   4883   4884   4885   4682   4691
         4932   4933   4934   4935   4936   4962   4963   4964   4965   4966   4988   4989   4990   4991   4992
         5139   5180   5181   5182   5183   5184   5496   5501   5612   5618   5676   5677   7223   7274   7278
         7553   7613   8540
ERRTYP   1705#  3590
ER.NDX   1705#  4803   4834   4877   4926   4956   4982   5174
ESCAPE    198#  2694   2795   2885   2983   4802   4833   4876   4924   4955   4981   5137   5173   5495   5539
         5611   5616   5664
GETPRI    198#  4520   8481
GETSWR      3#   198#  1779
LOOP     1705#  2109   2117   2152   2156   2197   2219   2273   2317   2380   2394   2478   2516   3098   3103
         3106   3114   3121   3170   3180   3197   3203   3343   3349   3359   3421   3442   3448   3456   3462
         3470
MORETA    468#   523
MORE.S   1705#  2060   2054   2092   2135   2175   2255   2298   2348   2448   2605   2671   2775   2865   2963
         3077   3148   3277   3397   3486
MSG      2011#  2013   2743#  2045   2079#  2081   2126#  2128   2165#  2167   2242#  2244   2287#  2289   2337#
         2339   2412#  2415   2592#  2594   2658#  2661   2764#  2766   2852#  2854   2950#  2952   3064#  3066
         3134#  3136   3215#  3217   3372#  3374   3477#  3479
MULT      198#
NEWTST    198#  2011   2043   2079   2126   2165   2242   2287   2337   2413   2592   2659   2764   2852   2950
         3064   3134   3215   3372   3477
POP       198#  3333   4299   4488   4571   4716   4765
PUSH      198#  3892   4279   4469   4523   4671   4729
REPORT    198#  1705#  2755   2758   2847   2946   3044
RP04.D      2#  6421
SAV.RH   1705#  2712   2726   2741   2811   2835   2899   2917   2934   2997   3015   3032
SCOPE      93#  2041   2077   2124   2163   2240   2285   2335   2410   2590   2636   2761   2849   2948   3046
         3132   3212   3369   3473   3510
SETPRI    198#  4123
SETTRA   4330#  4339   4340   4341   4342   4344   4346   4347   4348   4349   4350
SETUP     198#  1719
SET.TN   1705#  2021   2055   2093   2136   2176   2256   2299   2349   2449   2606   2672   2776   2866   2964
         3078   3149   3278   3398   3487
SKIP      198#
SLASH     198#
SPACE     198#
STAPS      68    198#   202    244    248    439    470    522   1607   2011   2019   2043   2053   2079   2091
         2126   2134   2165   2174   2242   2254   2287   2297   2337   2347   2413   2447   2592   2604   2659
         2670   2764   2774   2852   2864   2950   2962   3064   3076   3134   3147   3215   3276   3372   3396
         3477   3485   3516   3576   3626   3733   3804   3882   3950   4026   4041   4112   4136   4206   4263
```

```
        4309    4354    4373    4436    4460    4497    6420    6430    8469    8498
SWRSU    198#  1740#
TRMTRP  4330#
TYPBIN   198#
TYPB2D  1705#   5277    5284    5288    5293    5300    5304    5316    5321    6080    6155
TYPDEC   198#   3707
TYPEND  1705#   3523
TYPNAM   198#
TYPNUM   198#
TYPOCS   198#   1807    1883    1932    3533    6049
TYPOCT   198#   3541    3704    4054    5804    6118    6159
TYPTXT   198#   3523    3529    3537    3554    3666    3683    5773    5853    5885    6045    6150
$$CMRE   468#    507     508     509     510     511     512
$$CMTM   468#    513     514     515
$$ESCA   198#
$$NEWT   198#   2011    2043    2079    2126    2165    2242    2287    2337    2413    2592    2659    2764    2852    2950
        3064    3134    3215    3372    3477
$$SET   4330#   4339    4340    4341    4342    4344    4346    4347    4348    4349    4350
$$SKIP   198#
.EQUAT     3#     88
.HEADE     3#
.SETUP     3#   1709
.SWRHI     3#     37
.SWRLO    47#     48      49      50      51      52      53      54
.$ACTI     3#     66
.$CATC     3#     56
.$CMTA     3#    468
.$D82D     3#   4371
.$DIV      3#   4495
.$EOP      3#   3514
.$ERRO     3#   3574
.$RAND     3#   4458
.$RDDE     3#
.$RDOC     3#
.$READ     3#   3948
.$SAVR     3#   4267
.$SB2D     3#   4352
.$SCOP     3#   4214
.$SIZE     3#   8417
.$SUPR     3#   4474
.$TRAP     3#   4307
.$TYPD     3#   3830
.$TYPE     3#   3731
.$TYPO     3#   3802

. ABS.  050554     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 .NOW.SEQ/SOL/CRF/LI:ME/NL:TOC:MC:MD:CND=RP0456.010,DZRJA.014
 RUN-TIME: 89 92 10 SECONDS
 RUN-TIME RATIO: 710/192=3.6
 CORE USED:  46K  (91 PAGES)
```

Spooler runtime 30 Seconds, 187 KCS, 830 disk reads, 3 disk writes, 182 pages